პარიზ ფრანგიშვილი, ზურაბ მასიტაშვილი,
ილია აბულაძე, ვლადიმერ ნოღაება

იმიტაციური მოდელირების ენა GPSS World და რთული სისტემების მოდელები

„ტექნიკური უნივერსიტეტი"

საქართველოს ტექნიკური უნივერსიტეტი

არჩილ ფრანგიშვილი, ზურაბ გასიტაშვილი,
ინგა აბულაძე, ვლადიმერ წვერავა

იმიტაციური მოდელირების ენა
GPSS World
და რთული სისტემების მოდელები

დამტკიცებულია სტუ-ს
სარედაქციო-საგამომცემლო
საბჭოს მიერ

თბილისი
2009

შაკ 681.3:629

სერია
„მართვა, გამოთვლითი ტექნიკა, ინფორმატიზაცია“

განხილულია იმიტაციური მოდელირების ძირითადი ცნებები და მოდელირების სისტემის ობიექტები. მოცემულია **GPSS World** მოდელირების საერთო მიზნობრივი სისტემის აგების საფუძვლები და ფუნქციონირების პრინციპები. მნიშვნელოვანი ყურადღება ეთმობა **GPSS World** ენის გამოყენების თავისებურებებს რთული სისტემების მოდელირებისათვის. კომპიუტერზე გადასაწყვეტად მოყვანილია პრაქტიკულ ამოცანათა მოდელების პროგრამები და მათი შედეგების მიხედვით ხორციელდება გადაწყვეტილებათა მიღება. ინსტრუმენტულ საშუალებად გამოიყენება **GPSS World** მოდელირების ენა, რომელიც დღეისათვის ყველაზე პოპულარულ ენად ითვლება აშშ-ის და ევროპის მრავალი ქვეყნის წამყვან უნივერსიტეტებში.

განკუთვნილია სტუდენტების, მაგისტრანტებისა და დოქტორანტები-სათვის, აგრეთვე დიდ დახმარებას გაუწევს მოდელირების სფეროში დასაქმებულ სპეციალისტებს და, საერთოდ, თანამედროვე მოდელირების პრობლემებით დაინტერესებულ მკითხველთა ფართო წრეს.

სერიის რედაქტორი
საქ. მეცნიერებათა აკადემიის წევრ-კორესპონდენტი,
სრული პროფესორი **არჩილ ფრანგიშვილი**

რეცენზენტი სრული პროფესორი **ზურაბ ფვერაიძე**

ISBN 978-9941-14-278-9
http://www.gtu.ge/publishinghouse/

ა ვ ტ ო რ ე ბ ი ს გ ა ნ

წარმოდგენილი ნაშრომი ეძღვნება მოდელირების გამოყენების საკითხების განვითარებას მასობრივი მომსახურების სისტემების ამოცანების ეფექტურად გადასაწყვეტად, კერძოდ **GPSS World** მოდელირების ენის გამოყენებით.

ამჟამად რთული სისტემების მოდელირებისათვის ყველაზე დახვეწილ, მძლავრ და პროფესიულ სისტემად ითვლება **GPSS World.** ის წარმატებით ისწავლება ევროპისა და აშშ-ის მრავალ წამყვან უნივერსიტეტში.

ავტორებს ნაშრომში შემოთავაზებული აქვთ იმიტაციური მოდელირებისთვის **GPSS World** ენის გამოყენება. მასში აღწერილია **GPSS World** ენის ბლოკების დანიშნულება, გრაფიკულ ინტერფეისთან მუშაობა და მისი ბრძანებები. სახელმძღვანელოში მრავლადაა საილუსტრაციო მაგალითები, რაც ხელს შეუწყობს მასალის უკეთ აღქმასა და ათვისებას.

ნაშრომის ორიგინალობა მდგომარეობს იმაში, რომ მასში პირველად, ქართულ ენაზე მასალა გადმოცემულია მარტივი, პედაგოგიური სტილით.

ნაშრომი უდავოდ საინტერესო და სასარგებლო იქნება როგორც ინფორმატიკის, ასევე ტექნიკისა და ეკონომიკის მრავალი დარგის მეცნიერებისა და სპეციალისტებისთვის, რომლებიც მოღვაწეობენ მოდელირების სფეროში.

ავტორები მადლიერების გრძნობით მიიღებენ მკითხველთა შენიშვნებსა და მოსაზრებებს შემდეგ მისამართზე: zur_gas@ gtu.ge, i_abuladze@gtu.ge.

ს ა რ ჩ ე ვ ი

შ ე ს ა ვ ა ლ ი

მოდელირება წარმოადგენს სისტემათა ეფექტურობის კვლევისა და შეფასების მძლავრ, უნივერსალურ მეთოდს.

იმიტაციის მეთოდების გამოყენების სფერო ძალზე ფართო და მრავალფეროვანია. თუმცა, სამეცნიერო მუშაობა და აღნიშნულ დარგში ჩატარებული კვლევები ცხადყოფს, რომ მკვლევარები და მეცნიერები გამოკვლევათა ინსტრუმენტული საშუალებების სახით ძალზე იშვიათად მიმართავენ მოდელირების სისტემებს, რომელთა უპირეტესობაც, ნათლად მოგეხსენებათ აშკარაა [1, 12]. მოდელირების სისტემებს გააჩნიათ სპეციალიზებული საშუალებები, რომლებიც კომპიუტერზე სამოდელო ექსპერიმენტების ორგანიზაციის თვალსაზრისით ახდენს დამატებით შესაძლებლობათა რეალიზებას. ისინი ასევე საშუალებას გვაძლევს მოდელებში გავითვალისწინოთ დროის ფაქტორი, ანუ ავაგოთ დინამიკური იმიტაციური მოდელები, რაც ასე მნიშვნელოვანია მრავალი სისტემისათვის [13, 14, 15].

1960-იანი წლების ბოლოს აშშ-ში პირველად გამოიცა სახელმძღვანელო „მოდელირება **GPSS**-ზე“. 1976 წელს ფირმა **IBM**-მა (**International Business Machines**) შეიმუშავა **GPSS**-ის ახალი ვერსია სახელწოდებით **APL-GPSS**. **APL** (**APL – A Programming Language**) ენამ და **GPSS**-მა საშუალება მისცა გაერთიანებულიყო ორივე ენის უპირატესობანი, რომლის შედეგადაც მოდელების აგების დროს შესაძლებელი იყო გამოყენებულიყო როგორც **APL**-ის ფართო გამოთვლითი შესაძლებლობები, ასევე **GPSS**-ის მიერ წარმოდგენილი მოდელირებისათვის აუცილებელი მაკრო-

ბრძანებათა დიდი ნაკრები. სსრკ-ში იმიტაციური მოდელირების პირველი სახელმძღვანელო გამოიცა სსრკ მეცნიერებათა აკადემიის წევრ-კორესპონდენტ ნ. ბუსლენკოს მიერ, ხოლო **GPSS** ენის ათვისების ე.წ. „პირველობის პალმა“ ეკუთვნის ქ. მოსკოვის МИЭТ-ის ინსტიტუტის თანამშრომელს ა. დედკოვს. **GPSS** ენის ბაზაზე **IBM** ფირმაში დამუშავდა **GPSS/360** ვერსია, ხოლო მოგვიანებით – მისი გაფართოებული ვერსია **GPSS/V.** 1976 წელს პირველად სსრკ-ში ი. იაკიმოვის და ვ. დევიატკოვის ხელმძღვანელობით მუშა ჯგუფმა დაასრულა მუშაობა დისკრეტული სისტემების მოდელირების გამოყენებითი პროგრამული პაკეტის სამრეწველო გამოცემასა და ათვისებაზე, რომლის საფუძველსაც წარმოადგენდა **GPSS/360** სისტემა [8, 11].

1977 წელს აშშ-ში ჯ. ო. ჰენრიქსონმა, რომელიც იმ დროს წარმოადგენდა მათემატიკური უზრუნველყოფის სპეციალისტს, ხოლო მოგვიანებით კი კონსულტანტს მიჩიგანის უნივერსიტეტში, შეიმუშავა **GPSS**-ის ახალი ვერსია, რომელსაც ეწოდა **GPSS/H**. აღნიშნულ პროგრამულ სისტემას გააჩნდა **GPSS/360**-სა და **GPSS/V**-ის ყველა თვისება (ანუ **GPSS/360** და **GPSS/V** ვერსიები მთლიანად თანხვედრილია **GPSS/H** ვერსიასთან) და აღნიშნულ ვერსიებთან შედარებით უდიდესი უპირატესობა, რომელიც მომხმარებელს საშუალებას აძლევდა ინტერაქტიულ რეჟიმში ნაბიჯ-ნაბიჯ ეკონტროლებინა მოდელირების პროცესი [2]. 1993 წელს ელნისკმა დაამუშავა პროგრამა, როგორც **GPSS/H** – ის გარემო, რომლის საშუალებითაც შესაძლებელია როგორც მოდელის ტექსტის სწრაფად შეტანა, ასევე მისი გაფორმება.

1990-იანი წლების დასაწყისში გერმანულმა ფირმამ **Gfl** (აახენი) დაამუშავა **GPSS/H EDITOR**, რომლის საშუალებითაც შესაძლებელი იყო **GPSS/H** პროგრამის სწრაფად შეტანა, მოცემული ბლოკის დასახელებებით, ღილაკებზე უბრალო დაჭერით. შემდეგი ვერსია **GPSS/H + PROOF** კი წარმოადგენს დისკრეტული სისტემების მოდელირების ენას და ადვილია ასათვისებლად, ხოლო მასში ფუნქციების, ცვლადების, სტანდარტული ატრიბუტების, გრაფიკებისა და სტატისტიკური ბლოკების არსებობა მნიშვნელოვნად აფართოებს მის შესაძლებლობებს [8].

იმიტაციური მოდელების რეალიზაციის დროს პროგრამირების უნივერსალური ენების გამოყენება მოდელის შემქმნელს საშუალებას აძლევს მოდელის დამუშავების, გამართვისა და გაშვების დროს მიაღწიოს მის მოქნილობას. თუმცა, მოდელირების ენები, რომლებიც ორიენტირებულია ცალკეულ საგნობრივ დარგზე, წარმოადგენს გაცილებით უფრო მაღალი დონის ენებს, ამიტომაც იძლევა საშუალებას მცირედი დანახარჯებით რთული სისტემების გამოსაკვლევად შეიქმნას მოდელირების პროგრამები.

მოდელირების სპეციალიზებული ენები იყოფა იმიტაციის სახეთა შესაბამის სამ ჯგუფად: უწყვეტი, დისკრეტული და კომბინირებული პროცესებისათვის. დისკრეტული სისტემების მოდელირებისათვის ფართო გავრცელება ჰპოვა დისკრეტული სისტემების მოდელირების პაკეტმა – **GPSS World (General Purpose**

Simulation System) – მოდელირების საერთო მიზნობრივმა სისტემამ [2]. იგი დამუშავებულია კომპანია **Minuteman**-ის მიერ (აშშ) [3]. **GPSS World** ენაზე დაწერილი მოდელები საკვლევ სისტემაში გაცილებით უფრო ადექვატურია, ვიდრე ცნობილი მასობრივი მომსახურების თეორიის გამოყენებით აგებული მოდელები. **GPSS World** ენაზე დაწერილ მოდელებში შესაძლებელია გათვალისწინებულ იქნას ფაქტორების დიდი რაოდენობა და ამასთანავე შეიძლება უკუიგდოს მრავალი შეზღუდვა და დაშვება.

GPSS/PC არის დისკრეტული მოდელირების ენა, **GPSS World** კი წარმოადგენს **GPSS/PC**-ის განვითარებულ ვერსიას [4, 5, 6, 7]. აღნიშნულმა სისტემამ შეიძინა კომბინირებული ხასიათი, ანუ შეუძლია შექმნას მოდელი, როგორც დისკრეტული, ასევე უწყვეტი სისტემებისათვის.

დაბოლოს, **GPSS World**-ის მოდელირების სისტემა მუშაობს **Windows** გრაფიკულ-ოპერაციულ სისტემაში და მაქსიმალურად ორიენტირებულია თანამედროვე ტექნოლოგიათა გამოყენებაზე, რომელიც უზრუნველყოფს ინფორმაციის მაღალ ინტერაქტიულობასა და ვიზუალურ წარმოდგენას.

I თავი

GPSS World იმიტაციური მოდელირების ენის ძირითადი ცნებები

1.1. GPSS World მოდელირების სისტემის ობიექტები

GPSS World მოდელირების სისტემაში განასხვავებენ ოთხი ტიპის ობიექტს: მოდელი, მოდელირების პროცესი, ანგარიში და ტექსტი.

მოდელი **GPSS** ენაზე შედგება ბლოკებისა და ოპერატორებისაგან; ხოლო *მოდელირების პროცესი* წარმოადგენს მოდელის ტრანსლაციის შედეგს, რომელიც მიიღება **Command ► Greate Simulation** მენიუს ბრძანების შესრულების შედეგად. მოდელირების პროცესის დასრულებისთანავე, როგორც წესი, ავტომატურად იქმნება ობიექტი *ანგარიში.*

ტექსტური ობიექტი (**GPSS World**-ის ტექსტური ფაილი) განკუთვნილია დიდი მოდელების დამუშავების გამარტივებისა და საწყისი ტექსტების ბიბლიოთეკის შესაქმნელად, ანუ მოდელი შეიძლება დაყოფილ იქნას ოპერატორების ნაკრებებად, რომლებიც წარმოადგენს ცალკეულ ტექსტურ ფაილებს, ხოლო ამის შემდეგ მათგან შექმნილია ობიექტი „მოდელირების პროცესი“. ობიექტს „მოდელირების პროცესი“ ასევე შეუძლია შექმნას ახალი

ტექსტური ფაილები მოდელის ფრაგმენტებით, მოდელირების შედეგებით, აგრეთვე წაიკითხოს და ჩაწეროს მონაცემები ტექსტურ ფაილებში.

1.2. იმიტაციური მოდელირების ენის ობიექტები

GPSS World სისტემა განკუთვნილია დისკრეტული და უწყვეტი პროცესების იმიტაციური მოდელირებისათვის.

მოდელირების სისტემებში ობიექტები განკუთვნილია სხვადასხვა დანიშნულებისათვის. ობიექტების არჩევა კონკრეტულ სამოდელო სისტემაში დამოკიდებულია მოდელის მახასიათებლებსა და სპეციალისტზე, რომელიც ადგენს მოდელს. არ არის აუცილებელი, რომ ერთ მოდელში მონაწილეობდეს ობიექტების ყველა ტიპი, მაგრამ ყველა მოდელს უნდა გააჩნდეს ბლოკები და ტრანზაქტები.

ობიექტები იყოფა 7 კატეგორიად და 15 ტიპად, რომლებიც წარმოდგენილია 1.1 ცხრილში. განვიხილოთ **GPSS** ობიექტების დანიშნულება.

ტრანზაქტები წარმოადგენს ***დინამიკურ ობიექტს***, რომლებიც იქმნება მოდელის განსაზღვრულ წერტილებში, გადაადგილდება მოდელში ბლოკიდან ბლოკში და ბოლოს ქრება. ტრანზაქტები ხასიათდება პარამეტრებით, რომლებიც გამოიყენება კონკრეტული მონაცემებისათვის. თითოეულ ტრანზაქტს შეიძლება გააჩნდეს პარამეტრების ნებისმიერი რიცხვი. აგრეთვე, ტრანზაქტს შეიძლება მივანიჭოთ პრიორიტეტი. პრიორიტეტით სარგებლობა შეუძლიათ,

მაშინ როდესაც ერთსა და იმავე რესურსზე პრეტენდენტობს რამდენიმე ტრანზაქტი. მაგალითად, ტრანზაქტი საპარიკმახერო სალონში შეიძლება იყოს კლიენტი, საავტომობილო გზატკეცილზე – ავტომობილი, მაღაზიაში – მყიდველი, ტელევიზორების სახელოსნოში – ტელევიზორები და ა.შ.

ცხრილი 1.1

GPSS–ის ობიექტები

კატეგორია	ობიექტების ტიპები
დინამიკური	ტრანზაქტები
ოპერაციული	ბლოკები
აპარატურული	ერთარხიანი მოწყობილობები, მრავალარხიანი მოწყობილობები (მეხსიერებები) და ლოგიკური გასაღებები
გამოთვლითი	ცვლადები, ფუნქციები, შემთხვევითი რიცხვების გენერატორები
სტატისტიკური	რიგები, ცხრილები
დამამახსოვრებელი	უჯრედები, მატრიცის უჯრედები
დაჯგუფებული	რიცხვითი ჯგუფები, ტრანზაქტების ჯგუფები, სიები

აპარატურული კატეგორიის ობიექტები – ესაა აბსტრაქტული ელემენტი, რომელზეც შესაძლებელია რეალური სისტემის დეკომპოზიცია. ამ ობიექტებზე ზემოქმედებისას ტრანზაქტებმა შეიძლება შეცვალონ მათი მდგომარეობა და გავლენა მოახდინონ

სხვა ტრანზაქტების მოძრაობაზე. ასეთ ობიექტებს განეკუთ-ვნება ერთარხიანი მოწყობილობები, მრავალარხიანი მოწყობილობები (მეხსიერებები) და ლოგიკური გასაღებები.

ერთარხიანი მოწყობილობა წარმოადგენს მოწყობილობას, რომელიც დროის გარკვეულ მომენტში შეიძლება მხოლოდ ერთი ტრანზაქტით იყოს დაკავებული. მაგალითად: სატელეფონო კავშირის ერთი ხაზი, მონაცემთა გადაცემის ერთი არხი და ა.შ.

მრავალარხიანი მოწყობილობა განკუთვნილია იმ მოწყობილობების იმიტაციისათვის, რომლებიც უნდა შესრულდეს პარალელურად. მაგალითად: სატელეფონო კავშირის მრავალი ხაზი, მონაცემთა გადაცემის მრავალი არხი და ა.შ.

ოპერაციული ობიექტები არის ბლოკები, სადაც სისტემის მოდელის ფუნქციონირების ლოგიკის თანახმად ტრანზაქტების მოძრაობა განისაზღვრება აპარატურული კატეგორიის ობიექტებს შორის. ბლოკებში შეიძლება მოხდეს ოთხი ძირითადი ტიპის მოვლენა:

1. ტრანზაქტის შექმნა და გაქრობა;
2. ობიექტის რიცხვითი ატრიბუტების შეცვლა;
3. ტრანზაქტის დაყოვნება გარკვეული დროით;
4. მოდელში ტრანზაქტის მოძრაობის მარშრუტის შეცვლა.

გამოთვლითი კატეგორია მოდელირების პროცესში ემსახურება ისეთი პროცესების აღწერას, როდესაც კავშირი მოდელირების სისტემის კომპონენტებს შორის გამოისახება მათემატიკური (ანალიტიკური ან ლოგიკური) დამოკიდებულებით. ამ მიზნით გამოთვლითი კატეგორიის ობიექტების ხარისხში გამოიყენება არითმეტიკული ან ბულის ცვლადები და ფუნქციები.

ცვლადები წარმოადგენს რთულ გამოსახულებებს, რომლებიც აერთიანებს კონსტანტებს, სისტემურ რიცხვით ატრიბუტებს (სტრა), ბიბლიოთეკურ არითმეტიკულ ფუნქციებს, არითმეტიკულ და ლოგიკურ ოპერაციებს.

თითოეულ ობიექტს შეესაბამება ატრიბუტები, რომლებიც დროის მოცემულ მომენტში აღწერს მის მდგომარეობას. ისინი ხელმისაწვდომია და გამოიყენება მთელი მოდელირების პროცესის განმავლობაში და ეწოდებათ **სისტემური რიცხვითი ატრიბუტები.** მაგალითად: გამოთვლითი კატეგორიის ობიექტების შემთხვევაში – შემთხვევითი რიცხვების გენერატორს გააჩნია **RNn** სისტემური რიცხვითი ატრიბუტი – თანაბარალბათური განაწილებისას **n** ნომრის მქონე შემთხვევითი რიცხვების გენერატორით გამოთვლილი რიცხვი, ხოლო დინამიკური კატეგორიის ობიექტის შემთხვევაში – შემთხვევითი რიცხვის გენერატორს გააჩნია: **PR** – ტრანზაქტის მოცემულ მომენტში დამუშავების პრიორიტეტი; **Pi** – აქტიური ტრანზაქტის **i**-ური პარამეტრის მნიშვნელობა და სხვ. **GPSS World**-ში არსებობს 50-ზე მეტი სისტემური რიცხვითი ატრიბუტი.

ბულის ცვლადების დახმარებით მომხმარებელს შეუძლია ერთ ბლოკში ერთდროულად შეამოწმოს რამდენიმე პირობა ან ამ პირობის მნიშვნელობები და მისი ატრიბუტები. ბულის ცვლადები შეიძლება წარმოდგენილი იყოს სტანდარტული რიცხვითი ატრიბუტების კომბინაციებით, რომლებიც ერთმანეთთან დაკავშირებულია ბულის ოპერატორებით, აგრეთვე სხვა ცვლადებით. ბულის ცვლადები განისაზღვრება ისევე როგორც

არითმეტიკული, მაგრამ არითმეტიკულ ოპერაციებთან ერთად მოწმდება სხვადასხვა ლოგიკური ოპერაციები.

ფუნქციის დახმარებით მომხმარებელს უწყვეტი და დისკრეტული ფუნქციონალური დამოკიდებულების ფუნქციის არგუმენტს (დამოუკიდებელი სიდიდე) და ფუნქციის მნიშნელობას (დამოკიდებულ) შორის შეუძლია აწარმოოს გამოთვლები. ფუნქცია **GPSS**-ში გამოიყენება ცხრილის სახით ფუნქციის აღწერის ბრძანების დახმარებით. ისევე, როგორც ცვლადები ფუნქციაც არ არის დაკავშირებული განსაზღვრულ ბლოკთან.

GPSS World-ს გააჩნია 24 ჩაშენებული შემთხვევითი რიცხვების გენერატორი.

სტატისტიკურ ობიექტებს მიეკუთვნება რიგები და ცხრილები. ნებისმიერ სისტემაში ტრანზაქტების ნაკადის მოძრაობა შეიძლება შეყოვნდეს მოწყობილობის გაუმართავობის გამო. მაგალითად, ერთარხიანი მოწყობილობის არხი შეიძლება უკვე იყოს დაკავებული ან მრავალარხიანი მოწყობლობა იყოს უკვე გადავსებული. ასეთ შემთხვევაში შეყოვნებული ტრანზაქტები რიგში დგებიან, ანუ – კიდევ ერთი ტიპის **GPSS** ობიექტი. სწორედ ამ რიგის ანგარიში წარმოადგენს სისტემის დამპროექტებლის ერთ-ერთ ძირითად ფუნქციას. სისტემის დამპროექტებელი მოწყობილობისა და რიგის დამოკიდებულებით ავტომატურად განსაზღვრავს სტატისტიკას. ამის გარდა მომხმარებელს შეუძლია შეკრიბოს დამატებითი სტატისტიკური ინფორმაცია, სადაც ნაჩვენები იქნება მოდელში რიგის შექმნის სპეციალური წერტილები.

GPSS-ში სტატისტიკური ინფორმაციის ტაბულაციის გაადვილების მიზნით განხილულია სპეციალური ობიექტი – ცხრილი. ცხრილები გამოიყენება შემთხვე-ვითი რიცხვების ალბათურად განაწილებისათვის. ცხრილი შედგება სიხშირული კლასებისაგან (მნიშვნელობათა დიაპაზონი), სადაც შეიტანება თითოეული კონკრეტული რიცხვითი ატრიბუტის რიცხვი. ცხრილისათვის აგრეთვე გამოითვლება მათემატიკური ლოდინი ან საშუალო კვადრატული გადახრა.

დამამახსოვრებელი კატეგორიის ობიექტები უზრუნველყოფენ მიმართვას შენახულ მნიშვნელობებთან. უჯრედები (ან მატრიცის უჯრედები), სადაც შენახული სიდიდეები გამოიყენება რიცხვითი ინფორმაციის შესანახად. მაგალითად, მნიშვნელობა, რომელიც შეტანილია უჯრედში, შეიძლება თავის მხრივ, დროის გარკვეულ მომენტში წარმოადგენდეს მოწყობილობის გამოყენების კოეფიციენტს. ნებისმიერმა აქტიურმა ტრანზაქტმა აღნიშნულ ობიექტში შეიძლება განახორციელოს ინფორმაციის ჩაწერა. ხოლო ამ ობიექტიდან შეიძლება ინფორმაცია წაიკითხოს ახალმა ტრანზაქტმა. მატრიცების შემთხვევაში შეიძლება განხორციელდეს ექვსი სახის ცვლილება.

დაჯგუფებულ კატეგორიას მიეკუთვნება რიცხვითი ჯგუფები, ტრანზაქტების ჯგუფები და სიები.

მოდელებში ტრანზაქტები წარმოადგენს ობიექტებს, რომლებიც ხასიათდებიან საერთო სისტემური რიცხვითი ატრიბუტებით, რომლებიც ჩვეულებრივ წარმოადგენს ტრანზაქტების პარამეტრებს და იცვლება **ASSIGN** ბლოკში შესვლისას, რაც ატრიბუტებთან შეღწევის საშუ-

ალებას გვაძლევს. ხშირად შესაძლებელია ერთდროულად იცვლებოდეს მოცემული სიმრავლის ყველა ტრანზაქტის ატრიბუტები.

1.3. მოდელის სტრუქტურა GPSS ენაზე

GPSS ენაზე ბლოკ-დიაგრამის აგების შემდეგ იგი უნდა ჩაიწეროს ობიექტი „მოდელის“ შექმნისათვის მოსახერხებელ ფორმაში. ამისათვის კი აუცილებელია ინფორმაცია **GPSS** ობიექტის შესახებ ჩაწერილ იქნას ოპერატორების თანამიმდევრობით. ოპერატორი შედგება შემდეგი ველებისაგან:

სტრიქონის ნომერი (არააუცილებელი); ჭდე (არააუცილებელი); ზმნა (აუცილებელი); ოპერანდები (დამოკიდებულია ზმნაზე); კომენტარი (აუცილებელი).

GPSS-ის ობიექტების გარჩევისათვის (მაგალითად, ერთარხიანი ან მრავალარხიანი მოწყობილობა, რიგი და ა.შ.) მათ არქმევენ სახელებს. სახელი აუცილებელია დაიწყოს ასოთი და შესაძლებელია შეიცავდეს 200-მდე ასოსა და ციფრს, აგრეთვე ხაზგასმის ნიშანს. სახელი არ უნდა დაემთხვეს **GPSS-**ის მომსახურე სიტყვებს. ერთი და იგივე სახელი შესაძლოა გამოვიყენოთ სხვადასხვა ობიექტისათვის.

1.4. გამოთვლითი კატეგორიის ობიექტები

როგორც 1.1 ცხრილში არის აღნიშნული, გამოთვლითი კატეგორიის ობიექტებს მიეკუთვნება ფუნქციები, არითმეტიკული და ბულის ცვლადები. გამოსახულების ჩასაწერად კი გამოიყენება შემდეგი ელემენტები:

- კონსტანტები;
- სისტემური რიცხვითი ატრიბუტები;
- არითმეტიკული, პირობითი და ლოგიკური ოპერატორები;
- მათემატიკური (ბიბლიოთეკური) ფუნქციები;
- არითმეტიკული ცვლადები;
- ბულის ცვლადები;
- მომხმარებლის ცვლადები.

1.4.1. კონსტანტები

კონსტანტა – ეს არის მთელი ან ნამდვილი ტიპის რიცხვი, აგრეთვე სტრიქონული ტიპის მონაცემი.

მთელი ტიპის მონაცემები ინახება 32 თანრიგა რიცხვის სახით. იმ შემთხვევაში თუ მოხდება თანრიგის გადავსება მთელი რიცხვი გარდაიქმნება ნამდვილ რიცხვში. ნამდვილი ტიპის მონაცემები ინახება 64 თანრიგიან ორმაგი სიზუსტის მცოცავი წერტილიანი რიცხვითი სახით. მანტისა იცვლება -308-დან 308-მდე, ხოლო სიზუსტე შეზღუდულია 15 ათობითი თანრიგით. სტრიქონული მონაცემები ინახება **ASCII** სიმბოლოების მასივში

ნებისმიერი სიგრძით. მისი სიგრძე შეზღუდულია მხოლოდ **Max Memory Request** (მეხსიერების მაქსიმალური შეკითხვა) მნიშვნელობით, რომლის შეცვლა შესაძლებელია ობიექტის დაყენებით „მოდელირების პროცეს“-ში. ეს დაყენება შესაძლოა შეიცვალოს **Edit ►Settings** (კორექტირება►დაყენება) მენიუს ბრძანების საფუძველზე.

1.4.2. სისტემური რიცხვითი ატრიბუტები

მოდელირების პროცესში **GPSS World** ავტომატურად უკეთებს კორექტირებას იმ ინფორმაციას, რომელსაც წარმოადგენს მოდელში გამოყენებადი სხვადასხვა ობიექტების სისტემური რიცხვითი ატრიბუტები.

სისტემური რიცხვითი ატრიბუტების სახელი დარეზერვებულია. სტრა-ის სახელის განსაზღვრის წესი იყოფა სამ ჯგუფად:

პირველი ჯგუფის სტრა-ის სახელი შედგება ორი ნაწილისაგან. *პირველი ნაწილი* უჩვენებს ჯგუფურ სახელს რომელშიც იდენტიფიცირდება ობიექტის ტიპი და ინფორმაციის ტიპი ობიექტის შესახებ. *მეორე ნაწილი* იდენტიფიცირდება ჯგუფის კონკრეტულ წევრთან.

ჯგუფური სახელი შედგება ერთი ან ორი ასოსაგან რომელიც განსაზღვრავს ინფორმაციას გარკვეული ტიპის ობიექტის შესახებ. მაგალითად, **Q** – რიგის სიგრძის მიმდინარე მნიშვნელობაზე

მიმართვა, **FR** - მოწყობილობის გამოყენების კოეფიციენტი, **SR** - მეხსიერების გამოყენების კოეფიციენტი და ა.შ.

პირდაპირი მიმართვისას თუ ობიექტი იდენტიფიცირებულია ნომრის დახმარებით, მაშინ მიმართვა მის სტრა-ზე ჩაიწერება როგორც **სტრაj,** სადაც **j** – არის ობიექტის ნომერი (დადებითი მთელი რიცხვი). მაგალითად, **Q3 –** 3 ნომრის მქონე რიგის სიგრძის მიმდინარე მნიშვნელობა, **FR2 –** 2 ნომრის მქონე მოწყობილობის გამოყენების კოეფიციენტი. ობიექტის სახელის შემთხვევაში მიმართვა მის სტრა-ზე ჩაიწერება როგორც სტრა$ სახელი, სადაც სახელი – ობიექტის სახელია. მაგალითად, **Q$Basa – Basa** სახელის მქონე რიგის სიგრძის მიმდინარე მნიშვნელობა**. SR$Rem – Rem** სახელის მქონე მრავალარხიანი მოწყობილობის გამოყენების კოეფიციენტი.

ირიბი მიმართვისას სტრა განისაზღვრება როგორც **სტრა*პარამეტრი**. მისი შინაარსი მდგომარეობს შემდეგში:

- **სტრა*j,** სადაც **j** არის აქტიური ტრანზაქტის პარამეტრის ნომერი (დადებითი მთელი რიცხვი), რომელიც შეიცავს საჭირო ბლოკის ნომერს;
- **სტრა*სახელი,** სახელი – აქტიური ტრანზაქტის პარამეტრის სახელი, რომელიც შეიცავს საჭირო ბლოკის ნომერს.

მაგალითად, **Q*3** არის რიგის სიგრძის მიმდინარე მნიშვნელობა, რომლის ნომერს წარმოადგენს აქტიური ტრანზაქტის მე-3-ე პარამეტრის მნიშვნელობა; **SR*Rem** – მეხიერების გამოყენების კოეფიციენტი, რომლის ნომერს შეიცავს **Rem** სახელით მითითებული აქტიური ტრანზაქტის პარამეტრი.

მეორე ჯგუფს, როგორც პირველი ჯგუფის კერძო შემთხვევა, შეადგენს მატრიცული სტრა. **MX** მატრიცული სტრა შესაძლებელია შეიცავდეს სამამდე იდენტიფიკატორს (ირიბი მიმართვით). მაგალითად:

MX * Result (* Striqoni; * Sveti)

წარმოადგენს მატრიცაზე მიმართვას, რომლის ნომერს შეიცავს **Result** სახელით მითითებული აქტიური ტრანზაქტის პარამეტრი, ხოლო მატრიცის ელემენტი – სტრიქონის ნომერი და სვეტის ნომერი, რომლითაც განისაზღვრება **Striqoni** და **Sveti** სახელით მითითებული აქტიური ტრანზაქტის შესაბამისი პარამეტრების მნიშვნელობები.

მესამე ჯგუფს შეადგენს ისეთი სტრა, რომლებიც არ საჭიროებს ნომრის ან ობიექტის სახელის მითითებას.

სტანდარტული რიცხვითი ატრიბუტები მომხმარებელთან შეღწევადობის თვალსაზრისით შესაძლებელია დაიყოს ორ ჯგუფად:

პირველ ჯგუფს შეადგენს ატრიბუტები, რომლებიც მათი ფორმირების დროს მიღწევადია მხოლოდ **GPSS World**-ში, რომელთა გამოყენება შესაძლებელია გამოსახულებებში, რომლებიც არ იცვლებიან მოდელირების პროცესში. ეს ჯგუფი შეიცავს სტრა-ების მცირე რიცხვს.

მეორე ჯგუფში შედის გამოთვლითი ობიექტების სტრა-ები, რომლებიც შესაძლებელია შეიცვალოს მოდელში და მათი გამოყენება შესაძლებელია გამოსახულებებში:

- **FNj (FN$Raspr) - j** ნომრით მითითებული ფუნქციის გამოთვლილი ნამდვილი მნიშვნელობა (**Raspr** სახელით);
- **Vj (V$Ver) - j** ნომრით მითითებული ნამდვილი ან მთელი ტიპის ცვლადის გამოთვლილი მნიშვნელობა (**Ver** სახელით);
- **BVj (BV$Per) - j** ნომრით მითითებული ბულის ცვლადის გამოთვლილი ნამდვილი მნიშვნელობა (**Per** სახელით).

1.4.3. არითმეტიკული, პირობითი და ლოგიკური ოპერატორები

არითმეტიკული, პირობითი და ლოგიკური ოპერატორები განკუთვნილია ოპერაციის საჩვენელად, რომლებიც უნდა შესრულდეს გამოსახულების ელემენტებზე. **GPSS World**-ში გამოიყენება შემდეგი ოპერატორები (ნახ. 1.2):

თუ ოპერატორი მოითხოვს განსაზღვრული ტიპის მონაცემებს, მაშინ ეს მონაცემები ავტომატურად გარდაიქმნება შესაბამის ტიპის მონაცემში. მაგალითად, თუ სტრიქონული ტიპის სიდიდესთან მიმდინარეობს რიცხვითი ოპერაცია, მაშინ ამ სტრიქონის სიმბოლოებთან გამოიყენება რიცხვითი ექვივალენტი.

მთელად გაყოფის (\ ) ოპერაციის შედეგს წარმოადგენს გაყოფის შედეგად მიღებული მთელი რიცხვი, წილადი ნაწილი კი უგულებელყოფილია. ხოლო მოდულის მიხედვით (@) ოპერაციის შედეგს – გაყოფის შედეგად მიღებული ნაშთი. ქვემოთ მოყვანილია მაგალითები აღნიშნული ოპერაციების გამოყენებით.

53 \ 26 = 1	53 @ 26 = 1	8 \ 10 = 0	8 @ 10 = 8
7 \ 7 = 1	7 @ 7 = 0	17 \ 5 = 3	17 @ 5 = 2
54 \ 9 = 6	54 @ 9 = 0	3 \ 5 = 0	3 @ 5=3

ცხრილი 1.2

სიმბოლო	ოპერაცია	მნიშვნელობა
^	ახარისხება	A ^ B; A აყვანილია B ხარისხში
#	გამრავლება	A # B; A-ს B-ზე გამრავლება
/	გაყოფა	A / B; A-ს B-ზე გაყოფა
\	გაყოფის შედეგად მიღებული მთელი რიცხვი	A \ B; A-ს B-ზე გაყოფის შედეგად მოღებული მთელი რიცხვი
@	გაყოფის შედეგად მიღებული ნაშთი	A@B; A-ს B-ზე გაყოფის შედეგად მოღებული ნაშთი
+	შეკრება	–
_	გამოკლება	–
> ‘G’	მეტია	1, თუ A > B, წინააღმდეგ შემთხვევაში 0
>= ‘GE’	მეტია ან ტოლი	1, თუ A ≥ B, წინააღმდეგ შემთხვევაში 0
< ‘L’	ნაკლებია	1, თუ A < B, წინააღმდეგ შემთხვევაში 0
<= ‘LE’	ნაკლებია ან ტოლი	1, თუ A ≤ B, წინააღმდეგ შემთხვევაში 0
= ‘E’	ტოლი	1, თუ A = B, წინააღმდეგ შემთხვევაში 0
!= ’NE’	განსხვავებულია	1, თუ A != B, წინააღმდეგ შემთხვევაში 0
& ‘AND’	ლოგიკური „და“	1, თუ A და B ოპერანდიდან არც ერთი არ უდრის ნულს, წინააღმდეგ შემთხვევაში 0
| ‘OR’	ლოგიკური „ან“	1, თუ A და B ოპერანდიდან ერთ-ერთი არ უდრის ნულს, წინააღმდეგ შემთხვევაში 0

გამოსახულების გამოთვლისას უპირატესობა ენიჭება ( ), ხოლო შემდეგ ოპერაციების პრიორიტეტის გათვალისწინებით, გამოთვლა მიმდინარეობს მარცხნიდან მარჯვნივ: ^ , # / \ , @ , –, +, >= < =, < >, =, ! = , & , | .

1.4.4. მათემატიკური (ბიბლიოთეკური) ფუნქციები

GPSS World-ის ბიბლიოთეკას გააჩნია შემდეგი მათემატიკური ფუნქციები (პროცედურები):

ცხრილი 1.3

ფუნქცია	მ ნ ი შ ვ ნ ე ლ ო ბ ა
ABS(x)	აბსოლუტური მნიშვნელობა
ATN(x)	არკტანგენსი
COS(x)	კოსინუსი
EXP(x)	ექსპონენტა
INT(x)	მთელი ნაწილის გამოყოფა
LOG(x)	ნატურალური ლოგარითმი
SIN(x)	სინუსი
SQR(x)	კვადრატული ფესვი
TAN(x)	ტანგენსი

1.4.5. არითმეტიკული ცვლადები

არითმეტიკული ცვლადი წარმოადგენს სტრა-ს, რომელიც განისაზღვრება მომხმარებლის მიერ. **GPSS World**-ში არითმეტიკული ცვლადი განისაზღვრება ბრძანებით **VARIABLE** ან **FVARIABLE.**

ბრძანებას გააჩნია შემდეგი ფორმატი:

Name VARIABLE A
Name FVARIABLE A

სადაც **Name** – არის არითმეტიკული ცვლადის სახელი;

A ოპერანდი კი – გამოსახულება.

არითმეტიკულ ცვლადთან მიმართვა შესაძლებელია როგორც **V$** სახელი ან **Vj** სადაც **j** – არის ცვლადის ნომერი. სიმბოლური სახელი შესაძლებელია შეიცვალოს ნომრით – დადებითი მთელი რიცხვი, რისთვისაც შესაძლებელია **EQU** ბრძანების გამოყენება. მაგალითად:

Srskop EQU 2
Srskop VARIABLE X$12/MX$Vrem (3, P1)

ამ შემთხვევაში არითმეტიკულ ცვლადს **Srskop** შესაძლებელია მივმართოთ როგორც **V2.** ბრძანება **EQU** უნდა მოთავსდეს მოდელის დასაწყისში. ცვლადების აღწერის ბრძანებაში სახელის ნაცვლად ნომრის მითითება გამოიწვევს შეცდომას.

არითმეტიკული ცვლადის მნიშვნელობა გამოიყენება შემდეგი სახით:

- როგორც ოპერანდი; ამ შემთხვევაში არითმეტიკული ცვლადის მნიშვნელობას წარმოადგენს: **j** ობიექტის ნომერი; **j** ლოგიკური ატრიბუტი (**GATE** ბლოკი); ტრანზაქტის პარამეტრის ნომერი (**ASSIGN, INDEX, LOOP, MARK, SPLIT** ბლოკები); ატრიბუტის მნიშვნელობა.
- როგორც ფუნქციის მნიშვნელობა;
- როგორც ატრიბუტული ფუნქციის დამოკიდებული ცვლადის მნიშვნელობა;
- როგორც ცხრილის არგუმენტი;
- როგორც სხვა არითმეტიკული ცვლადის ან ბულის ცვლადის ოპერანდი.

ქვემოთ წარმოდგენილია არითმეტიკული გამოსახულების მაგალითები:

```
Pas    VARIABLE     R$Damg + S$Damg
Mak    VARIABLE     Q$Parik - Q$Box
Pak    VARIABLE     P5 + Q$Bul / 5
Vse    VARIABLE     N$Tev @ 10
```

პირველ მაგალითში **Pas** წარმოადგენს სიდიდეს, რომელიც დარჩენილი ტევადობისა და **Damg** მეხსიერების მიმდინარე შემცველობის ტოლია. მოცემული მაგალითიდან ჩანს, რომ არითმეტიკული ცვლადი შეიძლება მოცემულ იქნას როგორც

მეხსიერების ტევადობის სიდიდე. მეორე მაგალითში **Mak** ცვლადი განისაზღვრება როგორც **Parik** რიგის მიმდინარე შემცველობას მინუს **Box** რიგის მიმდინარე შემცველობა. მესამე მაგალითში **Pak** ცვლადი განისაზღვრება თავდაპირველად **Bul** რიგის სიგრძის მიმდინარე შემცველობის 5-ზე გაყოფით, ხოლო შემდეგ დაემატება **P5** აქტიური ტრანზაქტის პარამეტრის მნიშვნელობა. მეოთხე მაგალითში გამოითვლება **Vse – Tev** სახელით მოცემულ ბლოკში შესასვლელი რიცხვის მრიცხველი გაყოფილი 10-ზე მოდულით.

არითმეტიკული გამოსახულების მნიშვნელობა გამოითვლება მაშინ, როდესაც აქტიური ტრანზაქტი შევა ბლოკში, რომლის ოპერატორი არითმეტიკულ გამოსახულებაზე თავის ოპერანდებს შორის შეიცავს ერთ ან რამდენიმე მიმართვას. გამოთვლილი მნიშვნელობა იქნება ნამდვილი ტიპის. **GPSS World**-ში შუალედური შედეგების ან სტრა-ის მნიშვნელობების დამრგვალება არ არის შესაძლებელი. მაგალითად:

Kom1 VARIABLE MX$Gegma (P$Striqoni, P$Sveti) \ User

ამ შემთხვევაში **Kom1** სახელით მოცემული ფუნქციის მნიშვნელობა იქნება მთელი ტიპის. რადგან გამოსახულებაში ადგილი აქვს მთელად გაყოფის ოპერაციას. ნამდვილი რიცხვის დასამრგვალებლად შესაძლებელია გამოვიყენოთ ბიბლიოთეკური ფუნქცია **INT (x)** რომლის საშუალებით ხდება მთელი ნაწილის გამოყოფა.

მაგალითად:

Kom1 VARIABLE INT (MX$Gegma (P$Striqoni, P$Sveti) / User)

ამ შემთხვევაში გამოიყენება ჩვეულებრივი გაყოფის ოპერაცია. **VARIABLE** ბრძანებით ცვლადების აღწერისას ყველა შუალედური გამოთვლები და საბოლოო შედეგი დამრგვალდება მთელ რიცხვამდე. როგორც ცნობილია, ნამდვილი ტიპის ცვლადები (მცოცავი წერტილით) განისაზღვრება ბრძანებით **FVARIABLE.** ამ შემთხვევაში დამრგვალდება მხოლოდ საბოლოო შედეგი, ხოლო შუალედური მნიშვნელობები არ დამრგვალდება. მაგალითად, განსხვავების დანახვა ფიქსირებული და მცოცავი წერტილით გამოთვლილი ცვლადების შედეგებს შორის შესაძლებელია ქვემოთ მოყვანილ მაგალითებში:

Gan1 VARIABLE 30 # (4/3)
Gan2 FVARIABLE 30 # (4/3)

Gan1 ცვლადის მნიშვნელობა ტოლია 30, რადგან შუალედური ოპერაციის შედეგი 1, 33 დამრგვალდება 1-მდე. **Gan2** ცვლადის მნიშვნელობა კი ტოლია 39, რადგან კონსტანტა 30 გამრავლდება 1, 33-ზე და შედეგიდან (39, 9) აიღება მთელი ნაწილი.

შენიშვნა: **GPSS World**-ში შესაძლებელია ოპერანდის რანგში გამოსახულების გამოყენება. ამიტომ მოდელის გამარტივების მიზით რეკომენდებულია შემდეგი: თუ მოდელის სხვადასხვა ნაწილებში გამოიყენება ერთი და იგივე გამოსახულება, იგი ერთხელ უნდა განისაზღვროს ცვლადის სახით და მასთან მიმართვა შესაძლებელია მრავალჯერ. თუ მოდელის რომელ

ადგილზე უნდა იქნას განთავსებული გამოსახულება, ამას წყვეტს თავად მოდელის შემქმნელი. პროგრამის გამარტივების მიზნით რთულ მოდელებში, ყველა, როგორც არითმეტიკული ასევე ბულის ცვლადების გამოსახულება კომენტარის სწორად გაგების თვალსაზრისით შესაძლოა მოთავსდეს ერთ ადგილზე.

1.4.6. ბულის ცვლადები

რიცხვითი მონაცემების გარდა კომპიუტერი ოპერირებს აგრეთვე ლოგიკურ მონაცემებზე. ლოგიკურ ინფორმაციასთან მუშაობას საფუძვლად უდევს მათემატიკური ლოგიკის ერთ-ერთი მიმართულების – ბულის ალგებრის ცნებები. ინგლისელი მათემატიკოსი ჯორჯ ბული (1815-1864) კი – მათემატიკური ლოგიკის მიმართულების ფუძემდებელია.

ბულის ცვლადები მხოლოდ ერთი ბლოკის გამოყენებით საშუალებას იძლევა მიღებულ იქნას გადაწყვეტილებები, რომლებიც დამოკიდებულია **GPSS**–ის მრავალი ობიექტის ატრიბუტების მდგომარეობასა და მნიშვნელობაზე. ბულის ცვლადი წარმოადგენს სხვადასხვა სტანდარტულ რიცხვით ატრიბუტებს (სტრა) და მათ შორის სხვა ბულის ცვლადებისაგან შემდგარ ლოგიკურ გამოსახულებებს. გამოსახულება ასევე შესაძლებელია შეიცავდეს ბიბლიოთეკურ ფუნქციებს. ბულის ცვლადი განისაზღვრება **BVARIABLE** ბრძანებით. ბრძანების ფორმატს გააჩნია შემდეგი სახე:

Name BVARIABLE A

Name – არის ბულის ცვლადის სახელი, რომელიც ისევე როგორც არითმეტიკული ცვლადის სახელი **EQU** ბრძანებით მომხმარებლის სურვილისამებრ შესაძლოა შეცვლილ იქნას ნომრით.

A ოპერანდი – ლოგიკური გამოსახულება.

გამოსახულებაში შესაძლებელია გამოყენებულ იქნას არითმეტიკული და ლოგიკური ოპერატორები (იხ. ცხრილი 1.2). საბოლოო შედეგი გარდაიქმნება მთელ მნიშვნელობაში 0, თუკი უდრის ნულს, ან მთელ მნიშვნელობაში 1, თუ განსხვავდება ნულისაგან.

ლოგიკური ოპერატორები დაკავშირებულია აპარატურული კატეგორიის ობიექტებთან და გამოიყენება აღნიშნული ობიექტების მდგომარეობის განსაზღვრისათვის.
არსებობს შემდეგი ლოგიკური ოპერატორები:

- **FVj** = 1, თუ **j** მოწყობილობა ხელმისაწვდომია, წინააღმდეგ შემთხვევაში – 0;
- **FIj** = 1, თუ **j** მოწყობილობა ემსახურება წყვეტას, წინააღმდეგ შემთხვევაში – 0;
- **SFj** = 1, თუ მრავალარხიანი მოწყობილობა **j** მთლიანად შევსებულია, წინა-აღმდეგ შემთხვევაში – 0;
- **SEj** = 1, თუ მრავალარხიანი მოწყობილობა **j** ცარიელია, წინააღმდეგ შემთხვევაში – 0;
- **SVj** = 1, თუ მრავალარხიანი მოწყობილობა **j** მისაწვდომია, წინააღმდეგ შემთხვევაში – 0;

- **LSj** = 1, თუ **j** ლოგიკური გასაღები (გადამრთველი) ჩართულია, წინააღმდეგ შემთხვევაში – 0.

j -ს ქვეშ იგულისხმება ნომერი ან სახელი. მაგალითად:

```
Mam    BVARIABLE    FV$Rem1
Can3   BVARIABLE    SF$Kvandz
Prov1  BVARIABLE    SV$Gashv
Prov2  BVARIABLE    LS2
```

პირველ მაგალითში **Mam** ბულის ცვლადი უდრის 1-ს, თუ **Rem1** მოწყობილობა მისაწვდომია და 0-ს, თუ იგი არ არის მისაწვდომი. მეორე მაგალითში **Can3** ბულის ცვლადი უდრის 1-ს, თუ **Kvandz** მრავალარხიანი მოწყობილობა ბოლომდე შევსებულია. მესამე მაგალითში **Prov1** ბულის ცვლადი უდრის 1-ს, თუ **Gashv** მრავალარხიანი მოწყობილობა მისაწვდომია. მეოთხე მაგალითში **Prov2** ბულის ცვლადი უდრის 1-ს, თუ ნომერი 2 ლოგიკური გასაღები ჩართულია.

დამოკიდებულების ოპერატორები (იხ. ცხრილი 1.2.) აწარმოებენ ოპერანდების ალგებრულ შედარებას. მაგალითად:

```
Prov1    BVARIABLE    V$VseAvt 'G' 16
Prov2    BVARIABLE    Q5$ 'LE' P3
Prov3    BVARIABLE    MX$Gegma ( Striqoni, Sveti ) 'GE' P2
```

Prov1 ბულის ცვლადი უდრის 1-ს, თუ **VseAvt** ცვლადი მეტია 16-ზე, წინააღმდეგ შემთხვევაში უდრის 0-ს. მეორე მაგალითში

Prov2 ბულის ცვლადი უდრის 1-ს, თუ ნომერი 5 რიგის მიმდინარე სიგრძე ნაკლებია ან ტოლი ტრანზაქტის პარამეტრი 3-ის მნიშვნელობაზე. მესამე მაგალითში **Prov3** ბულის ცვლადი უდრის 1-ს, თუ **Gegma** მატრიცის განსაზღვრული ელემენტის მნიშვნელობა მეტია ან ტოლი პარამეტრი 2-ის მნიშვნელობაზე.

ბულის ოპერატორები ორი სახისაა: **OR – ან** და **AND – და. ან** ოპერატორი ამოწმებს თუნდაც ერთი პირობის შესრულებას, ხოლო **და** ოპერატორი მოითხოვს ორივე პირობის შესრულებას. მაგალითად:

```
Con1    BVARIABLE   FI$Rem ’OR’ SF4
Con2    BVARIABLE   FI$Rem ’AND’ SF$Porg
Con3    BVARIABLE( V3 ’G’ 7 ) ’AND’ ( FN$Rav ’OR’ LS7)
```

Con1 ბულის ცვლადი უდრის 1-ს, თუკი სრულდება ერ-ერთი პირობა: **Rem** მნიშვნელობა ემსახურება წყვეტას, ან ნომერი 4 მეხსიერება არ არის შევსებული. **Con2** ბულის ცვლადი უდრის 1-ს, თუკი სრულდება ორივე პირობა: **Rem** მოწყობილობა ემსახურება წყვეტას და მეხსიერება **Porg** სახელით არ არის შევსებული. მესამე მაგალითში **Con3** ცვლადი უდრის 1-ს, თუკი სრულდება ორივე პირობა: ნომერი 3 ცვლადის მნიშვნელობა მეტია 7-ზე და ნომერი 7 ლოგიკური გასაღები ჩართულია. ფრჩხილები მესამე მაგალითში საჭიროა მხოლოდ განსაზღვრულ ბულის თანაფარდობათა დასაწესებლად. ფრჩხილები გამოყენებული უნდა იქნეს მხოლოდ იმ შემთხვევაში, როდესაც ისინი აუცილებელია.

თუ ბულის ცვლადი მოიცემა სტრა-ს სახით, როგორც მაგალითად,

Stan BVARIABLE V$TreAvt

ამ შემთხვევაში გამოითვლება **TreAvt** სახელის მქონე არითმეტიკული ცვლადის მნიშვნელობა, და თუ იგი განსხვავდება ნულისაგან, მაშინ **Stan** ბულის ცვლადის მნიშვნელობა 1-ის ტოლი იქნება, წინააღმდეგ შემთხვევაში – 0.

წინამდებარე თავში განხილულია გამოსახულების ყველა ელემენტი, გარდა ფუნქციებისა. ფუნქციათა გამოყენების მეთოდები განხილული იქნება მომდევნო თავებში.

1.4.7. მომხმარებლის ცვლადები

GPSS მომხმარებელს საშუალებას აძლევს როგორც რიცხვითი, ასევე სტრიქონული ტიპის მონაცემების აუცილებელი მოდელირების დროს მათ შესანახად გააჩნდეს თავისი ცვლადები. მომხმარებლის ცვლადები იქმნება **EQU** ბრძანების მეშვეობით. მაგალითად:

Ver1 EQU 2. 67
Striqoni EQU 11
Sveti EQU 9
Name EQU „ Name“

პირველ სამ მაგალითში **Ver1, Striqoni, Sveti** მომხმარებლის ცვლადებს მიენიჭებათ რიცხვითი მნიშვნელობები, ხოლო **Name** ცვლადს – სტრიქონული.

მომხმარებლის ცვლადის დახმარებით **GPSS** ობიექტების სახელი შესაძლებელია შეიცვალოს ნომრებით. ამისათვის **EQU** ბრძანებით წინასწარ უნდა განისაზღვროს მომხმარებლის ცვლადი.

მოდელში გამოყენებული ობიექტების ჭდეები შესაძლებელია მოცემული იქნას ასევე **EQU** ბრძანებით.

1.5. GPSS მოდელების დინამიკური ელემენტები. ტრანზაქტები

GPSS მოდელის ბლოკ-სქემის კონფიგურაცია ასახავს მიმართულებებს რომლებზედაც წარმოებს გადანაცვლებადი ელემენტების მოძრაობა. თითოეულ ამგვარ ელემენტს ეწოდება ტრანზაქტი. ტრანზაქტები წარმოადგენს **GPSS**-მოდელის დინამიკურ (ანუ მოძრავ) ელემენტებს. **GPSS** მოდელის მუშაობა მდგომარეობს ბლოკიდან ბლოკებისკენ (ბლოკებში) ტრანზაქტების გადანაცვლებაში.

მოდელირების დასაწყისში **GPSS** მოდელში არ არის არც ერთი ტრანზაქტი. მოდელირების პროცესში დროის განსაზღვრულ მომენტებში ტრანზაქტები შედის მოდელში იმ ლოგიკურ მოთხოვნილებათა შესაბამისად რომლებიც წარმოიშობა სამოდე-

ლირებელ სისტემაში. ამგვარად, დროის გარკვეულ მომენტებში, მოდელირების სპეციფიკიდან გამომდინარე ტრანზაქტები ტოვებს მოდელს. საერთო შემთხვევაში, მოდელში არსებობს ტრანზაქტების დიდი რიცხვი, თუმცა დროის ერთ მომენტში მოძრაობს მხოლოდ ერთი ტრანზაქტი. თუკი ტრანზაქტმა დაიწყო თავისი მოძრაობა, იგი ბლოკიდან ბლოკში გადაადგილდება ბლოკ-სქემის მიერ დადგენილი გზის მიხედვით. თითოეული ბლოკი შეიძლება განხილულ იქნეს როგორც რომელიმე წერტილი რომელშიც წარმოებს ქვეპროგრამებისადმი მიმართვა. იმ მომენტში, როდესაც ტრანზაქტი შედის ბლოკში, შესრულებაზე გამოიძახება შესაბამისი ქვეპროგრამა და ამის შემდეგ ტრანზაქტი (საერთო შემთხვევაში) ცდილობს შევიდეს მომდევნო ბლოკში. ტრანზაქტის ამგვარი წინ წაწევა (ანუ გადაადგილება) გრძელდება იქამდე, ვიდრე არ მოხდება ერთ-ერთი მომდევნო შესაძლო მოვლენათაგანი:

1. ტრანზაქტი შედის ბლოკში, რომლის ფუნქციას წარმოადგენს ტრანზაქტის დაყოვნება მოდელით განსაზღვრული რაიმე გარკვეული დროით;
2. ტრანზაქტი შედის რომელიმე ბლოკში, რომლის ფუნქციად გვევლინება ტრანზაქტის მოდელიდან გამოდევნა;
3. ბლოკ-სქემის მიერ დადგენილი ლოგიკის თანახმად ტრანზაქტი „ცდილობს“ შევიდეს მომდევნო ბლოკში, თუმცაღა ბლოკი „უარს აცხადებს“ მიიღოს აღნიშნული ტრანზაქტი. ამ შემთხვევაში ტრანზაქტი რჩება იმ ბლოკში რომელშიც იგი იმ პერიოდისათვის იმყოფებოდა. ოდნავ მოგვიანებით იგი გაიმეორებს მცდელობას შევიდეს

მომდევნო ბლოკში. როდესაც მოდელში არსებული პირობები შეიცვლება, ერთ-ერთი ამდაგვარი მცდელობა შესაძლოა წარმატებული აღმოჩნდეს. ამის შემდგომ ტრანზაქტი შეძლებს მოდელში თავისი გადაადგილების გაგრძელებას.

ტრანზაქტებსა და რეალური სისტემების ელემენტებს შორის შესაძლო ანალოგიათა ზოგიერთი მაგალითი მოყვანილია 1.5 ცხრილში. საპარიკმახეროს მოდელში, მაგალითად, ტრანზაქტებს შეუძლიათ წარმოგვიდგნენ როგორც კლიენტების სიმბოლო. რეალურ სისტემაში კლიენტი მიდის საპარიკმახერო სალონში, უერთდება თავისი რიგის მომლოდინე კლიენტებს, ელის თავის რიგს, მას ემსახურება პარიკმახერი, დაბოლოს იგი ტოვებს საპარიკმახერო სალონს.

ცხრილი 1.5

ტრანზაქტებსა და სისტემების ელემენტებს შორის შესაძლო შესაბამისობის მაგალითები

ს ი ს ტ ე მ ე ბ ი	ტრანზაქტების მიერ სიმბოლოების სახით წარმოდგენილი სისტემების ელემენტები
სუპერმარკეტი	მყიდველი
საავტომობილო გზა	ავტომობილი
რადიოსარემონტო	რადიომიმღებები
საწყობი	განაცხადი
საპარიკმახერო	კლიენტი

საპარიკმახეროს მოდელში **GPSS**-ზე შესაძლოა გამოყენებულ იქნას სხვადასხვა ბლოკები მომსახურების ისეთი ელემენტების აღსანიშნავად, როგორიც გახლავთ კლიენტის მოსვლა (ტრანზაქტის შესვლა), პარიკმახერის მიერ კლიენტის მომსახურება, კლიენტის მიერ საპარიკმახერო სალონის დატოვება (ტრანზაქტის გასვლა) და ა.შ.

ბლოკიდან ბლოკში ტრანზაქტის გადაადგილება მოდელში, საპარიკმახეროში ამგვარად ერთი სტადიიდან მეორეში კლიენტის გადანაცვლების ანალოგიური გახლავთ.

1.6. სამოდელო დროის წამმზომი

რეალურ სისტემათა სხვადასხვა მოვლენები ხდება დროის რაღაც პერიოდის განმავლობაში. კლიენტი(ები) მიდის საპარიკმახერო სალონში; როდესაც ახლოვდება მათი რიგი, ისინი ხვდებიან პარიკმახერთან მომსახურებაზე. თმის შეჭრის პროცესი სრულდება და კლიენტი ტოვებს საპარიკმახერო სალონს.

თუკი ყველა ამ მოვლენას წარმოვიდგენთ მოდელში, ამ შემთხვევაში მათი წარმოშობა უნდა წარმოებდეს სამოდელო დროის ფონზე. მაშასადამე, **GPSS**-ის ინტერპრეტატორი ავტომატურად უნდა ემსახურებოდეს სამოდელო დროის წამმზომს.

როდესაც იწყება მოდელირება, ინტერპრეტატორში იგეგმება პირველი ტრანზაქტის შესვლა. ამის შემდეგ, სამოდელო დროის წამმზომი დგება მოდელში პირველი ტრანზაქტის გამოჩენის

მომენტის შესაბამისი დროის მნიშვნელობაში. აღნიშნული ტრანზაქტი (აგრეთვე სხვა ტრანზაქტებიც თუკი ისინი შემოდის დროის იმავე მომენტში) შედის მოდელში. ამის შემდგომ იგი წინ მიიწევს მოდელის ყველა შესაძლო ბლოკების გავლით რომლებიც რა თქმა უნდა ხვდება მას. სავსებით ბუნებრივია, რომ პირველად წამმზომით აღნიშნული დროის მომენტისათვის სისტემაში მეტი არაფერი აღარ ხდება. **GPSS**-ის ინტერპრეტატორი წინ სწევს წამმზომის მნიშვნელობას დროის იმ მნიშვნელობამდე, როდესაც ხდება მის მიერ დაგეგმილი მომდევნო მოვლენა (ან მოვლენები). ეს მოვლენები, რომლებიც როგორც წესი წარმოიშობა ბლოკების გავლით ტრანზაქტების გადაადგილების შედეგად, წარმოიქმნება დროის შემდგომ მომენტებში. იმ შემთხვევაში, როდესაც მეორედ წამმზომით აღნიშნული დროის მომენტში აღარ რჩება ტრანზაქტი, რომელიც უნდა გადაადგილდეს, წამმზომი კვლავ წინ მიიწევს და ა. შ. ზუსტად ამგვარად ხორციელდება სამოდელო დროის ცვლილება. აღნიშ-ნული მოვლენა ავხსნათ მაგალითის მეშვეობით, რომელიც რიცხობრივად გარკვეულ წარმოდგენას შეგვიქმნის მოდელში დროის ცვალებადობის შესახებ. დავუშვათ, რომ საპარიკმახერო სალონში, სადაც მხოლოდ ერთი პარიკმახერი მუშაობს, დროის პირველი რამდენიმე მოვლენა მოხდა ისე, როგორც ეს წარმოდგენილია 1.6 ცხრილის სახით. **GPSS**-ზე მოდელირების დროს ინტერპრეტატორი წამმზომთან ერთად იმუშავებს შემდეგნაირად. მოდელირების დასაწყისში იღება საპარიკმახერო სალონი. არაფერი ხდება დროის მომენტის დადგომამდე, რომელიც 22-ის ტოლია. წამმზომი ჩვენს მაგალითში დგება მნიშვნელობაში

22, შემოდის პირველი კლიენტი (ტრანზაქტი შედის მოდელში). იგი ხედავს, რომ მას არ მოუწევს ლოდინი და მიემართება სავარძლისაკენ (ტრანზაქტი ბლოკიდან ბლოკის გავლით მიიწევს წინ იქამდე, ვიდრე არ ხვდება იმ ბლოკში, რომელიც აყოვნებს მას, რაც თავის მხრივ შეესაბამება თმის შეჭრის დროს). ამის შემდეგ არ ხდება არავითარი მოვლენა იმ დროის მნიშვნელობის დადგომამდე, რომელიც 29-ის ტოლია. წამმზომი დგება მნიშვნელობაში 29, შემოდის მეორე კლიენტი (მომდევნო ტრანზაქტი შედის მოდელში). აღნიშნული კლიენტი იძულებულია მოიცადოს ვიდრე პარიკმახერი არ განთავისუფლდება (ტრანზაქტი არ იღებს თანხმობას იმ ბლოკში შესვლაზე, რომელიც მოდელირების დროს პარიკმახერის დაკავებულობის მასიმბოლირებელ მოვლენად გვევლინება). სამოდელო დროის მოცემულ მომენტში უკვე მეტი აღარაფერი ხდება. ინტერპრეტატორი წამმზომის მნიშვნელობას აყენებს 33-ში, როდესაც შემოდის მესამე კლიენტი (კიდევ ერთი ტრანზაქტი შედის მოდელში) და ა. შ.

ახლა ნათელია, თუ როგორ ხდება მოვლენათა თანამიმდევრობის თვალყურის დევნება მოდელში მიმდინარე სამოდელო დროის წამმზომის მეშვეობით. **GPSS** სისტემის ერთ-ერთ უპირატესობას წარმოადგენს ის, რომ სამოდელო დროის წამმზომი ავტომატურად კორექტირდება მოდელით მითითებული ლოგიკის შესაბამისად. ამგვარად, მოდელის შემქმნელისათვის არ წარმოადგენს აუცილებლობას თავად ადევნოს თვალყური დროის ზუსტ ცვალებადობას, როგორც ამას აქვს ადგილი რეალურ სისტემებში.

ცხრილი 1.6

მოვლენათა თანამიმდევრობის მაგალითი ერთი სავარძლის მქონე საპარიკმახერო სალონში

მოვლენათა წარმოშობის თანამიმდევრობა	მოვლენა	მოვლენის წარმოშობის ფაქტიური დრო	მოვლენის წარმოშობის სამოდელო დრო, წთ
1	პარიკმახერი აღებს სალონს	8 სთ 00 წთ	0
2	პირველი კლიენტი შემოდის და მომსახურების მიზნით მიემართება სავარძლისაკენ	8 სთ 22 წთ	22
3	შემოდის მეორე კლიენტი	8 სთ 29 წთ	29
4	შემოდის მესამე კლიენტი	8 სთ 33 წთ	33
5	პირველი კლიენტი მომსახურებულია	8. სთ 47 წთ	47
6	მეორე კლიენტი მიემართება მოსამსახურებლად	8. სთ 47 წთ	47
7	შემოდის მეოთხე კლიენტი	9 სთ 07 წთ	67

განვიხილოთ **GPSS**-ის და მისი წამმზომის ზოგიერთი მნიშვნელოვანი თავისებურებანი.

1. **GPSS**-ის წამმზომი არეგისტრირებს მთელ მნიშვნელობებს. ეს ნიშნავს, რომ ვითარება შესაძლოა წარმოიშვას მხოლოდ დროის „მთელ“ მომენტებში.

2. დროის ერთეულს, რომელიც შესაძლოა აღნიშნულ იქნას წამმზომის მეშვეობით, განსაზღვრავს მოდელის შემქმნელი, თუმცა დროის ერთეულს არასოდეს არ ატყობინებენ ინტერპრეტატორს. აღნიშნულ მნიშვნელობას მოდელში შესაყვანი დროების მონაცემთა ფორმით არაცხადად გამოხატავენ. თუკი ყველა ასეთი მონაცემი გამოხატულია წუთებში, ამ შემთხვევაში დროის საგულისხმო ერთეულად წარმოგვიდგება წუთი. თუ ყველა მონაცემი გამოსახულია მილიწამებში, მაშინ დროის იმგვარი ერთეული იქნება მილიწამი. მოდელის შემქმნელს შეუძლია დააწესოს დროის ისეთი ერთეული, რომელიც მისთვის ხელსაყრელია, იმ თვალსაზრისით, რომ მოდელში ზუსტად ასახოს რეალური სისტემის მოვლენები. ამასთან ერთად, იგი ვალდებულია თვალყური ადევნოს იმას, რომ დროსთან დაკავშირებული ყველა მონაცემი გამოსახულ იქნას დროის განსაზღვრულ მინიმალურ ერთეულში. 1.6 ცხრილში წარმოდგენილ საპარიკმახერო სალონთან დაკავშირებულ მაგალითში დროის ერთეულად გვევლინება 1 წთ. ეს ნიშნავს, რომ არ შეიძლება, მაგალითად, რომ კლიენტი მივიდეს სალონის გახსნის შემდეგ 40 წმ-ის შემდეგ. თუკი აუცილებელია მოდელში მოვლენის წარმოშობის დაწესება 1 წმ-მდე სიზუსტით, ამ შემთხვევაში დროის ერთეული უნდა იქნას ინტერვალი, რომელიც არ აჭარბებს 1 წმ-ს.
3. **GPSS** წარმოადგენს ინტერპრეტატორს ე.წ. „მომდევნო მოვლენისთვის.“ სხვაგვარად რომ ვთქვათ, იმის შემდეგ, რაც სამოდელო დროის მოცემულ მომენტში მოდელი

ბოლომდე არის კორექტირებული, წამმზომი წინ მიიწევს დროის უახლოეს მნიშვნელობამდე, რომელშიც ხდება შემდეგი მოვლენა. სამოდელო დროის ინტერვალი გატარდება თუ ამ ინტერვალზე არ ვითარდება მოვლენა. პრაქტიკულად ეს აღნიშნავს იმას, რომ მოდელის გაშვების დრო არ არის დამოკიდებული მოდელის შემქმნელის მიერ შერჩეული დროის ერთეულისაგან. დაბოლოს, განვსაზღვროთ რაში მდგომარეობს განსხვავება სამოდელო და რეალურ დროებს შორის. როდესაც სამოდელო დროის წამმზომის მნიშვნელობა წინ მიიწევს მომდევნო წერტილისაკენ, ხდება რაღაც შეფერხება იმისათვის, რომ მოხდეს მოდელის კორექტირება. რეალურ სისტემაში კი კორექტირებაზე ამგვარი დრო არ არსებობს. არ არის გამორიცხული, რომ სამოდელო დროის მხოლოდ ერთი წუთის მოდელირებისათვის საჭირო იქნება რეალური დროის რამდენიმე საათი. მეორეს მხრივ, ექსპერიმენტებს, რომლებსაც ახორციელებენ სამოდელო დროის რამდენიმე კვირის, თვეების ან წლების განმავლობაში, კომპიუტერზე მოდელირების დროს შესაძლოა დაიკავოს რეალური დროის მხოლოდ რამდენიმე წამი. მოდელების ამგვარი თვისება, რომელიც საშუალებას იძლევა გარკვეულწილად შეამციროს დრო, სისტემებთან ექსპერიმენტის თვალსაზრისით მათი კომპიუტერზე მოდელირების მეშვეობით წარმოადგენს ერთ-ერთ პოტენციურ უპირატესობას.

II თავი
მოდელების აგება
ერთარხიანი მოწყობილობებისათვის

ნებისმიერი სისტემის მომსახურებისათვის გამოიყენება ელემენტები, ასეთი ელემენტები შეიძლება იყოს ადამიანი ან საგანი. **GPSS** ენაში მსგავს ელემენტებს უწოდებენ აპარატურული კატეგორიის ობიექტებს, რომელთაც მიკუთვნება ერთარხიანი და მრავალარხიანი მოწყობილობები. აგრეთვე, ლოგიკური გასაღები (გადამრთველი).

ერთარხიანი მოწყობილობა ხასიათდება ორი ძირითადი თვისებით:

- დროის ნებისმიერ მომენტში ყველა ერთარხიანი მოწყობილობა უნდა მოემსახუროს მხოლოდ ერთ ტრანზაქტს; თუ მომსახურების პროცესში გამოჩნდება ახალი ტრანზაქტი, ის უნდა დაელოდოს თავისი მომსახურების დროს ან გადაინაცვლოს სხვა ადგილზე, ან თუ მას გააჩნია შედარებით მაღალი პრიორიტეტი დასრულებამდე შეწყვიტოს მისი მომსახურება;
- როდესაც ტრანზაქტი შედის ერთარხიან მოწყობილობაში მოდელში აუცილებლად უნდა გამოიყოს დრო მისი მომსახურებისათვის.

მოდელში შეიძლება იყოს რამდენიმე ერთარხიანი მოწყობილობა, ამიტომ თითოეულ მათგანს გააჩნია სახელი.

ერთარხიანი მოწყობილობის ფუნქციონირების ორგანიზებისათვის მოდელირებისას შესაძლებელია შემდეგი რეჟიმების არსებობა:

- ერთარხიანი მოწყობილობის დაკავება და მისი განთავისუფლება;
- ერთარხიანი მოწყობილობის მომსახურების შეწყვეტა;
- ერთარხიან მოწყობილობის მიუწვდომლობა და მისაწვდომლობის აღდგენა.

2.1. ტრანზაქტების მოდელში შესვლა. CENERATE ბლოკი

ბლოკი **CENERATE**-ის საშუალებით ხორციელდება ტრანზაქტების მოდელში შესვლა. მას გააჩნია შემდეგი ზოგადი სტრუქტურა:

CENERATE [A], [B], [C], [D], [E]

[ ] ფრჩხილები აღნიშნავს, რომ არ არის აუცილებელი ყველა ოპერანდის გამოყენება. ერთ მოდელში შესაძლებელია გამოვიყენოთ რამდენიმე **CENERATE** ბლოკი. **CENERATE** ბლოკში

ტრანზაქტების თანამიმდევრობით გამოჩენებს შორის ინტერვალს დროის ინტერვალი ეწოდება. დროის ინტერვალები განისაზღვრება **A** და **B** ოპერანდებით. აღნიშნული ინტერვალები განაწილებულია თანაბარ-ალბათურად.

A ოპერანდი არის მოდელში ტრანზაქტების თანამიმდევრობით გამოჩენის დროის საშუალო ინტერვალი;

B ოპერანდი – განისაზღვრება **A** ოპერანდზე დამოკიდებულებით.

არსებობს ორი ტიპის მოდიფიკატორი: მოდიფიკატორი-ინტერვალი და მოდიფიკატორი-ფუნქცია.

მოდიფიკატორი-ინტერვალის დახმარებით ტრანზაქტის გენერაციის დროებს შორის არსებობს თანაბარალბათური განაწილება. ამ შემთხვევაში **A** ოპერანდი არის სამოდელო დროის ქვედა ზღვარი, ხოლო **B** ოპერანდი კი – ზედა. ტრანზაქტების გენერაციის შემთხვევაში რიცხვები აიღება ამ ინტერვალიდან. მაგალითად, 7±4 შემთხვევეაში ქვედა ზღვარი ტოლია 3-ის და ზედა – 11, რაც იმას ნიშნავს, რომ ტრანზაქტების შემოსვლა მიმდინარეობს თანაბარალბათურად და დროის ინტერვალები ასე გამოიყურება: 3, 4, 5, 6, 7, 8, 9, 10, 11.

თანაბარალბათური განაწილებისას შესაძლებელია შემთხვევითი რიცხვის არჩევა. ამის გაკეთება შეიძლება **Edit►Setting (**რედაქტირება►დაყენება**)** ბრძანების საშუალებით. **Random Numbers** (შემთხვევითი რიცხვი) გვერდზე **GENERATE** ველში გენერირებს ნომრის შეტანით – ნებისმიერი დადებითი რიცხვი.ფარულად გამოიყენება გენერატორი 1 ნომრით.

მოდიფიკატორი-ფუნქციის შემთხვევაში ტრანზაქტების განაწილება მიმდინარეობს არათანაბარალბათურად. ამ შემთხვევაში **A** ოპერანდის მნიშვნელობა გამრავლდება **B** ოპერანდით მოცემული ფუნქციის მნიშვნელობაზე.

ნებისმიერი ხერხით გამოთვლილი დროითი ინტერვალების შემთხვევაში **B** ოპერანდის მნიშვნელობა არ უნდა აღემატებოდეს **A** ოპერანდის მნიშვნელობას, წინააღმდეგ შემთხვევაში შესაძლებელია **GENERATE** ბლოკმა მიიღოს უარყოფითი დროის ინტერვალი, რომელიც გამოიწვევს შემდეგი სახის შეცდომას "დაყოვნების უარყოფითი რიცხვი".

C ოპერანდი – ეს არის დროის მომენტი, როდესაც პირველი ტრანზაქტი უნდა შემოვიდეს მოდელში. ამის შემდეგ ყველა დანარჩენი ტრანზაქტები შემოდიან თანაბარალბათურად **A** და **B** ოპერანდებით განსაზღვრული დროის ინტერვალის განმავლობაში. **C** ოპერანდი გამოიყენება იმ ემთხვევაში, როდესაც საჭიროა პირველი ტრანზაქტის დაჩქარება ან შეყოვნება; აგრეთვე, იმ დროს როდესაც პირველი ტრანზაქტი უნდა შევიდეს მოდელში დროის გარკვეულ მომენტში.

D ოპერანდი – განსაზღვრავს მოდელირების დროის განმავლობში **CENERATE** ბლოკში შესასვლელი ტრანზაქტების რაოდენობას.

E ოპერანდი – განსაზღვრავს მოდელირების დროის განმავლობში **CENERATE** ბლოკში შესასვლელი ტრანზაქტების პრიორიტეტს. ამ ბლოკის არარსებობის შემთხვევაში ტრანზაქტების პრიორიტეტი ნულის ტოლია.

2.2. მოდელირების პროცესის დასრულებისას მოდელიდან ტრანზაქტების წაშლა. TERMINATE ბლოკი

TERMINATE ბლოკის საშუალებით ხორციელდება მოდელიდან ტრანზაქტების წაშლა. მას გააჩნია შემდეგი ზოგადი სტრუქტურა:

TERMINATE [A]

A ოპერანდის მნიშვნელობას წარმოადგენს რიცხვი 1, რომლითაც მოდელირების პროცესის დასრულებისას მცირდება **TERMINATE** ბლოკში შესული ტრანზაქტების რაოდენობა. მოდელირების პროცესის დაწყებისას აღნიშნული რიცხვი წინასწარ ჩაწერილია **START** ბრძანებაში და შენახულია მეხსიერებაში.

2.3. ერთარხიანი მოწყობილობის დაკავება და განთავისუფლება. SEIZE და RELEASE ბლოკები

როდესაც ტრანზაქტი შედის მოდელში აღნიშნული ტრანზაქტის მიერ წარმოებს ერთარხიანი მოწყობილობის დაკავება.

თუ ერთარხიანი მოწყობილობა დაკავებულია, მაშინ ტრანზაქტი მოდელში ვერ შედის და დგას რიგში; ხოლო თუ

ერთარხიანი მოწყობილობა არ არის დაკავებული, მაშინ ტრანზაქტი შედის ბლოკში და აღნიშნული მოწყობილობის სტატუსი იცვლება და იგი დაკავებული ხდება. ასეთი მდგომარების აღწერისათვის გამოიყენება **SEIZE** ბლოკი. მას გააჩნია შემდეგი ზოგადი სტრუქტურა:

SEIZE A

სადაც **A** ოპერანდი დაკავებული ერთარხიანი მოწყობილობის სახელია.

მომსახურების შემდეგ ეს ტრანზაქტი უნდა შევიდეს სხვა ბლოკში, ამისათვის კი უნდა გაანთავისუფლოს დაკავებული მოწყობილობა. ამგვარი მდგომარეობის აღწერისათვის გამოიყენება **RELEASE** ბლოკი. მას გააჩნია შემდეგი ზოგადი სტრუქტურა:

RELEASE A

სადაც **A** ოპერანდი განთავისუფლებული ერთარხიანი მოწყობილობის სახელია.

2.4. მომსახურების იმიტაცია (ტრანზაქტის იმიტაციაზე დახარჯული დრო). ADVANCE ბლოკი

ჩვეულებრივ ტრანზაქტი იკავებს ერთარხიან მოწყობილობას იმისათვის, რომ მოხდეს მისი მომსახურება რაღაც სამოდელო დროის განმავალობაში. ამ დროის განმავლობაში ტრანზაქტი არ მოძრაობს ბლოკიდან ბლოკში. მომსახურების დროის გასვლის შემდეგ ის უნდა შევიდეს **RELEASE** ბლოკში, რათა გაანთავისუფლოს ერთარხიანი მოწყობილობა.

ტრანზაქტის შეყოვნებისათვის რაღაც სამოდელო დროის განმავლობაში გამოიყენება **ADVANCE** ბლოკი. ყველაზე ხშირად ეს დრო მოიცემა შემთხვევითი ცვლადებით. ამ შემთხვევაში **CENERATE** ბლოკის მსგავსად მომსახურების დრო მოიცემა **A** და **B** ოპერანდებით. მას გააჩნია შემდეგი ზოგადი სტრუქტურა:

ADVANCE A, [B]

სადაც **A** ოპერანდი არის მოდელში ტრანზაქტის მომსახურების საშუალო დრო;

B ოპერანდი განისაზღვრება **A** ოპერანდზე დამოკიდებულებით, ისე როგორც **CENERATE** ბლოკის შემთხვევაში.

2.5. ერთარხიანი მოწყობილობის მდგომარეობის შემოწმება

წარმოიდგინეთ, რომ რაიმე სისტემაში არსებული მოდელი, რომლის დამუშავებასაც თქვენ აპირებთ, არის მაგალითად, მონაცემთა გადაცემის ორი არხი – ორი ერთარხიანი მოწყობილობა (ЭВМ). ბუნებრივია, თუკი ერთი არხი დაკავებულია, მაშინ საჭიროა გაარკვიოთ: თავისუფალია კი მეორე არხი? თუ იგი დაკავებული არ არის, მონაცემთა გადაცემას უნდა შეეცადოთ მეორე არხის საშუალებით. მაგრამ იბადება კითხვა: როგორ შეამოწმოთ მოდელში არხების მდგომარეობა? მოწყობილობათა მდგომარეობის შესამოწმებლად გამოიყენება **GATE** და **TEST** ბლოკები.

2.5.1. ერთარხიანი მოწყობილობის მდგომარეობის შემოწმება GATE ბლოკით

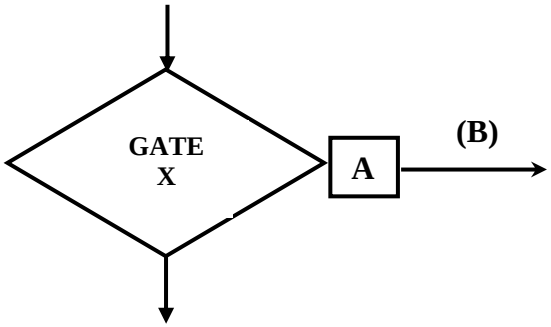
GPSS World-ში **GATE** ბლოკი გამოიყენება მოწყობილობათა მდგომარეობის განსაზღვრის მიზნით მათი მდგომარეობის შესაცვლელად. აღნიშნული ბლოკი მუშაობს ორ რეჟიმში:

- შესვლაზე უარი;
- შესვლაზე ნების დართვა და ალტერნატიული გამოსვლა (პირობითი გადასვლის რეჟიმი).

GATE ბლოკს გააჩნია ჩაწერის შემდეგი ფორმატი:

GATE X A, [B]

ხოლო გრაფიკულად ასე გამოისახება:



ნახ.2.1 **GATE** ბლოკი დამხმარე ოპერატორითა
და **A** და **B** ოპერანდებით

A ოპერანდი განსაზღვრავს ერთარხიანი მოწყობილობის სახელს ან ნომერს.

A და **B** ოპერანდები შესაძლოა იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სისტემური რიცხვითი ატრიბუტები.

პირობა შესაძლებელია გამოსახულ იქნას **X** პირობითი ოპერატორიდან ერთ-ერთი ფორმით, რომლებიც დაკავშირებულია ერთარხიან მოწყობილობასთან:

NU – A ოპერანდით მითითებული ერთარხიანი მოწყობილობა თავისუფალია;

U – A ოპერანდით მითითებული ერთარხიანი მოწყობილობა დაკავებულია. როგორც ჩანს, შესამოწმებელი ობიექტის ტიპი არანათლად წესდება პირობითი ოპერატორის მიერ.

თუ **B** ოპერანდი მითითებულია, მაშინ პირობის შემოწმება მიმდინარეობს პირობითი გადაცემის რეჟიმში და პირობის

მიხედვით ტრანზაქტი შესაძლებელია გადაადგილდეს ან მომდევნო ბლოკში ან **B** ოპერანდით მითითებულ ბლოკში. იმ შემთხვევაში თუ პირობით ოპერატორს გააჩნია მნიშვნელობა „ჭეშმარიტი“, მაშინ ტრანზაქტი გადაადგილდება შემდეგ ბლოკში, წინააღმდეგ შემთხვევაში კი – **B** ოპერანდით მითითებულ ბლოკში.

იმ შემთხვევაში თუ **B** ოპერანდი არ არის მითითებული, მაშინ პირობის შეუსრულებლობისას ტრანზაქტს არ შეუძლია შევიდეს **GATE** ბლოკში (უარის რეჟიმი). ამ შემთხვევაში ის მოთავსდება ყველა ობიექტის განმეორებითი მოსინჯვების სიაში, რომლებიც საჭიროებენ პირობის შემოწმებას. როდესაც რომელიმე ობიექტის მდგომარეობა შეიცვლება, განმეორებითი მოსინჯვების სიიდან ტრანზაქტი აქტიურდება და მიმდინარეობს პირობის ხელახალი შემოწმება. თუ პირობა სრულდება, ტრანზაქტს ნება ერთვება შევიდეს **GATE** ბლოკში და გადაადგილდეს შემდეგ ბლოკში.

მაგალითად:

```
GATE   U    Zrk
GATE   NU   (FN$Expert+4)
GATE   NU   Bab, Oper
```

პირველ მაგალითში **GATE** ბლოკი არ გაატარებს ტრანზაქტს **Zrk** სახელწოდების მქონე ერთარხიანი მოწყობილობის დაკავებულობის პირობის დროს. მეორე შემთხვევაში – როდესაც დაკავებულია ერთარხიანი მოწყობილობა, რომლის ნომერიც განისაზღვრება როგორც **FN$Expert+4** ფრჩხილებში ჩასმული გამოსახულების გამოთვლისა და შემდგომი დამრგვალების შედეგი. მესამე მაგალითში **Bab** სახელწოდების მქონე ერთარხიანი

მოწყობილობის დაკავებულობის შემთხვევაში ტრანზაქტი გადაადგილდება **Oper** სახელწოდებით მითითებულ ბლოკში.

შენიშვნა: **GATE** ბლოკის გამოყენების მოხერხებულობის (ხელსაყრელობის) მიუხედავად, ცალკეულ შემთხვევებში დაბლოკილი ტრანზაქტებისათვის შემოწმებათა ჩატარებამ იმ შემთხვევაში, როდესაც **B** ოპერანდი არ გამოიყენება, მან შეიძლება მიგვიყვანოს სამანქანო დროის გაზრდამდე. შემოწმებათა რიცხვის შემცირება და შემოწმებას დაქვემდებარებულ იმ ტრანზაქტთაგან, რომელთათვისაც საეჭვოა, რომ შემოწმების შედეგი ოდესმე აღმოჩნდეს „ჭეშმარიტი“, გამონაკლისი შესაძლოა დაშვებულ იქნას **LINK** და **UNLINK** ბლოკების მეშვეობით. აღნიშნული ბლოკების გამოყენების შესაძლებლობა ჩვენს მიერ განხილული იქნება VII თავში. სხვადასხვაგვარი ობიექტების, მათ შორის, ერთარხიანი მოწყობილობის მდგომარეობის შესამოწმებლად შეიძლება გამოყენებულ იქნას ბულის ცვლადები და **TEST** ბლოკი.

2.5.2. ერთარხიანი მოწყობილობის მდგომარეობის შემოწმება ბულის ცვლადების გამოყენებით და TEST ბლოკით

TEST ბლოკი აღწერს პირობას, რომელიც მოწმდება მასში ტრანზაქტის შესვლისას და შესული ტრანზაქტისათვის განსაზღვრავს მომდევნო ბლოკის ნომერს იმისგან დამოკიდებულებით, სრულდება თუ არა წაყენებული პირობა.

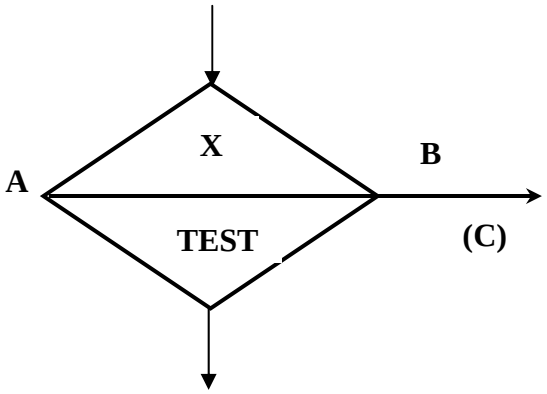
TEST ბლოკი ისევე როგორც **GATE** ბლოკი ფუნქციონირებს ორ რეჟიმში:

- შესვლაზე უარი;
- შესვლაზე ნების დართვა და ალტერნატიული გამოსვლა (პირობითი გადასვლის რეჟიმი).

TEST ბლოკს გააჩნია ჩაწერის შემდეგი ფორმატი:

TEST X A, B, [C]

ხოლო გრაფიკულად ასე გამოისახება:



ნახ. 2.2. **TEST** ბლოკი დამხმარე ოპერატორითა და **A, B** და **C** ოპერანდებით

სადაც **A** და **B** ოპერანდები შესადარებელი სიდიდეებია. ისინი შესაძლებელია იყოს რიცხვები, სტრიქონული ტიპის მონაცემები,

ფრჩხილებში ჩასმული გამოსახულებები, სტრა ან სტრა-ის პარამეტრები.

X პირობითი ოპერატორი შემდეგი ექვსი პირობითი ოპერატორიდან შესაძლებე-ლია გამოსახულ იქნას ერთ-ერთი ფორმით (იხ. ცხრილი 1. 3): **L** – ნაკლებია; **LE** – ნაკლებია ან ტოლი; **E** – ტოლი; **NE** – განსხვავებული; **G** – მეტია; **GE** – მეტია ან ტოლი.

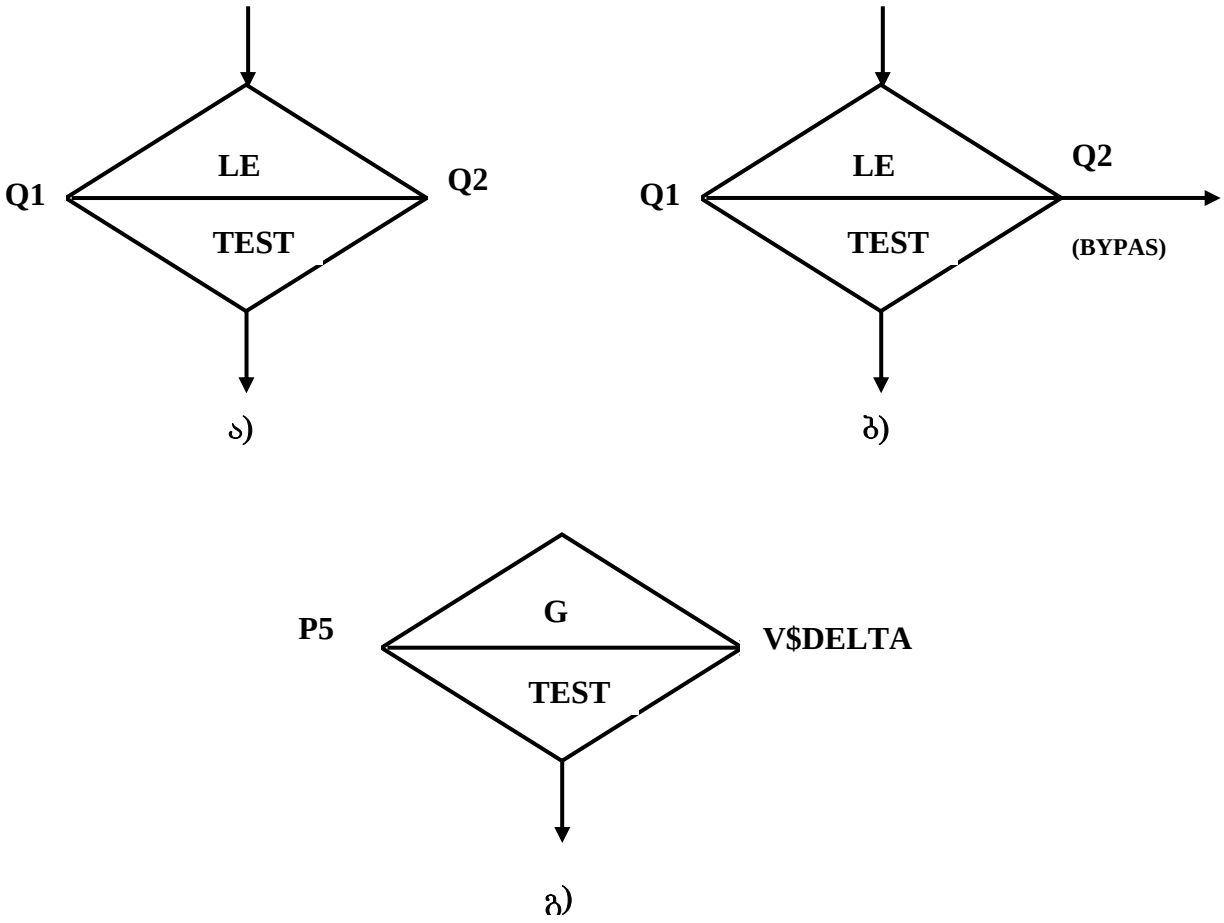
თუ **C** ოპერანდი მითითებულია, მაშინ პირობის შემოწმება მიმდინარეობს პირობითი გადაცემის რეჟიმში და პირობის მიხედვით ტრანზაქტი შესაძლებელია გადაადგილდეს მომდევნო ბლოკში ან **C** ოპერანდით მითითებულ ბლოკში. იმ შემთხვევაში თუ პირობა სრულდება, მაშინ ტრანზაქტი გადაადგილდება შემდეგ ბლოკში, წინააღმდეგ შემთხვევაში კი – **C** ოპერანდით მითითებულ ბლოკში.

იმ შემთხვევაში თუ **C** ოპერანდი არ არის მითითებული, მაშინ პირობის შეუსრულებლობისას ტრანზაქტს არ შეუძლია შევიდეს **TEST** ბლოკში (უარის რეჟიმი). ამ შემთხვევაში ის მოთავსდება ყველა ობიექტის განმეორებითი მოსინჯვების სიაში, რომლებიც საჭიროებენ პირობის შემოწმებას. როდესაც რომელიმე ობიექტის მდგომარეობა შეიცვლება, განმეორებითი მოსინჯვების სიიდან ტრანზაქტი აქტიურდება და მიმდინარეობს პირობის ხელახალი შემოწმება. თუ პირობა სრულდება, ტრანზაქტს ნება ერთვება შევიდეს **TEST** ბლოკში და გადაადგილდეს შემდეგ ბლოკში.

TEST ბლოკის მიმდინარე შემცველობის მნიშვნელობა შეიძლება განსხვავდებოდეს ნულისაგან. შემოწმების რეჟიმის

გარდა ამგვარი სიტუაცია ყოველთვის წარმოიშობა როდესაც ტრანზაქტისათვის დაგეგმილი მომდევნო ბლოკი უარს აცხადებს მის მიღებაზე. თუკი ტრანზაქტი გაჩერდება **TEST** ბლოკში, მისთვის მომდევნო ბლოკის ნომერი უკვე განსაზღვრულია. დაყოვნებული ტრანზაქტისათვის არ წარმოადგენს აუცილებლობას ხელახლა შეასრულოს **TEST** ბლოკის ქვეპროგრამა. ნახაზზე ნაჩვენებია **TEST** ბლოკის რამდენიმე მაგალითი.

2.3. ა ნახაზზე ტრანზაქტი წინა ბლოკში შეფერხებული იქნება მანამ, სანამ რიგი 1-ის შემცველობა არ იქნება რიგი 2-ის შემცველობაზე ნაკლები ან ტოლი. 2.3. ბ ნახაზზე ტრანზაქტი ასევე გადაადგილდება მომდევნო ბლოკში თუკი რიგი 1-ის შემცველობა ნაკლებია ან ტოლი რიგი 2-ის შემცველობაზე. თუკი ეს პირობა არ სრულდება ტრანზაქტი გადავა **BYPAS** სახელის მქონე ბლოკში. 2.3. გ ნახაზზე ტრანზაქტი იძულებულია მოიცადოს იმ დრომდე, ვიდრე მისი **P5** სიდიდე არ აღმოჩნდება **DELTA** ცვლადის სიდიდეზე მეტი. თუმცა არითმეტიკული ოპერაციების გამოყენება **TEST** ბლოკში იკრძალება, მათი გამოყენება დასაშვებია ცვლადების განსაზღვრისას. ამგვარად, ვიყენებთ რა არითმეტიკულ ცვლადებს **A** და/ან **B** ოპერანდებში, შეიძლება არითმეტიკულ გამოსახულებათა მნიშვნელობების შედარება **TEST** ბლოკში. ბულის ცვლადში შესაძლებელია გამოვიყენოთ სტრა **F**, იგი ტოლია 1 თუ ერთარხიანი მოწყობილობა დაკავებულია, წინააღმდეგ შემთხვევაში – 0.



ნახ.2.3. **TEST** ბლოკის გამოყენების მაგალითები:
ა – უარის რეჟიმი; ბ – პირობითი გადასვლის რეჟიმი; გ – უარის რეჟიმი

ქვემოთ მოყვანილია ერთარხიანი მოწყობილობის სტრა-ები:

- **FC** – ტრანზაქტების რაოდენობა, რომლებიც **SEIZE** და **FREEMPT** ბლოკების მეშვეობით იკავებენ ერთარხიან მოწყობილობას;
- **FR** – ერთარხიანი მოწყობილობის გამოყენების კოეფიციენტი;
- **FT** – ერთარხიანი მოწყობილობის მომსახურების საშუალო დრო ერთი ტრანზაქტის შემთხვევაში.

ქვემოთ მოყვანილია მაგალითი რომელიც ბულის ცვლადის და **TEST** ბლოკის გამოყენებით ასახავს ერთარხიანი მოწყობილობის მდგომარეობის შემოწმებას.

```
Prov   BVARIABLE      F$Server
                . .       .
               TEST E              BV$Prov, 0, PotZap
               SEIZE          Server
               ADVANCE        P$Obrab
               RELEASE         Server
                        .     .     .
```

მოყვანილ მაგალითში **TEST** ბლოკში ტრანზაქტის შესვლისას გამოითვლება **Prov** ბულის ცვლადი. ბულის ცვლადში გამოიყენება ერთარხიანი მოწყობილობის სტრა **F.** თუ **Server** სახელით მოცემული ერთარხიანი მოწყობილობა დაკავებულია, მაშინ **Prov** ბულის ცვლადის მნიშვნელობა 1-ის ტოლია. პირობა კი, რომელიც მოცემულია **TEST** ბლოკით, არ შესრულდება, ასე, რომ 1≠0 და ტრანზაქტს არ შეუძლია დაიკავოს ერთარხიანი მოწყობილობა. იგი გადადის **C** ოპერანდით მითითებულ ბლოკში, რომლის ჭდეა **PotZap.** იმ შემთხვევაში თუ **Server** სახელით მითითებული ერთარხიანი მოწყობილობა თავისუფალია, მაშინ **Prov** ბულის ცვლადის მნიშვნელობა 0-ის ტოლია. ამ შემთხვევაში **TEST** ბლოკში მითითებული პირობა სრულდება და ტრანზაქტი გადადის მომდევნო ბლოკში, ანუ დაიკავებს **Server** სახელით მითითებულ ერთარხიან მოწყობილობას.

2.6 რიგის რეგისტრატორი.
QUEUE და DEPART ბლოკები

GPSS-ში ობიექტის ტიპი „რიგი“ გამოიყენება სტატისტიკური მონაცემების შეგროვებისათვის. აღნიშნულმა სტატისტიკამ უნდა გასცეს პასუხი შემდეგ შეკითხვებს:

1. რამდენი ტრანზაქტი შემოვიდა მოდელში?
2. ფაქტიურად რამდენი მოსული ტრანზაქტი ჩადგა რიგში და რამდენმა ტრანზაქტმა დაიკავა მოწყობილობა ურიგოდ?
3. რა სიგრძის იყო რიგი?
4. რიგში მყოფი ტრანზაქტების საშუალო რიცხვი?
5. რიგში მყოფი ტრანზაქტების დაყოვნების (რიგში ყოფნის) საშუალო დრო?

GPSS-ში შესაძლებელია ასეთი სახის სტატისტიკის უზრუნველყოფა ე. წ. რიგის რეგისტრატორის დახმარებით. ამ მიზნით მოდელში გამოიყენება ორი ურთიერთდაკავშირებული ბლოკების წყვილი: ბლოკი **QUEUE** (რიგში ჩადგომა) და ბლოკი **DEPART** (რიგიდან წასვლა). ბლოკ **QUEUE**-ს გააჩნია შემდეგი ზოგადი სტრუქტურა:

QUEUE A, [B]

ბლოკი **QUEUE** რიგის სიგრძეს ზრდის ერთით. **A** ოპერანდი აღნიშნავს რიგის სახელს ან ნომერს, რომელიც იზრდება ერთით. **B** ოპერანდი განსაზღვრავს ტრანზაქტების დამატების რიცხვს

(ფარულად იგი ერთის ტოლია), რომლითაც იზრდება რიგის სიგრძე.

ბლოკ **DEPART**-ს გააჩნია შემდეგი ზოგადი სტრუქტურა:

DEPART A, [B]

ბლოკი **DEPART** ემსახურება რიგის სიგრძის ერთით შემცირებას. **A** ოპერანდი აღნიშნავს რიგის სახელს ან ნომერს, რომელიც უნდა შემცირდეს ერთით. **B** ოპერანდი განსაზღვრავს იმ რიცხვს, რომლითაც უნდა შემცირდეს რიგი. ეს რიცხვი არ უნდა აღემატებოდეს რიგის სიგრძეს. თუ **B** ოპერანდი არ გამოიყენება, მაშინ რიგი შემცირდება ერთით.

პროგრამა **GPSS** ენაზე შედგება შემდეგი განყოფილებებისაგან:

ცხრილი 2.1

ველი	ბლოკის დასახელება	ოპერანდები	კომენტარი
BK	**ADVANCE**	**30,5**	**;შემდეგი დეტალის აწყობა**

2.7. მოდელში ტრანზაქტების გადაადგილების მარშრუტების ცვლილების მეთოდები

მოდელში ტრანზაქტების გადაადგილების მარშრუტების ცვლილებისას გამოიყენება **GATE, TEST, TRANSFER, LOOP** და **DISPLACE** ბლოკები.

GATE და **TEST** ბლოკები ერთარხიანი მოწყობილობებისათვის უკვე იყო განხილული. რაც შეეხება **TRANSFER, LOOP** და **DISPLACE** ბლოკების გამოყენების მეთოდებს შემოგთავაზებთ ქვევმოთ.

2.7.1. ტრანზაქტების გადასვლა. TRANSFER ბლოკი

ხშირად წარმოიქმნება ისეთი სიტუაციები, როდესაც საჭიროა გარკვეული სახის ბლოკების განმეორება. ამისათვის გამოიყენება **TRANSFER** ბლოკი. იგი განკუთვნილია მოდელში შემავალი ტრანზაქტების ნებისმიერ სხვა ბლოკში გადასაცემად. მას გააჩნია შემდეგი ზოგადი სახე:

TRANSFER [A], [B], [C], [D]

TRANSFER ბლოკის ყველა რეჟიმი უპირობო გადასვლის რეჟიმის გარდა, არის ამორჩევითი. ეს რეჟიმები განსხვავდებიან ამორჩევის პირობის მიხედვით. ამორჩევის რეჟიმს განსაზღვრავს **A**

ოპერანდი. არსებობს **TRANSFER** ბლოკის მუშაობის შემდეგი რეჟიმები:

- , – უპირობო გადასვლის;
- . – სტატისტიკური, რომელიც ირჩევს ორი ბლოკიდან ერთ-ერთს;
- **BOTH** – ორი ბლოკიდან ერთ-ერთის თანამიმდევრობით ამორჩევა;
- **ALL** – რამდენიმე ბლოკიდან თანამიმდევრობით ერთ-ერთის ამორჩევა;
- **PICK** – რამდენიმე ბლოკიდან შემთხვევითი ამორჩევა;
- **FN** – ფუნქციონალური;
- **P** – პარამეტრული;
- **SBR** – ქვეპროგრამის;
- **SIM** – ერთდროული.

A ოპერანდმა შეიძლება მიიღოს ზემოთ ჩამოთვლილიდან ერთ-ერთი მნიშვნელობა, აგრეთვე, შესაძლებელია იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი..

B და **C** ოპერანდები განსაზღვრავენ იმ ბლოკების ნომრების შესაძლო მნიშვნელობებს ან მათ მდგომარეობებს, რომლებშიც უნდა შევიდეს აქტიური ტრანზაქტი. ეს ოპერანდები იმავე ტიპის უნდა იყვნენ რა ტიპისაც არის **A** ოპერანდი. **B** ოპერანდის არარსებობის შემთხვევაში მის ადგილზე ჩაიწერება იმ ბლოკის სახელი, რომელშიც **TRANSFER** ბლოკიდან უნდა შევიდეს ტრანზაქტი.

განვიხილოთ **TRANSFER** ბლოკის მუშაობის რეჟიმები.

2.7.1.1. უპირობო გადასვლის რეჟიმი

უპირობო გადასვლის რეჟიმის დროს **A** ოპერანდი არ გამოიყენება. **B** ოპერანდი არის იმ ბლოკის სახელი, რომელშიც უნდა შევიდეს აღნიშნული ტრანზაქტი. მაგალითად:

TRANSFER ,Oper

TRANSFER ბლოკიდან ტრანზაქტი გადადის იმ ბლოკში, რომლის ჭდეა **Oper**.

2.7.1.2 სტატისტიკური გადასვლის რეჟიმი

TRANSFER ბლოკი სტატისტიკური გადაცემის რეჟიმში გამოიყენება მაშინ, როდესაც ტრანზაქტები შემთხვევითი სახით ერთი ბლოკიდან უნდა გადავცეთ ორ ბლოკს.

A ოპერანდის მნიშვნელობა, რომელიც ჩაიწერება ათობითი წერტილის შემდეგ სამნიშნა რიცხვით, განსაზღვრავს ტრანზაქტების რა წილი უნდა შევიდეს **C** ბლოკში, ხოლო ტრანზაქტების დანარჩენი ნაწილი მიემართება **B** ბლოკში. ამ უკანასკნელი ოპერანდის არ არსებობის შემთხვევაში – იმ ბლოკში, რომელიც **TRANSFER** ბლოკს მოსდევს.

ტრანზაქტების სტატისტიკური გადაცემის რეჟიმში მიმდინარეობს იმ ბლოკების ამორჩევა, სადაც უნდა შევიდეს აქტიური ტრანზაქტი **TRANSFER** ბლოკიდან. შესაძლებელია **TRANSFER** ბლოკმა უარი თქვას ტრანზაქტის შესვლაზე. თუ **TRANSFER** ბლოკიდან გადასვლის შემდეგ ბლოკმა სადაც უნდა შევიდეს ეს ტრანზაქტი უარი თქვა მის შესვლაზე, მაშინ ეს ტრანზაქტი ისევ რჩება **TRANSFER** ბლოკში. ამის შემდეგ ტრანზაქტი კვლავ მოსინჯავს ამ ბლოკში შესვლას და რომელიღაც მოსინჯვის შემდეგ შეაღწევს აღნიშნულ ბლოკში.

სტატისტიკური გადასვლის რეჟიმი გამოიყენება ისეთ შემთხვევაში, მაგალითად, როდესაც ცნობილია რომ დეტალების საამქროში დეტალების 12,5% წუნდებულია. მოცემული შემთხვევა მოდელში ასე აღიწერება:

. . .

TRANSFER .125, Sam, Det

. . .

ტრანზაქტების 12,5%, რომლებიც ახდენენ საამქროში დამზადებული დეტალების იმიტაციას მიმართულია **Det** ჭდით აღნიშნულ ბლოკში, ხოლო დეტალების დანარჩენი 87,5%–ბლოკში ჭდით **Sam**.

2.7.1.3. BOTH რეჟიმი

თუ **A** ოპერანდის ნაცვლად გამოიყენება მომსახურე სიტყვა **BOTH** (ორივე), მაშინ **TRANSFER** ბლოკი მუშაობს **BOTH** რეჟიმში.

აღნიშნულ რეჟიმში მუშაობისას **TRANSFER** ბლოკში შემავალი თითოეული ტრანზაქტი ამოწმებს ორ გზას. თავდაპირველად მოწმდება **B** ბლოკში შესვლის შესაძლებლობა. თუ ტრანზაქტს არ შეუძლია ამ ბლოკში შესვლა, მაშინ ის ეცდება **C** ბლოკში შესვლას. თუ ამ ბლოკშიც ვერ შევიდა ტრანზაქტი, მაშინ ის გაჩერდება **TRANSFER** ბლოკში. **B** და **C** ბლოკებში შესვლის მიმართვა მეორდება ყოველი სამოდელო დროის ცვლილებისას იქამდე, ვიდრე ტრანზაქტი არ მიიღებს რომელიმე ბლოკში შესვლის შესაძლებლობას. თუ **B** ოპერანდი არ გამოიყენება, მაშინ მოწმდება შემდეგ ბლოკში შესვლის შესაძლებლობა.

2.7.1.4. ALL რეჟიმი

ALL რეჟიმში მუშაობისათვის **A** ოპერანდის რანგში გამოიყენება მომსახურე სიტყვა **ALL**. ამ რეჟიმში მუშაობისას **TRANSFER** ბლოკში შესული ტრანზაქტი ამოწმებს ნებისმიერ ბლოკში შესვლის შესაძლებლობას **B** ოპერანდით მითითებული ბლოკიდან დაწყებული **C** ოპერანდით მითითებული ბლოკით დამთავრებული. **D** ოპერანდი განსაზღვრავს შესამოწმებელი ბლოკის ნომრის ცვლილების **d** ბიჯს. ბიჯის მნიშვნელობა შესაძლებლობას იძლევა გამოკითხულ იქნას **B** და **C** ოპერანდებით მითითებულ ბლოკებს შორის მოთავსებული განსაზღვრული ბლოკები. **TRANSFER** ბლოკში შესული ტრანზაქტი ცდილობს შევიდეს **B** ოპერანდით მითითებულ ბლოკში. თუ ეს ბლოკი დაკავებულია, მაშინ ტრანზაქტი ეცდება შევიდეს

ბლოკებში შემდეგი ნომრებით **v** + **d, v + 2d**, …, **z**, სადაც **v** არის **B** ბლოკის ნომერი, **z** – **C** ბლოკის ნომერი. ამ დროს **z = v + nd**, სადაც **n** – მთელი დადებითი რიცხვია.

თუ **C** ოპერანდი არ გამოიყენება, მაშინ მოწმდება მხოლოდ ერთი ბლოკი. ჩვეულებრივ, **B** და **C** ოპერანდების რანგში ჩაიწერება ბლოკების ჭდეები, ამისათვის საჭიროა ბლოკები ერთმანეთის მიყოლებით იმგვარად განლაგდეს, რომ **B** და **C** ოპერანდებით მითითებულ ბლოკების ნომრებს შორის სხვაობა ნომრების მინიჭებისას იყოს **D** ოპერანდში მითითებული ბიჯის ჯერადი. მაგალითად,

TRANSFER ALL, Wde1,Wde2, 4

ამ მაგალითში **ALL** რეჟიმში შეღწევა შესაძლებელია მაშინ, თუ **Wde1** და **Wde2** ჭდეებით მითითებულ ბლოკების ნომრებს შორის სხვაობა 4-ის ჯერადია.

მაგალითი 2.1. განაცხადი განაწილებულია სამი ერთარხიანი მოწყობილობის მიხედვით. მორიგ განაცხადს მოემსახურება პირველი განთავისუფლებული ერთარხიანი მოწყობილობა. ამ მაგალითის შეასაბამის მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                    CPSS WORLD SIMULATION

*       განაცხადის შესვლისა და დამუშავების იმიტაციის სეგმენტი

        GENERATE (Exponencial (315, 0, 11.7))   ; განაცხადების შემოსვლა
        TRANSFER  ALL, Wde1, Wde2               ; განაცხადების განაწილება

*           I ერთარხიანი მოწყობილობის მუშაობის იმიტაცია

 Wde1   SEIZE     Apd1      ; I ერთარხიანი მოწყობილობის დაკავება
        ADVANCE   23. 5, 12. 1      ; განაცხადის მომსახურება
        RELEASE   Apd1              ; I ერთარხიანი მოწყობილობის
                                      განთავისუფლება
        TERMINATE                   ; I ერთარხიანი მოწყობილობის
                                      მომსახურებული განაცხადები

*           II ერთარხიანი მოწყობილობის მუშაობის იმიტაცია

        SEIZE     Apd2      ; II ერთარხიანი მოწყობილობის დაკავება
        ADVANCE   (Normal(357, 28. 9, 13. 8))   ; განაცხადის
                                                  მომსახურება
        RELEASE Apd2 ; II ერთარხიანი მოწყ-ის განთავისუფლება
        TERMINATE    ; II ერთარხიანი მოწყობილობის მომ-
                         სახურებული განაცხადები

*           III ერთარხიანი მოწყობილობის მუშაობის იმიტაცია

  Wde2  SEIZE     Apd3      ; III ერთარხიანი მოწყ-ის დაკავება
        ADVANCE   (Exponencial (15, 0, 15.7))  ; განაცხადის
                                                 მომსახურება
        RELEASE Apd3 ; III ერთარხიანი მოწყ-ის განთავისუფლება
        TERMINATE    ; III ერთარხიანი მოწყობილობის მომ-
                         სახურებული განაცხადები

*       მოდელირების დროის სეგმენტი

        GENERATE     3600           ; მოდელირების დრო
        TERMINATE    1              ; დასასრული
```

2.7.1.5. PICK რეჟიმი

TRANSFER ბლოკის **PICK** რეჟიმში მუშაობისას **A** ოპერანდის ნაცვლად გამოიყენება მომსახურე სიტყვა **PICK**. ამ რეჟიმში მუშაობისას **v, v+1, v+2, …, n** ბლოკების თანამიმდევრობიდან, სადაც **v** – ბლოკის ნომერი მოცემულია **B** ოპერანდით, ხოლო **n** არის იმ ბლოკის ნომერი, რომლითაც მოცემულია ოპერანდი **C,** შემთხვევით აირჩევა იმ ბლოკის ნომერი, სადაც უნდა გადავიდეს ტრანზაქტი. **B** და **C** ოპერანდები აირჩევა თანაბარალბათურად **1/[(n-v)+1].** ტრანზაქტი ეცდება გადავიდეს მხოლოდ მისთვის შერჩეულ ბლოკში. თუ ტრანზაქტს არ შეუძლია მაშინვე გადავიდეს შემდეგ ბლოკში, მაშინ ის უნდა დაელოდოს **TRANSFER** ბლოკს ვიდრე არ მოიხსნება მაბლოკირებელი პირობა. **C** ბლოკის ნომერი მეტია ან ტოლია **v+1.**

2.7.1.6. ფუნქციონალური რეჟიმი

TRANSFER ბლოკის ფუნქციონალურ რეჟიმში მუშაობისას **A** ოპერანდის ნაცვლად გამოიყენება მომსახურე სიტყვა **FN**. **TRANSFER** ბლოკში ტრანზაქტების შესვლისას გამოითვლება ფუნქციის მნიშვნელობა, რომლის სახელი მოცემულია **B** ოპერანდით. თუ გამოთვლის შედეგი ნამდვილი რიცხვია, მაშინ მისგან აიღება მთელი ნაწილი. შემდეგი ბლოკის ნომრის განსაზღვრისათვის გამოიყენება მიღებული მთელი რიცხვი,

რომელიც მოიცემა **C** ოპერანდის სახით (**C** ოპერანდის მნიშვნელობა შესაძლებელია იყოს ნულის ტოლი). თუ ბლოკი გამოთვლილი ნომრით დაკავებულია, მაშინ ტრანზაქტი რჩება **TRANSFER** ბლოკში ვიდრე ამ ბლოკში არ გადავა. მაგალითად:

TRANSFER FN, MniS, P6
TRANSFER FN, 7, P6

პირველ მაგალითში ტრანზაქტი მიემართება იმ ბლოკში, რომლის ნომერი მიიღება, როგორც უახლოეს მთელ რიცხვამდე დამრგვალებული **MniS** სახელით გამოთვლილი ფუნქციის მნიშვნელობისა და **TRANSFER** ბლოკში შესული ტრანზაქტის მეექვსე პარამეტრის მნიშვნელობის შეკრების შედეგი. მეორე მაგალითშიც, იგივე პრინციპით განისაზღვრება შესასვლელი ტრანზაქტის ბლოკის ნომერი, განსხვავება მხოლოდ ისაა, რომ ფუნქცია მოცემულია არა სახელით, არამედ ნომრით.

2.7.1.7. პარამეტრული რეჟიმი

პარამეტრული რეჟიმის შემთხვევაში **A** ოპერანდის რანგში გამოიყენება მომსახურე სიტყვა **P.** აღნიშნულ რეჟიმში აქტიური ტრანზაქტი მიემართება იმ ბლოკისაკენ, რომლის ნომერი განისაზღვრება როგორც პარამეტრის მნიშვნელობისა და **C** ოპერანდის მნიშვნელობის ჯამი. თუ **C** ოპერანდი არ გამოიყენება,

მაშინ პარამეტრის ნომერი წარმოადგენს მომდევნო ბლოკის ნომერს. მაგალითად:

TRANSFER P, Axali, 2

TRANSFER P, 5, 2

პირველ მაგალითში ტრანზაქტი მიემართება იმ ბლოკისაკენ, რომლის ნომერი უდრის **Axali** სახელით მითითებულ პარამეტრის მნიშვნელობის და **C** ოპერანდის მნიშვნელობის ჯამს, ანუ 2-ს. მეორე შემთხვევაში პარამეტრი მოცემულია არა სახელით, არამედ ნომრით.

2.7.1.8. ქვეპროგრამის რეჟიმი

ქვეპროგრამის რეჟიმში მუშაობისათვის **A** ოპერანდის რანგში გამოიყენება მომსახურე სიტყვა **SBR.** ამ რეჟიმში აქტიური ტრანზაქტი ყოველთვის მიემართება **B** ოპერანდით მითითებული ბლოკისაკენ. **TRANSFER** ბლოკის ნომერი მოთავსებულია პარამეტრში, რომლის ნომერი მითითებულია **C** ოპერანდით. მაგალითად:

TRANSFER SBR, Axali, Saxli

აღნიშნულ ბლოკში შესული ტრანზაქტი მიემართება ბლოკისკენ, რომლის ნომერი მითითებულია **Axali** ჭდით, ხოლო **TRANSFER** ბლოკის ნომერი ჩაიწერება პარამეტრში **Saxli** სახელით. თუ ამგვარი პარამეტრი არ არსებობს, იგი იქმნება.

2.7.1.9. ერთდროული რეჟიმი

SIM (ერთდროული) რეჟიმი გამოიყენება იმ შემთხვევაში, როდესაც საჭიროა რამდენიმე პირობის ერთდროული შესრულება. თითოეულ ტრანზაქტს გააჩნია თავისი დაყოვნების **SIM** ინდიკატორი. ამ ინდიკატორში ჩაიწერება ტრანზაქტის ნებისმიერი მცდელობის შედეგი – შევიდეს მომდევნო ბლოკში. თუ დამპროექტებელი აღმოაჩენს პირობას, რომ ბლოკში ფერხდება ტრანზაქტის შესვლა, მაშინ ამ ტრანზაქტის **SIM** ინდიკატორი დგება 1-ში. თუ მომდევნო ბლოკში გადასვლის ყველა პირობა კმაყოფილდება, მაშინ **SIM** ინდიკატორი რჩება ნულის ტოლი. თუ არ სრულდება თუნდაც ერთი პირობა, მაშინ **SIM** ინდიკატორი დგება 1-ში. ასეთ შემთხვევებში **C** ოპერანდი მიუთითებს იმ ბლოკის ნომერს, რომელშიც შემოწმდა პირველი პირობა და ტრანზაქტი ასრულებს ყველა პირობის შემოწმებას ვიდრე არ მოხდება მათი ერთდროული დაკმაყოფილება. ამასთან, **ADVANCE** ბლოკის **SIM** ინდიკატორი დგება 0-ში. მდგომარეობის შემოწმება დაკავშირებულია იმ ბლოკებთან, რომელთაც შეუძლიათ ტრანზაქტების დაყოვნება. ირიბი შემოწმებისას გამოიყენება **GATE** ბლოკი. მაგალითად:

```
Wde 1GATE      NU   Mas1
       GATE    NU   Mas2
       GATE    NU   Mas3
       TRANSFER     SIM, Wde1
       SEIZE        Rem1
```

მოყვანილ მაგალითში ტრანზაქტს არ შეუძლია **SEIZE** ბლოკში შესვლა თუ **Mas1, Mas2** და **Mas3** ერთარხიანი მოწყობილობები არ იქნება ერთდროულად თავისუფალი. **TRANSFER** ბლოკში ტრანზაქტის შესვლისას მოწმდება მისი **SIM** ინდიკატორის მნიშვნელობა. თუ ტრანზაქტი დაყოვნებული იქნა **GATE** ბლოკიდან რომელიმეში, მაშინ მისი ინდიკატორი დგება 1-ში.

TRANSFER ბლოკში ინდიკატორის შემოწმებისას ირკვევა, რომ ტრანზაქტი იქნა დაყოვნებული, იგი გადაიგზავნება იმ ბლოკში, სადაც მიმდინარეობდა მისი პირველი შემოწმება (**Wde 1**) და შემდეგ შემოწმებათა სერიის ყველა თანამიმდევრობები მეორდება.

ASSEMBLE, GATHER და **MATCH** ბლოკებში ტრანზაქტების დაყოვნებისას **SIM** ინდიკატორი არ დგება 1-ში.

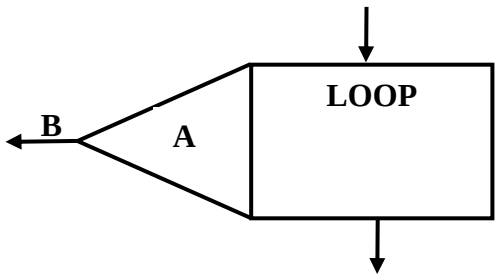
შენიშვნა: **SIM** რეჟიმი იშვიათად გამოიყენება, ამიტომ დიდი რაოდენობის ობიექტების მდგომარეობის სულ უფრო ეფექტურად მართვისათვის გამოიყენება ბულის ცვლადები ან **TEST** ბლოკი.

2.7.2. LOOP ბლოკი

LOOP ბლოკი განკუთვნილია მოდელში ციკლის ორგანიზებისათვის. მას გააჩნია შემდეგი ფორმატი:

LOOP A, B

ხოლო გრაფიკულად ასე გამოისახება:



ნახ. 2.4. **LOOP** ბლოკი და მისი **A** და **B** ოპერანდები

სადაც **A** ოპერანდი არის ტრანზაქტის პარამეტრი, ანუ ციკლის პარამეტრი, რომელიც გვიჩვენებს ციკლის განმეორების რიცხვს.

B ოპერანდი გვიჩვენებს ციკლის დასაწყისს.

LOOP ბლოკში ტრანზაქტის შესვლისას **A** პარამეტრის მნიშვნელობა მცირდება 1-ით, შემდეგ კი მოწმდება პირობა: ტოლია თუ არა მისი მნიშვნელობა ნულის. იმ შემთხვევაში თუ ეს მნიშვნელობა არ არის ნულის ტოლი, ტრანზაქტი კვლავ გადადის **B** ოპერანდით მითითებულ ბლოკში. როდესაც **A** ოპერანდის მნიშვნელობა გახდება ნულის ტოლი, ტრანზაქტი გამოვა ციკლიდან და შევა **LOOP** ბლოკის შემდეგ მდგარ ბლოკში.

შენიშვნა: ვინაიდან ციკლის ყოველ დაბრუნებაზე მისი პარამეტრის მნიშვნელობა 1-ით მცირდება, ეს ნიშნავს, რომ ციკლის პარამეტრის სიდიდე უნდა იცვლებოდეს დიდი მნიშვნელობიდან მცირე მნიშვნელობისაკენ და არა პირიქით.

2.7.3. DISPLACE ბლოკი

DISPLACE ბლოკი განკუთვნილია ნებისმიერი ტრანზაქტის მდებარეობის განსაზღვრისა და მომდევნო ბლოკში მის გადასაადგილებლად. აღნიშნულ ბლოკს გააჩნია შემდეგი სახის ფორმატი:

DISPLACE A, B, [C], [D]

A ოპერანდი არის ტრანზაქტის ნომერი, რომელიც აუცილებელია გადაადგილდეს;

B ოპერანდი – ბლოკის ჭდე, რომელშიც გადაადგილდება **A** ოპერანდით მითითებული ტრანზაქტი;

C ოპერანდი – გადასაადგილებელი ტრანზაქტის პარამეტრის ნომერი, რომელშიც ჩაიწერება მისი მომსახურების დასრულებამდე დარჩენილი დრო, თუ ის იმყოფება სამომავლოდ მოსახდენ მოვლენათა სიაში;

D ოპერანდი – აქტიური ტრანზაქტისათვის ალტერნატიული ბლოკის ჭდე.

A, B, C და **D** ოპერანდები შესაძლებელია იყოს დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი. მაგალითად:

DISPLACE (P2+32), Wde1, Sed, Wde2

A ოპერანდი მოცემულია ფრჩხილებში ჩასმული გამოსახულების სახით. ეს გამოსახულება გამოითვლება და დამრგვალდება მთელ რიცხვამდე. მიღებული შედეგი წარმოადგენს იმ ტრანზაქტის ნომერს, რომელიც აუცილებლად უნდა გადაადგილდეს. შემდეგ **DISPLACE** ბლოკი ეძებს აღნიშნულ ტრანზაქტს. ამისათვის დასაშვებია შემდეგი შემთხვევების განხილვა:

- ტრანზაქტი არის მოდელში და არ იმყოფება სამომავლო მოვლენათა სიაში;
- ტრანზაქტი არის მოდელში და იმყოფება სამომავლო მოვლენათა სიაში;
- შესაბამისი ნომრის მქონე ტრანზაქტი მოდელში არ არსებობს.

პირველ შემთხვევაში ტრანზაქტი გადაადგილდება **Wde1** ჭდით მითითებულ ბლოკში. მეორე შემთხვევაში განისაზღვრება დრო, რომელიც დარჩა მოდელირების პროცესში მის განმეორებით შესვლამდე და ჩაიწერება **Sed** სახელით მითითებულ პარამეტრში. თუ ასეთი პარამეტრი არ არსებობს, მაშინ ის იქმნება. ტრანზაქტი ასევე გადაადგილდება **Wde1** ჭდით მითითებულ ბლოკში. მესამე შემთხვევაში, ანუ როდესაც მოდელში საჭირო ნომრით მითითებული ტრანზაქტი არ არსებობს, მაშინ აქტიური ტრანზაქტი, რომელიც შედის **DISPLACE** ბლოკში, გადავა **Wde2**-ით მითითებულ ბლოკში. როდესაც ტრანზაქტი გადაადგილდება ახალ ბლოკში, ის *გამოირიცხება სიებიდან*:

- მომავალ მოვლენათა;

- გადადებული შეწყვეტის (შემწყვეტი ტრანზაქტებისათვის);
- დაყოვნების (პრიორიტეტის თანრიგში);
- მომხმარებლის;
- განმეორებითი მოსინჯვების.

  და არ გამოირიცხება სიებიდან:

- მიმდინარე მოვლენათა;
- შეწყვეტის (შეწყვეტილი ტრანზაქტებისათვის);
- ჯგუფური.

ტრანზაქტის გადაადგილებისას მოწყობილობებში შეწყვეტილი მოქმედებები არ იგნორირდება, რაც ნიშნავს, რომ გადაადგილებული ტრანზაქტი განაგრძობს მოწყობილობის დაკავებას.

III თავი
მოდელების აგება
მრავალარხიანი მოწყობილობებისათვის

ორი ან რამდენიმე მოწყობილობის მომსახურებისათვის გამოიყენება მრავალარხიანი მოწყობილობა ანუ პარალელურად მომუშავე არხები. მრავალარხიანი მოწყობილობაში შეიძლება გამოყენებულ იქნას რამდენიმე ტრანზაქტი. მოდელში ტრანზაქტების რაოდენობა შეუზღუდავია და მათ გააჩნიათ სახელები რათა განვასხვავოთ ერთმანეთისაგან. მრავალარხიანი მოწყობილობის გამოყენების შემთხვევაში მისი წინასწარ აღსაწერად გამოიყენება ბრძანება (ანუ ოპერატორი) **STORAGE**. მას გააჩნია შემდეგი ზოგადი სახე:

Name STORAGE A

სადაც **Name** მრავალარხიანი მოწყობილობის სახელია.

მრავალარხიანი მოწყობილობის ფუნქციონირების ორგანიზებისათვის შესაძლებელია შემდეგი ორი რეჟიმის არსებობა:

- მრავალარხიანი მოწყობილობის დაკავება ან განთავისუფლება;
- მრავალარხიან მოწყობილობის მიუწვდომლობა.

3.1. მრავალარხიანი მოწყობილობის დაკავება და მისი განთავისუფლება. ENTER და LEAVE ბლოკები

მრავალარხიანი მოწყობილობის დაკავებისა და განთავისუფლების იმიტაცია შესაძლებელია ბლოკებით **ENTER** და **LEAVE**. მათ გააჩნიათ შემდეგი ზოგადი სახე:

ENTER A, [B]
LEAVE A, [B]

სადაც **A** ოპერანდი ორივე შემთხვევაში არის მრავალარხიანი მოწყობილობის სახელი; ხოლო **B** ოპერანდი განსაზღვრავს მოწყობილობის რიცხვს (მეხსიერების ელემენტებს), რომლებმაც უნდა დაიკავონ **ENTER** ბლოკი და გაანთავისუფლონ **LEAVE** ბლოკი. ამ შემთხვევაში იგულისხმება რომ **B = 1**. როდესაც **B = 0** ნიშნავს რომ ეს ბლოკი არ იმყოფება მუშა მდგომარეობაში.

როდესაც ტრანზაქტი შედის **ENTER** ბლოკში, **A** ოპერანდში მითითებული სახელით წარმოებს მრავალარხიანი მოწყობილობის მოძებნა. თუ ასეთი სახელის მრავალარხიანი მოწყობილობა ვერ იქნა ნაპოვნი, მაშინ წარმოიქმნება შეცდომა „მიმართვა არარსებულ მეხსიერებასთან". ხოლო თუ ასეთი სახელის

მრავალარხიანი მოწყობილობა არსებობს და **B** ოპერანდი განსაზღვრულია მაშინ, მისი მნიშვნელობა გამოითვლება, დამრგვალდება მთელ რიცხვამდე და მიღებული შედეგი გამოიყენება თავისუფალი მეხსიერების შესაფასებლად. ტრანზაქტი **ENTER** ბლოკში უნდა შევიდეს მხოლოდ მაშინ როდესაც მრავალარხიანი მოწყობილობა იმყოფება ტრანზაქტების მიღების რეჟიმში და ტევადობა საკმარისია იმისათვის რომ მოთხოვნა დაკმაყოფილდეს. წინააღმდეგ შემთხვევაში ტრანზაქტი მოთავსდება მოწყობილობის დაყოვნების სიაში შესაბამისი პრიორიტეტით.

როდესაც ტრანზაქტი შედის **ENTER** ბლოკში, სრულდება შემდეგი მოქმედებები:

- მრავალარხიანი მოწყობილობაში მრიცხველი იზრდება ერთით;
- **B** ოპერანდის მნიშვნელობით იზრდება (იგულისხმება 1) მრავალარხიანი მოწყობილობის მიმდინარე შემცველობა;
- **B** ოპერანდის მნიშვნელობით მცირდება (იგულისხმება 1) მრავალარხიანი მოწყობილობის მისაღწევი ტევადობა.

თუ ტრანზაქტი **ENTER** ბლოკში შესასვლელად ითხოვს იმაზე მეტ მოწყობილობას (მეხსიერების ელემენტებს), ვიდრე აღნიშნული **STORAGE** ბრძანებითაა განსაზღვრული, ანუ თუ **ENTER** ბლოკში **A** ოპერანდი ნაკლებია **B** ოპერანდზე, წარმოიქმნება შეცდომა „შესაკითხი მეხსიერების ელემენტების (მოწყობილობების) რიცხვი აღემატება საერთო ტევადობას”.

მრავალარხიანი მოწყობილობა შეიძლება გადავაკეთოთ ანუ **STORAGE** ბრძანებით აღწერილ მრავალარხიანი მოწყობილობის სახელს შევუცვალოთ ტევადობა.

მაგალითად:

Bank SRORAGE 17

მეორედ აღწერისას მივიღებთ

Bank SRORAGE 47

დროის გარკვეულ მომენტში იმიტაციის მომსახურების დროის შუალედების აღსაწერად გამოიყენება ბლოკი **ADVANCE**.

მაგალითი 1:

Nak STORAGE 25

. . .

ENTER Nak, 2

ADVANCE 122, 45

LEAVE Nak, 2

STORAGE ბრძანებით განისაზღვრება მრავალარხიანი მოწყობილობის სახელი **Nak**25 ერთეულით. **ENTER** ბლოკში ტრანზაქტების შესვლისას 122±45 დროის ინტერვალის განმავლობაში დაკავებული იქნება 2 მოწყობილობა და მრავალარხიანი მოწყობილობიდან ტრანზაქტების გამოსვლისას **LEAVE** ბლოკის საშუალებით განთავისუფლდება ამდენივე მოწყობილობა.

მაგალითი 2:

```
Nak   STORAGE     25
      . . .
      ENTER     Nak, 26
      ADVANCE   122, 45
      LEAVE     Nak, 2
```

ENTER ბლოკში ტრანზაქტების შესვლისას წარმოიქმნება შეცდომა, რადგან ტრანზაქტებს (26), რომელთაც სურს ბლოკში შესვლა უფრო მეტია, ვიდრე ეს **STORAGE** ბრძანებითაა განსაზღვრული. იგივე წარმოიქმნება ტრანზაქტების **LEAVE** ბლოკიდან გამოსვლისას, რადგან ყველა ტრანზაქტი არ ტოვებს მოწყობილობას.

მაგალითი 3:

```
Pun1  EQU          1
Pun2  EQU          2
Pun3  EQU          3
Pun1  STORAGE          6
Pun2  STORAGE          5
Pun3  STORAGE          2
      . . .
      ENTER        *1
      ADVANCE          MX$NorVr (P2,P3)
      LEAVE        *1
      . . .
```

მოცემულ მაგალითში განსაზღვრულია 3 მრავალარხიანი მოწყობილობა **Pun1, Pun2, Pun3** სახელებით და შესაბამისად 6, 5,

2 ტევადობებით. იგულისხმება, რომ **ENTER** ბლოკში ტრანზაქტების შესვლისას მისი პირველი პარამეტრის შემცველობაში იქნება სამი სახელიდან (ნომრიდან) ერთ-ერთი. ამ ნომრის მიხედვით მოხდება შესაბამისი მრავალარხიანი მოწყობილობის დაკავება და განთავისუფლება. **ENTER** და **LEAVE** ბლოკებში **B** ოპერანდი არ გამოიყენება, ამიტომ ტრანზაქტებით დაკავებული იქნება და განთავისუფლდება მრავალარხიანი მოწყობილობიდან მხოლოდ ერთი (ერთი ერთეულის ტევადობის მრავალარხიანი მოწყობილობა).

IV თავი

იმიტაციურ მოდელებში ფუნქციის გამოყენება

4.1. ფსევდოშემთხვევითი რიცხვების გენერატორი

მოდელებში „შემთხვევითობის“ წყაროს წარმოადგენს ერთი ან რამდენიმე ფუნქცია, რომელთან, [0, 1] ინტერვალში თანაბარალბათური განაწილების ამონარჩევიდან, მიმართვის შედეგად შეიძლება მივიღოთ შემთხვევითი მნიშვნელობები. მნიშვნელობების გათამაშება თანაბრალბათური განაწილებისას ჩვეულებრივ შედგება ორი ეტაპისაგან:

- [0, 1] ინტერვალში თანაბარალბათური განაწილების ამონარჩევიდან გათამაშდება შემთხვევითი სიდიდე;
- მიღებული რიცხვი რაღაც სახით გარდაიქმნება მის ექვივალენტურ მნიშვნელობაში, რომელიც აუცილებელია ჩვენთვის საჭირო ამოცანებში გამოსაყენებლად.

GPSS World-ში [0, 1] ინტერვალში ფსევდოშემთხვევითი რიცხვების თანაბარალბათური განაწილებისას შესაძლებელია არსებობდეს გენერატორების ნებისმიერი რიცხვი. გენერატორთან მიმართვა წინასწარი აღწერის (გამოცხადების) გარეშე ხორციელდება **RNn** სახელით, სადაც **n** არის გენერატორის ნომერი.

შემთხვევითი რიცხვების გენერატორის საწყისი მნიშვნელობა ემთხვევა **n** გენერატორის ნომერს. მაგალითად, **RN2127.** ნომერი 2127 გენერატორი გაიშვება 2127 საწყისი რიცხვით.

მოდელირების დროს შესაძლოა წარმოიშვას იმის აუცილებლობა, რომ კვლევას დაექვემდებაროს:

- მოდელის სხვადასხვა ვარიანტები შემთხვევითი რიცხვების ერთსა და იმავე შემავალი ნაკადის შემთხვევაში;
- მოდელის ერთი ან რამდენიმე ვარიანტი შემთხვევითი რიცხვების სხვადასხვა ნაკადების დროს.

პირველი საჭიროება ადვილად რეალიზდება, ვინაიდან გენერატორები საშუალებას იძლევა აღქმულ იქნას თანაბრად განაწილებული შემთხვევითი რიცხვების თანამიმდევრობები.

მეორე შემთხვევაში **GPSS** იძლევა იმის შესაძლებლობას რათა შეცვლილ იქნას პირველი შვიდი **RN1…RN7** გენერატორის საწყისი რიცხვები და შესაბამისად ფორმირებას დაექვემდებაროს სხვადასხვა თანამიმდევრობები. ამისათვის **GPSS**-ში არსებობს შემდეგი ფორმატის **RMULT** ბრძანება:

RMULT [A], [B], [C], [D], [E], [F], [G]

სადაც **A – RN1**-ის საწყისი მნიშვნელობაა; **B – RN2**-თვის, **C – RN3**-თვის, **D – RN4**-თვის, **E – RN5**-თვის, **F – RN6**-თვის, **G – RN7**-თვის. მაგალითად:

RMULT , 112, 17, , ,295

ამ ჩანაწერის საფუძველზე დგინდება საწყისი რიცხვები მეორე, მესამე და მეექვსე გენერატორებისათვის. დანარჩენი მნიშვნელობები კი უცვლელი რჩება.

შემთხვევითი რიცხვების **RN** გენერატორმა შესაძლებელია მიიღოს მთელი მნიშვნელობები 0-დან 999-ის ჩათვლით, ხოლო შემთხვევითი ფუნქციების გამოთვლისას გამოიყენება 0-დან 0, 999999-ის ჩათვლით ინტერვალიდან არჩეული შემთხვევითი რიცხვი.

როგორც ზემოთ **GENERATE** ბლოკის განხილვისას აღვნიშნეთ, შემთხვევითი რიცხვების ყველა გენერატორს გააჩნია მასთან დაკავშირებული რიცხვთა სიმრავლე. ნებისმიერი გენერატორის მნიშვნელობა იცლება გენერაციის პროცედურის შესასრულებლად მასთან ყოველი მიმართვისას.

თუ არ არის მითითებული რაიმე დამატებითი მონაცემები, მაშინ ნებისმიერი გენერატორის საწყის მნიშვნელობად ითვლება 1.

4.2. ფუნქციათა ტიპები

GPSS-ში ფუნქცია შესაძლებელია მოცემულ უნდა იქნას ცხრილის სახით. ცხრილის მიხედვით ფუნქციის მნიშვნელობის მოძებნის წესი განისაზღვრება ფუნქციით **FUNCTION,** რომლის შემდეგაც მოდის არგუმენტისა და ფუნქციის მნიშვნელობები.

Name FUNCTION A, B

სადაც **Name** ფუნქციის სახელია.

A ოპერანდი – ფუნქციის არგუმენტია, რომელიც შესაძლებელია იყოს სახელი; მთელი დადებითი რიცხვი; სტრიქონი; გამოსახულება, რომელიც ჩაწერილია ბრჭყალებში; სისტემური რიცხვითი ატრიბუტები და სისტემური რიცხვითი ატრიბუტების პარამეტრი.

B ოპერანდი – შედგება ერთი ასოსაგან, რომელიც განსაზღვრავს ფუნქციის ტიპს და მთელი დადებითი რიცხვისაგან, რომელიც იძლევა არგუმენტისა და ფუნქციის შესაძლო მნიშვნელობების წყვილთა რაოდენობას (ფუნქციის წერტილების რაოდენობა).

GPSS World-ში გამოიყენება შემდეგი ხუთი ტიპის ფუნქცია:

- **D** – დისკრეტული რიცხვითი;
- **C** – უწყვეტი რიცხვითი;
- **E** – დისკრეტული ატრიბუტული;
- **L** – სიითი რიცხვითი;
- **M** – სიითი ატრიბუტული.

არსებობს ფუნქციის კიდევ ერთი ტიპი, რომელიც **B** ოპერანდის სახით გამოიყენება **GENERATE** და **ADVANCE** ბლოკებში. მას ეწოდება მოდიფიკატორი-ფუნქცია. ამ ფუნქციის მნიშვნელობა გამოითვლება ორმაგი სიზუსტით და მრავლდება **A** ოპერანდის მნიშვნელობაზე. მიღებული შედეგი გამოიყენება, როგორც მოცემულ ბლოკში საჭირო დროითი შეფერხება.

არათანაბარი განაწილებისას ოპერანდების რანგში გამოიყენება ფუნქცია. ამისათვის კი საჭიროა:

1. ფუნქციის განსაზღვრა არათანაბარი განაწილებისას;
2. **GENERATE** და **ADVANCE** ბლოკებში **A** და **B** ოპერანდები, რომლებიც აღწერენ თანაბარ განაწილებას შეცვლილია შესაბამისი ფუნქციებით.

4.2.1. GENERATE და ADVANCE ბლოკებში დისკრეტული ფუნქციის გამოყენება

ტრანზაქტების განაწილების ინტერვალი დისკრეტული არათანაბარი განაწილებისას **GENERATE** ბლოკში არ წარმოადგენს თანაბარს. ტრანზაქტების **GENERATE** ბლოკში შესვლისათვის აუცილებელია შემდეგი ორი მოქმედების შესრულება:

1. ფუნქციის განსაზღვრა, რომელიც აღწერს დროის ინტერვალების შესაბამის განაწილებას.
2. **GENERATE** ბლოკში **A** ოპერანდის რანგში გამოიყენება ფუნქცია, ხოლო **B** ოპერანდის მნიშვნელობა ნულის ტოლია.

4.2.2. ფსევდოშემთხვევითი რიცხვების გათამაშების მეთოდები დისკრეტული არათანაბარალბათური განაწილებისას

განაცხადების ან სხვა სიდიდეების შემოსვლისა და მომსახურების ინტერვალების განსაზღვრისას ხშირად აუცილებელი ხდება არათანაბარი განაწილებისას შემთხვევითი რიცხვების გათამაშება. განვიხილოთ ამგვარი რიცხვების გათამაშების მეთოდები დისკრეტული განაწილებისას.

ფუნქცია, რომელიც უნდა იქნეს ჩვენს მიერ გამოყენებული მოცემულია 4.1. ცხრილში. აღნიშნული ცხრილის თანახმად შემთხვევითმა სიდიდემ უნდა მიიღოს მნიშვნელობები 2, 5, 8, 9, 12 სიხშირით შესაბამისად 0, 15; 0, 2; 0, 25; 0, 22; 0, 18. ფუნქციის გრაფიკული ინტერპრეტაცია წარმოდგენილია 4.1 ნახაზის სახით.

[0, 1] ინტერვალში ხდება რიცხვის გათამაშება. დავუშვათ აღნიშნული რიცხვია 0, 47547. მაშინ 4.1 ცხრილის თანახმად ჩვენს მიერ ზემოხსენებული რიცხვი ჩავარდება მესამე ინტერვალში. შემთხვევითი სიდიდის საძიებელი მნიშვნელობა 8-ის ტოლია.

ამრიგად, როდესაც ადგილი აქვს შემთხვევითი რიცხვის გათამაშებას დისკრეტული განაწილებისას მოდელის შემქმნელმა ავტომატურად უნდა მოიძიოს ცხრილი რომლითაც მოცემულია აღნიშნული ფუნქცია. მოცემული შემთხვევისათვის კი ამისათვის საჭიროა:

ცხრილი 4.1

ფუნქციის დავალება

შემთხვევითი სიდიდის მნიშვნელობა	ფარდობითი სიხშირე	ჯამური სიხშირე	დიაპაზონი	ინტერვალი
2	0,15	0,15	0,0...0,15	1
5	0,5	0,35	0,15...0,35	2
8	0,25	0,6	0,35...0,6	3
9	0,22	0,82	0,6...0,82	4
12	0,18	1,0	0,82...1,0	5

შემთხვევითი რიცხვების წყარო [0, 1] ინტერვალში;

შემთხვევითი სიდიდის მნიშვნელობა;

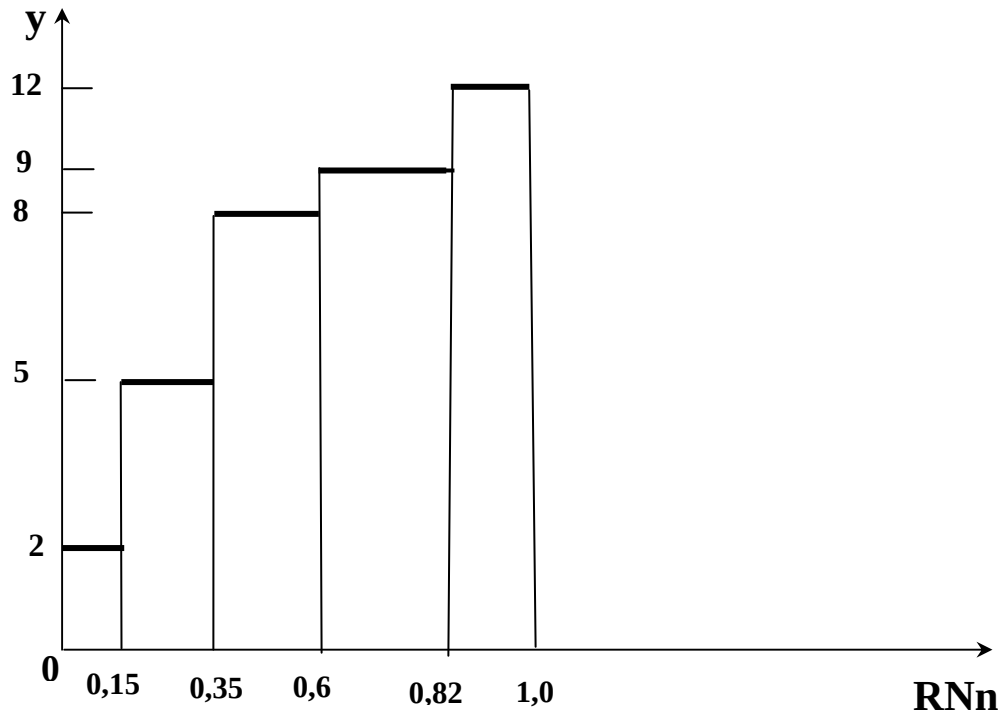
შემთხვევითი სიდიდის მნიშვნელობის მისაღებად ჯამური სიხშირე.

აღნიშნული მონაცემები მომხმარებელს მიეწოდება **D** – დისკრეტული ტიპის ფუნქციის დახმარებით. ამ ცხრილში **D** ტიპის მნიშვნელობის **x** არგუმენტი შეიძლება იყოს **Expression** ტიპის, **y** ფუნქციის მნიშვნელობა – **Integer, Real** ან **Name** ტიპის. ჩვენი მაგალითისათვის 4.1 ცხრილში მოცემული ფუნქცია განსაზღვრულია **FUNCTION** ბრძანებით შემდეგნაირად:

GHUMELI FUNCTION RN1, D5
.15, 2/ . 35, 5/ . 6, 8 /. 82, 9/1, 12

ფუნქციის სახელწოდებაა **GHUMELI.** შემთხვევითი რიცხვების წყაროს წარმოადგენს **RN1** გენერატორი. შემთხვევით სიდიდეს შეიძლება გააჩნდეს 5 სხვადასხვა მნიშვნელობა. ჯამური სიხშირეები და მისი შესაბამისი ფუნქციის მნიშვნელობები ჩაიწერება როგორც 5 წყვილი რიცხვი.

x და **y** წერტილების კოორდინატების განსაზღვრულ რიცხვთა წყვილის დაწერისას ფუნქცია შესაძლებელია იყოს არა მარტო დისკრეტული, არამედ ნებისმიერი ტიპის, რომლისათვისაც აუცილებელია:



ნახ. 4.1. დისკრეტული ფუნქციის გრაფიკული ინტერპრეტაცია

- ფუნქციის ერთი წერტილის **x** და **y** კოორდინატების მნიშვნელობათა მძიმით გამოყოფა;
- კოორდინატების წყვილები გამოიყოფა / ნიშნით ან სტრიქონის გადატანით;
- **x**-ის ყოველი მომდევნო მნიშვნელობა უნდა იყოს მის წინა მნიშვნელობაზე მეტი;
- ყოველ ფუნქციას უნდა გააჩნდეს სულ მცირე ორი აღწერილი წერტილი მაინც;
- **FUNCTION** ბრძანების **B** ოპერანდის მონაცემთა წყვილების რაოდენობა უნდა დაემთხვეს ფუნქციის მონაცემთა სიაში / სიმბოლოთი გამოყოფილ წყვილთა რაოდენობას;
- ფუნქციის მონაცემთა სიებს არ გააჩნიათ კომენტარის ველები.

როგორც ცნობილია, **TRANSFER** ბლოკს გააჩნია მუშაობის ისეთი რეჟიმები როგორიცაა ამორჩევითი (**PICK**) და ფუნქციონალური (**FN**) რეჟიმები. **PICK** რეჟიმში ბლოკის ამორჩევა, რომლისაკენაც მიმართულია აქტიური ტრანზაქტი მიმდინარეობს შემთხვევით. ბლოკები განლაგებული უნდა იყოს ერთმანეთის თანამიმდევრობით და გადასვლა თითოეულ მათგანში უნდა განხორციელდეს თანაბარალბათურად. როგორ მოვიქცეთ იმ შემთხვევაში როდესაც ბლოკები არ არის განლაგებული ერთმანეთის თანამიმდევრობით და თითოეული მათგანის არჩევა არ არის თანაბარალბათური? ამ შემთხვევაში გამოყენებული უნდა იქნეს ფუნქცია და ბლოკი **TRANSFER FN** რეჟიმში. მოდელში ეს შემთხვევა აღიწერება შემდეგნაირად:

Per FUNCTION RN13,D4

0 . 2, Met1 / 0 . 5, Met2 / 0. 76 , Met3 / 1, Met 4

GENERATE (Exponential (305, 0, 17, 3))

. . .

TRANSFER FN, Per

Met 1 . . .

Met 2 . . .

Met 3 . . .

Met 4 . . .

GENERATE ბლოკი ახდენს ტრანზაქტების გენერირებას. **TRANSFER** ბლოკში ტრანზაქტის შესვლის შემთხვევაში გამოითვლება ფუნქცია, რომლის არგუმენტს წარმოადგენს **RN13.** აირჩევა ფუნქციის მნიშვნელობა. ჩვენი მაგალითის შემთხვევაში

ფუნქციის მნიშვნელობას წარმოადგენს ჭდეების ბლოკები. არჩეული შესაბამისი ჭდის თანახმად წარმოებს აქტიური ტრანზაქტის უპირობო გადასვლა ბლოკში, რომელიც მონიშნულია აღნიშნული ჭდით. **GENERATE** ბლოკში **D** ტიპის ფუნქცია შესაძლოა გამოვიყენოთ დისკრეტული წარმოსახვითი ინტერვალებით ტრანზაქტების გენერირებისათვის. მაგალითად:

```
DInt    FUNCTION    RN831,D4
15, 3.2/ .55,4.7/ . 75,7.1/1, 1. 5
GENERATE    FN$DInt
TERMINATE   1
```

GENERATE ბლოკი 3,2; 4,7; 7,1 და 1,5 ინტერვალებით ახდენს ტრანზაქტების გენერირებას, რომელთა გამოჩენის ალბათობა შეადგენს შესაბამისად 0,15, 0,4 0,2 და 0,25. აღნიშნულში შესაძლებელია დავრწმუნდეთ მოყვანილი მაგალითის საფუძველზე **START** ბრძანებაში 1, 2, 3, 4-ის ტოლი **A** ოპერანდის ჩასმით იმავდროულად გავაანალიზებთ რა მოდელირების დროის დასრულებას.

სწორედ ასევე **D** ტიპის ფუნქცია შესაძლებელია გამოყენებულ იქნას **ADVANCE** ბლოკში. ხშირად **ADVANCE** ბლოკით ერთსა და იმავე ერთარხიან ან მრავალარხიან სისტემებზე აუცილებელია ორგანიზებულ იქნას განსხვავებული დროითი მახასიათებლების მქონე განაცხადების მომსახურება. ამ შემთხვევაში ფუნქციის არგუმენტად გვევლინება ტრანზაქტის რომელიმე პარამეტრი. მაგალითად:

```
  Obs        FUNCTION      P1,D4
1, 3. 75 / 2, 4. 3 / 3, 2. 1 / 4, 5. 6
GENERATE      FN$DInt
TERMINATE     1
. . . . . .
ADVANCE FN$Obs
. . . . . . .
```

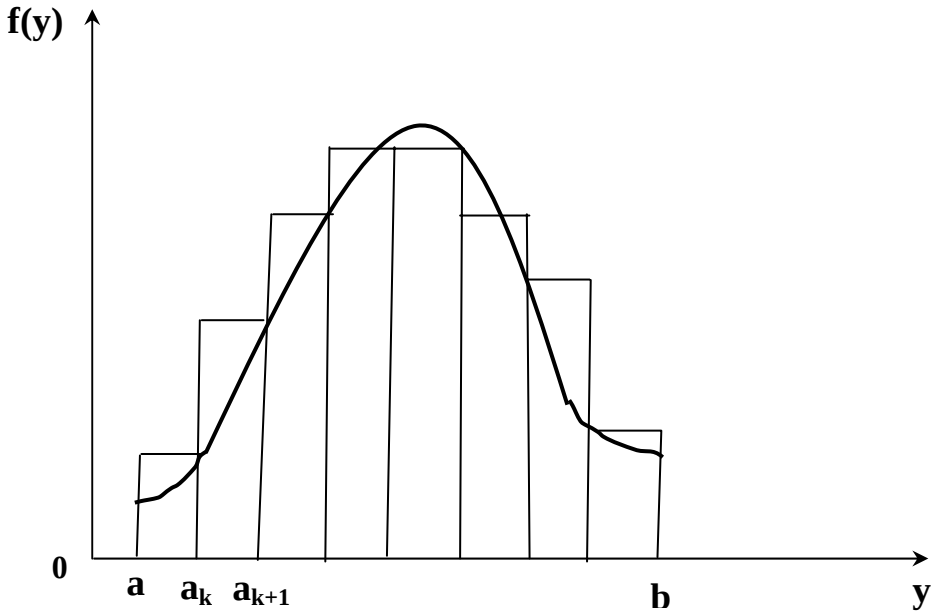
ADVANCE ბლოკში აქტიური ტრანზაქტის შესვლისას გამოითვლება **Obs** სახელით ცნობილი ფუნქცია, რომლის არგუმენტსაც წარმოადგენს **P1** პარამეტრი. მიღებული შედეგი კი გამოიყენება როგორც დაყოვნების დრო.

4.2.3. ფსევდოშემთხვევითი რიცხვების გათამაშების მეთოდები უწყვეტი არათანაბარი განაწილებისას

ზოგიერთ შემთხვევაში წარმოიქმნება რომელიმე მრუდის მიერ იმთავითვე მოცემული რომელიღაც უწყვეტი განაწილების თანახმად განსაზღვრული ფსევდოშემთხვევით რიცხვთა გათამაშების აუცილებლობა.

ამგვარ რიცხვთა გენერაციის მიზნით გამოიყენება განაწილების სიმჭიდროვის ნაწილობრივი აპროქსიმაციის მეთოდი.

დავუშვათ სამოდელირებელი შემთხვევითი სიდიდე **[a, b]** ინტერვალში მოცემულია **f(Y)** განაწილების სიმჭიდროვით. მოვახდინოთ **f(Y)** ფუნქციის აპროქსიმირება (ნახ.4.2).



ნახ. 4.2. სიმჭიდროვის განაწილების ნაწილობრივი აპროქსიმაცია

[a, b] ინტერვალის ნაწილებად დაყოფა უნდა მოხდეს ისე, რომ **Y** შემთხვევითი სიდიდის **[a_k, a_{k+1}]** ნებისმიერ ნაწილზე მოხვედრის ალბათობა იყოს მუდმივი, ანუ, არ იქნას დამოკიდებული განლაგებაზე, რომელიც გადის **[a, b]** ინტერვალზე. თუკი ნაწილების რიცხვი არის **m**, ამ შემთხვევაში, **[a_k, a_{k+1}]** ნებისმიერ ნაწილისათვის უნდა შესრულდეს ტოლობა

$$\int_{a_i}^{a_{i+1}} f(Y)\,dY = \frac{1}{m}, \qquad (4.1)$$

რომლის საფუძველზედაც **[a_k, a_{k+1}]** ნაწილზე შესაძლებელი იქნება განსაზღვრულ იქნას ფუნქციის აპროქსიმმირებელი მნიშვნელობა.

Y შემთხვევითი სიდიდის ერთი მნიშვნელობის მოდელირების პროცედურა მოიცავს შემდეგ ეტაპებს:

1. [0, 1] ინტერვალში თანაბარი განაწილების მქონე შემთხვევითი სიდიდის ფორმირება.

2. მნიშვნელობის მიხედვით **[a, b]** ინტერვალზე მისთვის შესაბამისი $[a_k, a_{k+1}]$ ნაწილის შერჩევა.

ასე, თუკი **[a, b]** შეიცავს 10 ნაწილს, ამ შემთხვევაში $0 \le Xi \le a_1$ მნიშვნელობებს შეესაბამება პირველი ნაწილი, $0 \le Xi \le a_2$ მნიშვნელობებს – მეორე ნაწილი და ა.შ.

2. მოცემული განაწილების მქონე შემთხვევითი სიდიდის მნიშვნელობათა გამოთვლა:

$$Y_i = a_k + (a_{k+1} - a_k) \cdot X_{i+1} \quad , \quad (4.2)$$

ექსპონენციალურ და ნორმალურ განაწილებათა მოდელირებისათვის განვიხილოთ ნაწილობრივი აპროქსიმაციის მეთოდის გამოყენების შესაძლებლობა.

4.2.3.1. ექსპონენციალური განაწილების მოდელირება

როდესაც ადგილი აქვს პუასონის ნაკადს (უბრალო ნაკადი, რომელიც აკმაყოფილებს ორდინარულობისა და სტაციონარულობისადმი წაყენებულ მოთხოვნებს), განაცხადების შემოსვლა აღიწერება პუასონის კანონით

$$P_k(T) = \frac{(\lambda T)^k}{K!} e^{-\lambda T}, \quad K = 0,1...., \qquad (4.3)$$

სადაც $\mathbf{P_k(T)}$ არის იმის ალბათობა, რომ **K** განაცხადი შევა **T** დროში;

λ – განაცხადების შემოსვლის საშუალო ინტენსიურობა.

პუასონის ნაკადის მოდელირების დროს არ წარმოიქმნება $\mathbf{P_k(T)}$ ალბათობათა განსაზღვრის აუცილებლობა. სავსებით საკმარისია გაგვაჩნდეს მოთხოვნათა შემოსვლის დროის ინტერვალების მნიშვნელობები. **GENERATE** ბლოკში შემდეგი ტრანზაქტის შემოსვლის მნიშვნელობის ანგარიში წარმოებს როგორც სისტემური დროის მიმდინარე მნიშვნელობების, ასევე განაწილების მიხედვით გათამაშებული დროის ინტერვალების დალაგების სიდიდის მეშვეობით. აღნიშნული განტოლება შეიძლება გამოსახულ იქნას დროსთან მიმართებაში. საბოლოოდ ყოველივე ამის შედეგს წარმოადგენს ექსპონენციალური განაწილება.

ექსპონენციალური განაწილება შებრუნებული ფუნქციის მეთოდით შეიძლება მოცემულ იქნას ისეთი ფორმით, რომ თუ გაგვაჩნია [0, 1] ინტერვალში თანაბრად განაწილებული რიცხვები, ამ შემთხვევაში შესაბამისი დროის ინტერვალები შესაძლოა გამოთვლილ იქნას უშუალოდ გამოსახულებიდან

$$T_{\text{გათ}} = T_{\text{საშ}}[-LN(1-RNn)], \qquad (4.4)$$

სადაც $\mathbf{T_{გათ}}$ – არის დროის გათამაშებული ინტერვალი;

$\mathbf{T_{საშ}}$ – დროის საშუალო ინტერვალი;

RNn – თანაბრადგანაწილებულ შემთხვევით სიდიდეთა გენერატორის სახელი.

აღნიშნული განტოლების გამოყენების დროს **GENERATE** ბლოკში შემავალ პუასონის ნაკადის მოდელირებისათვის აუცილებელია:

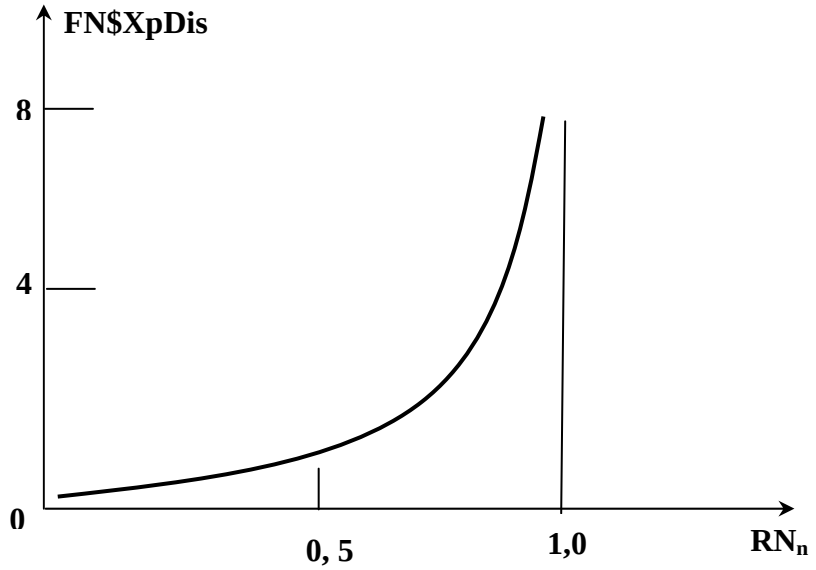
RNn-იდან მივიღოთ რიცხვი და ავიღოთ (**1 - RNn**) მინუს ნიშნის მქონე სიდიდის ლოგარითმი, ხოლო შემდეგ მიღებული შედეგი გავამრავლოთ $\mathbf{T_{საშ}}$-ზე. პირველი ეტაპი შეიძლება შევასრულოთ **GPSS**-ში ფუნქციის შეტანის გზით, რომელიც არგუმენტის სახით **RNn**-ს ეკუთვნის და იღებს [**-LN(1 - RNn)**] მნიშვნელობას. **GPSS**-ში ლოგარითმის ფუნქცია შეიძლება განვსაზღვროთ შემდეგი წესით. სახეობის [**-LN(1 - RNn)**] განაწილება უნდა აპროქსიმმირდეს ჭეშმარიტ ფუნქციაზე დართული ნაწილების (ნაკვეთების) სახით.

GPSS-სათვის ამგვარი აპროქსიმაცია განხორციელებულია 23 ნაწილით, რომლებიც მოიცავს შემთხვევითი ცვლადის ექსპონენციალური ფუნქციის მნიშვნელობას [0, 8] ინტერვალში. ამგვარი ფუნქციის განსაზღვრა, რომლის სახელწოდებაც გახლავთ **XpDis (EXPONENTIAL DISTRIBUTION)** შედგება 24 წყვილ მნიშვნელობათაგან, რომლებიც გამოიყენება **RNn** მნიშვნელობათა გარდასაქმნელად **[-LN(1 - RNn)]** მნიშვნელობად.

XPDIS FUNCTION RN1,C24; ექსპონენციალური განაწილების ფუნქცია

```
0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915
.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.3
.92,2.52/.94,2.81/.95,2.99/.96,3.2/.97,3.5/.98,3.9
.99,4.6/.995,5.3/.998,6.2/.999,7/.9998,8
```

[**-LN(1 - RNn)**] გამოსახულების თეორიული მნიშვნელობები შეიძლება მდებარეობდეს 0-დან ∞ რიცხვთა დიაპაზონში. პრაქტიკულად, ვინაიდან **RNn**-ის ყველაზე დიდ მნიშვნელობას წარმოადგენს 0, 999999, გამოსახულების გაცილებით უფრო დიდ მნიშვნელობად გვევლინება 13, 8155. იმასთან დაკავშირებით, რომ ფუნქციის ბოლო მნიშვნელობას წარმოადგენს წყვილი .9998, 8, ფუნქციას თავისი მნიშვნელობის სახით გამოაქვს სიდიდე 8 და არა 13, 8155 (ნახ. 4.3).



ნახ. 4.3. **FN$XpDis** ფუნქციის გრაფიკული ინტერპრეტაცია

თუმცა, **RNn** იღებს 0,9998-ზე დიდ მნიშვნელობას მხოლოდ 0, 0199%-ის შემთხვევაში. უმრავლეს შემთხვევაში ამგვარი აპროქსიმაცია წარმოადგენს სრულებით საკმარისს.

მეორე ეტაპზე ფუნქციის მნიშვნელობა უნდა გავამრავლოთ $\mathbf{T}_{საშ}$-ზე იმისათვის, რომ მივიღოთ პუასონის ნაკადის შესაბამისი დროის ინტერვალის მნიშვნელობა.

აღნიშნულის ავტომატურად შესასრულებლად **GENERATE** ბლოკს გააჩნია სპეციალური თვისება. პუასონის ნაკადი მოდელირდება **GENERATE** ბლოკის მეშვეობით შემდეგ მოქმედებათა ეტაპობრივი შესრულების გზით:

- **A** ოპერანდის სახით გამოიყენება დროის ინტერვალთა მნიშვნელობა;
- **B** ოპერანდის სახით გამოიყენება **FN$XpDis** ჩანაწერი.

დროის ინტერვალის მნიშვნელობა გამოითვლება ბლოკში ფუნქციის მნიშვნელობის **A** ოპერანდის მნიშვნელობაზე გამრავლების გზით.

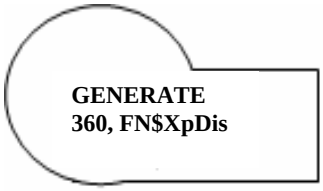
განვიხილოთ ექსპონენციალური განაწილების მაგალითების გამოყენება **GPSS**-ში.

მაგალითი 4.1. დავუშვათ, რომ აუცილებელია მოვახდინოთ პუასონის ნაკადის მოდელირება 24 საათის განმავლობაში 4-ის ტოლი ინტენსიურობის საშუალო მნიშვნელობით. დროის ერთეული – 1 წუთი. როდესაც ტრანზაქტი შედის ბლოკში და გადაინაცვლებს შემდეგი ბლოკისკენ. შემდეგ ბლოკში ტრანზაქტის შესვლის დრო გამოითვლება **FN$XpDis** ფუნქციის შედეგის 360-ზე გამრავლებისა და წამმზომის მიმდინარე დროზე მისი დამატების გზით. **FN$XpDis** ფუნქციის გამოსათვლელად აუცილებელია განსაზღვრულ იქნას **RN23 = 0,35, FN$XpDis = 0,432**

(საშუალო 0, 355 და 0, 509 შორის) მნიშვნელობა. წარმოებული უდრის 155,52. დროის ინტერვალი – 155,52.

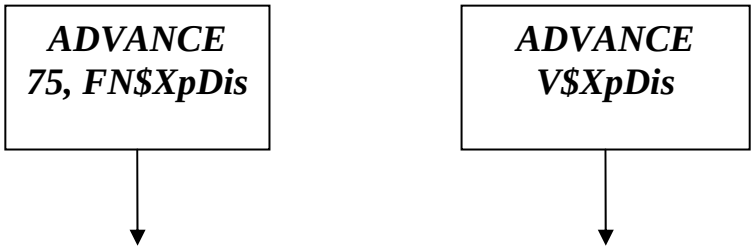
ცვლადის გამოყენების დროს, მაგალითად:

Fun FVARIABLE 360#FN$XpDis



ბლოკები **GENERATE 360, FN$XpDis** და **GENERATE V$Fun** ექვივალენტურია.

მაგალითი 4.2. ყოველივე, რაზეც ზემოთ იყო ნახსენები, რომელიც უშუალოდ შეეხებოდა **GENERATE** ბლოკის მიერ პუასონის ნაკადის მოდელირებას ძალაში რჩება **ADVANCE** ბლოკში დროის დაყოვნების ექსპონენციალური მოდელირებისთვისაც. მაგალითად, თუკი ერთარხიანი მოწყობილობა ექსპონენციალური მომსახურებით ერთ მომსახურებაზე საშუალოდ მოითხოვს სამოდელო დროის 75 ერთეულს, იგი მოდელის სახით შეიძლება გამოსახულ იქნას **ADVANCE**-ის შემდეგი ბლოკებით:



4.2.3.2 ნორმალური განაწილების მოდელირება

ნორმალური განაწილების მქონე შემთხვევითი სიდიდე სრულად აღიწერება ორი მნიშვნელობის მინიჭებით: მათემატიკური ლოდინით და სტანდარტული გადახრით. ნორმალური განაწილების მქონე ნორმირებულ შემთხვევით სიდიდეს გააჩნია ნულის ტოლი მათემატიკური ლოდინი და ერთის ტოლი სტანდარტული გადახრა. როგორც წესი, შემთხვევითი სიდიდეები, რომლებიც წარმოადგენს ინტერესს არის არანორმირებული. იმისათვის, რომ არანორმირებული ნორმალური განაწილებიდან მოვახდინოთ ამონარჩევი, თავდაპირველად უმჯობესია ანალოგიური ქმედება განვახორციელოთ ნორმირებული ნორმალური განაწილებიდან, ხოლო ამის შემდეგ მიღებულ სიდიდეზე ვაწარმოოთ არითმეტიკული მოქმედებები გარდავქმნით რა მას არანორმირებულ ნორმალურ განაწილებად.

$$GNorm_{\text{ამონ}} = (GNorm_{\text{სტგ}})(SNorm_{\text{ამონ}}) + GNorm_{\text{მთლ}}, \qquad (4.5)$$

სადაც **SNorm**$_{\text{ამონ}}$ – არის ნორმირებული ნორმალური განაწილებიდან შერჩეული სიდიდე;

GNorm$_{\text{მთლ}}$ და **GNorm**$_{\text{სტგ}}$ – არანორმირებული ნორმალური განაწილების შესაბამისი მათემატიკური ლოდინი და სტანდარტული გადახრა.

ნორმირებული ნორმალური განაწილებისათვის შემთხვევითი სიდიდე შესაძლოა ნაპოვნი იქნას შემდეგი გამოსახულებიდან

$$RNn = \int_{-\infty}^{SNorm_{\text{ამონ}}} \frac{e^{\frac{x}{2}}}{\sqrt{2\pi}} dX \quad , \qquad (4.6)$$

სადაც **RNn** – არის **GPSS**-ის თანაბარალბათური განაწილებიდან ნებისმიერის შემთხვევითი სიდიდე 0-დან 1-მდე;

SNorm$_{\text{ამონ}}$ – ინტეგრირების ზედა ზღვარი.

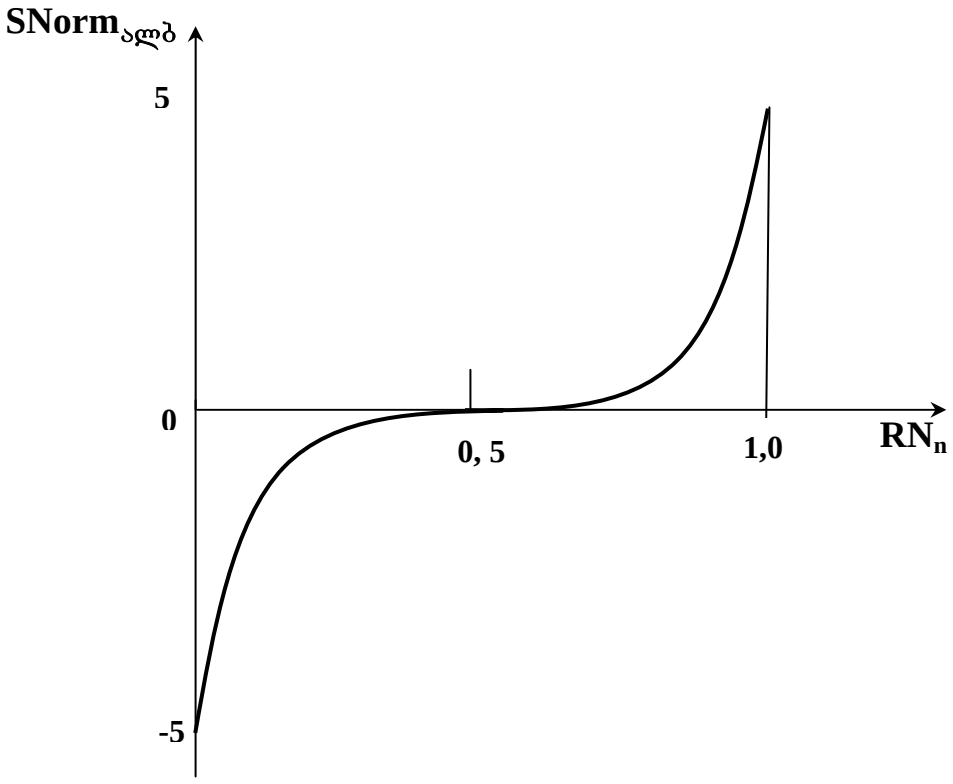
RNn – სათვის **SNorm**$_{\text{ამონ}}$ ზუსტი მნიშვნელობის მოსაძებნად ანალიზურად პრაქტიკულად შეუძლებელია განტოლების ამოხსნა. როგორც შედეგი, **RNn** – სათვის განტოლებიდან განსაზღვრული **SNorm**$_{\text{ამონ}}$ სიდიდეები წარმოდგენილ უნდა იქნას ცხრილის სახით.

4.4 ნახაზზე ნაჩვენებია დამოკიდებულება **RNn** – სა და **SNorm**$_{\text{ამონ}}$ შორის.

აღნიშნული მრუდი აპროქსიმმირდება **GPSS**-ის უწყვეტი ფუნქციის მეშვეობით, რომელიც თავის მხრივ შედგენილია მისთვის შესაბამისი 24 სწორხაზოვანი (წრფივი) სეგმენტის მიერ. **SNorm**-ის სახელწოდების მქონე ფუნქციის განსაზღვრა შედგება 25 წყვილ მნიშვნელობათაგან:

```
SNORM  FUNCTION RN123, C25    ;ნორმალური განაწილების ფუნქცია

0,-5/.00003,-4/.00135,-3/.00621,-2.5/.02275,-2
.06681,-1.5/.11507,-1.2/.15866,-1/.21186,-.8
.27425,-.6/.34458,-.4/.42074,-2 /.5,.0/.57926,.2
.65542,.4/.72575,.6/.78814,.8/.84134,1/.88493,1.2
.93319,1.5/.97725,2/.99379,2.5/.99865,3/.99997,4/1,5
```



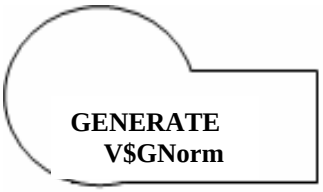
ნახ. 4.4. **SNorm** ამონ ფუნქციის გრაფიკული ინტრპრეტაცია

აქვე შევნიშნოთ, რომ **SNorm** ამონ სიდიდეები შესაძლოა მდებარეობდეს -∞ +∞ საზღვრებში, **FN$SNorm** –სათვის აღებული სიდიდეები შეიცვლება მხოლოდ -5 +5 ინტერვალში. პრაქტიკისათვის ასეთი განშლადობა სავსებით დასაშვებია.

განვიხილოთ ნორმალური განაწილების გამოყენების მაგალითი **GPSS**-ში.

მაგალითი 1. შეტყობინებათა გადაცემაზე კავშირის არხის მიხედვით განაცხადების შემოსვლებს შორის დროის ინტერვალი განაწილებულია შემთხვევითი სიდიდეების განაწილების ნორმალური კანონის მიხედვით მათემატიკური ლოდინით 24 და სტანდარტული გადახრით 2 სთ. დროის ერთეულად მიღებულია 1 სთ. **GENERATE** ბლოკი, რომელიც ახდენს განაცხადების

შემოსვლის პროცესის იმიტირებას შეიძლება წარმოდგენილ იქნას შემდეგი სახით. **GENERATE** ბლოკის **A** ოპერანდში გამოყენებული **GNorm** ცვლადი (იხ. სქემა) აღწერილია შემდეგნაირად:



GNorm FVARIABLE 2#FNSN_{orm}$+24

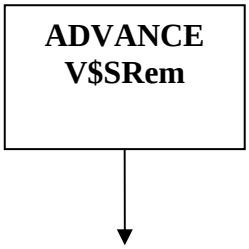
ამგვარად, ყველა იმ მომენტისათვის, როდესაც ტრანზაქტი გამოდის **GENERATE** ბლოკიდან, სისტემის შემქმნელი შემთხვევით განსაზვრავს მომდევნო ტრანზაქტის გამოსვლის დროის სიდიდეს, რომელიც აკმაყოფილებს შემთხვევითი სიდიდეების განაწილების ნორმალური კანონის მიხედვით განაწილებულ არანორმირებული შემთხვევითი სიდიდის განსაზღვრის წესს.

მაგალითი 2. მომსახურების დროს მოდელის რაღაც წერტილში გააჩნია ნორმალური განაწილება, ხოლო მათემატიკური ლოდინი და სტანდარტული გადახრა ნამდვილად დამოკიდებულია განსახორციელებელი მომსახურების სახისაგან. მაგალითად, კავშირის საშუალებაა განსხვავებული ტიპების რემონტი ერთსა და იმავე სარემონტო დაწესებულებაში ერთი და იგივე ოსტატების მიერ (ცხრილი 4.2)

ცხრილი 4.2

განაწილების კანონის წარმართვა მომსახურების სახისაგან დამოკიდებულებით

მომსახურების სახე	მათემატიკური ლოდინი	სტანდარტული გადახრა
1	30	5
2	20	3
3	36	6



დავუშვათ, რომ ტრანზაქტისათვის აუცილებელია მომსახურების სახე ჩაწერილია მის 4-ე პარამეტრში. მაშინ **ADVANCE** ბლოკი შეიძლება წარმოდგენილ იქნას ისე, როგორც ეს ნაჩვენებია მარჯვნივ. ბლოკის ფუნქციონირების უზრუნველმყოფი ფუნქციისა და არითმეტიკული ცვლადის განსაზღვრა შემდეგნაირად ჩაიწერება:

Mean FUNCTION P4,D3

1, 30 / 2, 20 / ,3, 36

StDe FUNCTION P4,D3

1, 5 / 2, 3 / , 3, 6

Srem FVARIABLE FN$StDe#FN$SNorm+FN$Mean

მოყვანილ მაგალითებში თავისი არგუმენტის სახით ფუნქციები გამოიყენებენ აქტიური ტრანზაქტის **P4** პარამეტრის მნიშვნელობას. **Srem** ფუნქციასთან მიმართვის დროს თავდაპირველად გამოითვლება **Srem, StDe** და **SNorm** ფუნქციები. ფუნქციის მიღებული მნიშვნელობები გამოიყენება **Srem** ფუნქციის გამოთვლის დროს.

შენიშვნა: ფუნქციის ყველაზე მცირე მნიშვნელობა 5-ის ტოლია. შეიძლება აღმოჩნდეს, რომ ამონარჩევის შესაბამისი მნიშვნელობა იქნეს უარყოფითი. თუმცა, პრაქტიკაში ცნობილია, რომ ზოგიერთ უარყოფით სიდიდეებს არ აქვს აზრი (მაგ. დრო). მოდელის შემქმნელი დარწმუნებული უნდა იყოს იმაში, რომ მათემატიკური ლოდინი თითქმის ყოველთვის 5-ჯერ მეტია სტანდარტულ გადახრაზე. წინააღმდეგ შემთხვევაში შეცდომის – „დაყოვნების უარყოფითი დრო“ თანახმად მოხდება პროგრამის შეჩერება.

4.2.4. E, L და M ტიპის ფუნქციები

***E** ტიპის ფუნქცია.* როგორც ცნობილია **D** ტიპის დისკრეტული რიცხვითი ფუნქციის **X** არგუმენტის მნიშვნელობა შეიძლება იყოს **Expression** ტიპის, ხოლო **Y** ფუნქციის მნიშვნელობები – **Integer, Real** ან **Name** ტიპის. **E** ტიპის დისკრეტულ ატრიბუტულ ფუნქციაში **X** არგუმენტი შეიძლება იყოს **Expression** ტიპის, ხოლო **Y – Integer, Real, Name,** სტრა ან ფრჩხილებში ჩასმული გამოსახულება. მაგალითად:

```
Archeva FUNCTION P3, E4
1, V$Norm1 / 7, V$Norm2 / 12, R$Usel / 14, fn$Sp
```

E ტიპის ფუნქცია ისევე გამოითვლება, როგორც **D** ტიპის ფუნქცია, მხოლოდ იმ განსხვავებით, რომ **X** არგუმენტის პოვნის შემდეგ გამოითვლება შესაბამისი სტრა ან ფრჩხილებში ჩასმული

გამოსახულება. მაგალითად, თუ **P = 13**, მაშინ გამოითვლება **Sp** სახელით მითითებული ფუნქცია.

L *ტიპის ფუნქცია.* **E** ტიპის ფუნქციაში **X** შეიძლება იყოს **Expression** ანუ გამოთვლის შემდეგ მიიღოს ნებისმიერი მნიშვნელობები. **L** ტიპის სიით რიცხვით ფუნქციაში **X** არგუმენტი შეიძლება იყოს მხოლოდ **Integer** ტიპის. ამიტომ არგუმენტი განიხილება, როგორც რიგითი ნომერი. ფუნქციის მნიშვნელობები შეიძლება იყოს **Integer, Real** ან **Name** ტიპის. მაგალითად:

Blok FUNCTION Q$Sigrdze, L5
1, Wde1/2, Wde2/3, Wde3/4, Wde4/5, Wde5

ფუნქციის მნიშვნელობა განისაზღვრება რიგითი ნომრის მიხედვით. თუ არგუმენტის გამოთვლისას მიიღება 1-ზე ნაკლები და მის მოცემულ მაქსიმალურ მნიშვნელობაზე მეტი შედეგი, მაშინ შესაბამისად წარმოიქმნება შემდეგი სახის შეცდომები: „ფუნქციის სიაში არის უარყოფითი არგუმენტი“ ან „სიითი ფუნქციის არგუმენტი არის ძალიან დიდი“. არგუმენტის მნიშვნელობები უნდა დაიწყოს 1-დან და თითოეულ შემდეგ მონაცემთა წყვილთათვის გაიზარდოს 1-ით. შეცდომა წარმოიქმნება იმ შემთხვევაშიც, როდესაც არგუმენტის მნიშვნელობა არ არის მთელი რიცხვი. მაგალითად, 1,5.

M *ტიპის ფუნქცია.* **M** ტიპის სიითი ატრიბუტული ფუნქცია **L** ტიპის რიცხვითი ფუნქციისაგან განსხვავდება იმით, რომ ფუნქციის მნიშვნელობა შეიძლება იყოს **Integer, Real, Name,** სტრა

ან ფრჩხილებში ჩასმული გამოსახულება. არგუმენტი წარმოადგენს მთელ რიცხვს და განიხილება როგორც რიგითი ნომერი. ამიტომ, იგი თითოეულ მონაცემთა წყვილთათვის შეიძლება დაიწყოს 1-დან და გაიზარდოს 1-ით. მაგალითად:

Posl FUNCTION X$Kartli, M5
1, Q$Stan / 2, N$Term3 / 3, S$Usel / 4, V$Zirk / 5, FN$Susp

M ტიპის ფუნქცია ისევე გამოითვლება, როგორც **L** ტიპის ფუნქცია, მხოლოდ არგუმენტის რიგითი ნომრის პოვნის შემდეგ გამოითვლება შესაბამისი სტრა ან ფრჩხილებში ჩასმული გამოსახულება.

მაგალითად, როდესაც **X = 4** გამოითვლება **Zirk** სახელით მითითებული არითმეტიკული ცვლადი. როდესაც **5<X<1** შესაბამისად წარმოიშობა შემდეგი სახის შეცდომები: „ფუნქციის სიაში არის უარყოფითი არგუმენტი“ და „სიითი ფუნქციის არგუმენტი არის ძალიან დიდი“.

V თავი
გაცილებით უფრო რთული მოდელების აგება

წინა თავში ჩვენ ირიბად შევეხეთ ოპერანდების მნიშვნელობების სპეციფიკაციას, სადაც ბლოკებში ოპერანდების ნაცვლად გამოიყენებოდა ფუნქცია, რომლის დროსაც იგულისხმებოდა, რომ ოპერანდების მნიშვნელობები წარმოადგენენ ფუნქციის მნიშვნელობებს. წინა თავში **GENERATE** და **ADVANCE** ბლოკებში **A** და **B** ოპერანდების მნიშვნელობათა განსაზღვრისას გამოიყენებოდა ფუნქცია.

წინამდებარე თავში შემოთავაზებულია **GPSS** ბლოკების ოპერანდების ირიბი სპეციფიკაციის დამატებითი მეთოდები. ოპერანდების ყველა მნიშვნელობების წყარო დაყოფილია ორ კატეგორიად: სისტემური ატრიბუტები და ტრანზაქტების ატრიბუტები.

სისტემური ატრიბუტები – ეს გახლავთ პარამეტრები, რომლებიც აღწერენ მოდელის მდგომარეობას. ისეთი რაოდენობრივი მაჩვენებლები, როგორიცაა „მე-5 რიგის მიმდინარე შემცველობა“ ან „დროის ერთეული, რომლის განმავლობაშიც დაკავებულია მოწყობილობა“ – წარმოადგენს ტიპიურ სისტემურ ატრიბუტებს.

ზემოაღნიშნულ ატრიბუტებს უწოდებენ სტანდარტულ რიცხვით ატრიბუტებს (სტრა). მათი მნიშვნელობები, რომლებიც ჩვეულებისამებრ იცვლება მოდელირების პროცესში ხელმისაწვდომია მომხმარებლის მიერ.

შესვლა ხორციელდება აღნიშნული ატრიბუტების სპეციალური დასახელებების გამოყენებისას. ოპერანდების სახით აღნიშნული დასახელებების გამოყენების დროს, ამ უკანასკნელთა მნიშვნელობებად გვევლინება ატრიბუტების შესაბამისი მიმდინარე მნიშვნელობები.

ამასთან, ტრანზაქტებს ასევე შეიძლება გააჩნდეთ რაღაც რიცხვითი მახასიათებლები. ერთ-ერთ მათგანს წარმოადგენს პრიორიტეტის დონე, რაზეც ჩვენ უკვე ვისაუბრეთ, მაგრამ ჯერ-ჯერობით არ გამოგვიყენებია ოპერანდის რანგში. გარდა ამისა, თითოეული ტრანზაქტი მარაგდება პარამეტრების რაიმე რიცხვით. ტრანზაქტებს შესაძლოა გააჩნდეს ასამდე სხვადასხვა პარამეტრი. პარამეტრებს გააჩნიათ მთელი მნიშვნელობები. ისინი შეიძლება დავაწესოთ და ვცვალოთ მოდელირების ლოგიკასთან შესაბამისობაში. პარამეტრების მნიშვნელობები ასევე შეიძლება გამოყენებულ იქნას მოდელის ბლოკების ოპერანდების მნიშვნელობათა სახით. დაბოლოს, ტრანზაქტები განისაზღვრება ისეთი მახასიათებლებით, როგორიც გახლავთ დროის აღნიშვნა. იგი შესაძლოა ინტერპრეტირებული იქნეს როგორც ტრანზაქტის „დაბადების“ ან მისი მოდელში შესვლის დრო.

5.1 სტანდარტული რიცხვითი ატრიბუტები

მოდელირების პროცესში **GPSS** ინტერპრეტატორი ავტომატურად არეგისტრირებს და აკორექტირებს მოდელში გამოყენებადი სხვადასხვა ელემენტების შესახებ დაგროვილ ზოგიერთ ინფორმაციას. აღნიშნული ინფორმაციის რაღაც ნაწილი იბეჭდება მოდელის გაშვების დასრულებისთანავე. ასეთი სახის ინფორმაციად გვევლინება ბლოკების მრიცხველები, ხელსაწყოებისა და მრავალარხიანი მოწყობილობების დატვირთვები, რიგებში ყოფნის საშუალო დრო და ა.შ. მოდელის გაშვების დასრულების დროს მიღებული ინფორმაციის გარდა არსებობს კიდევ ისეთი ინფორმაცია, რომელიც ხელმისაწვდომია მოდელირების პროცესში. თვით მოდელირების პროცესი შესაძლოა იმართებოდეს დინამიურად, გვერდიდან მოდელის ატრიბუტების უშუალო ზემოქმედების გზით. მაგალითად, ინტენსიურობა, რომლითაც მოწყობილობა ემსახურება მოთხოვნებს სინამდვილეში შესაძლოა დამოკიდებული იყოს მომსახურების მომლოდინე მოთხოვნათა რიცხვისაგან. რიგის სიგრძის გაზრდის დროს მოწყობილობამ შეიძლება იმუშაოს უფრო სწრაფად. რეალური სისტემის მოდელირებისათვის შესაძლოა აუცილებელი აღმოჩნდეს ის გარემოება, რომ მომსახურების დრო დამოკიდებული იქნას რიგის მიმდინარე სიგრძისაგან. თუკი ასეთი ატრიბუტი დასაშვებია სისტემაში, იგი შეიძლება გამოყენებულ იქნას მომსახურების დროის განსაზღვრისათვის მომსახურებაზე მოთხოვნის ყოველი შემოსვლის დროს.

5.1-5.3 ცხრილებში ნაჩვენებია სხვადასხვა რიცხვითი ატრიბუტები ხელსაწყოებისათვის, მრავალარხიან მოწყობილობათა და შესაბამისად რიგებისათვის. აღნიშნულ ატრიბუტებზე შეიძლება მითითებულ იქნას მოდელირების პროცესში. 5.1-5.3 ცხრილებში წარმოდგენილ ატრიბუტებს ეწოდებათ სტანდარტული რიცხვითი ატრიბუტები ან უბრალოდ სტრა. სტანდარტული რიცხვითი ატრიბუტის სახელი შედგება ორი ნაწილისაგან. პირველი ნაწილი მიუთითებს ჯგუფურ სახელს. იგი ერთდროულად ახდენს როგორც ელემენტის (ანუ ხელსაწყო, მრავალარხიანი მოწყობილობა ან რიგი), ასევე ინფორმაციის ტიპის (მაგალითად მოწყობილობის დაკავების მრიცხველი, მრავალარხიანი მოწყობილობის დატვირთვა, რიგის მიმდინარე მდგომარეობა) იდენტიფიცირებას. მეორე ნაწილი აწარმოებს ჯგუფის კონკრეტული წევრის (ანუ როგორი მოწყობილობა, როგორი რიგი) იდენტიფიცირებას. 5.1-5.3 ცხრილებში სხვადასხვა სტრა-ის ჯგუფების სახელები სვეტში „დასახელება“ ჩაწერილია მთავრული ასოებით. ჯგუფის სახელთან ერთად იწერება მოცემული ჯგუფის შესაბამისი წევრის რიცხვითი ან სიმბოლური სახელი. ცხრილში ნომერი წარმოდგენილია **j** სიმბოლოთი. სიმბოლური სახელი ჩაწერილია როგორც „სახელი“. მაგალითად 5.1 ცხრილის მეორე **FCj** ელემენტი აღწერს **j**-ური მოწყობილობის დაკავებულობის რიცხვს მოდელირების დროის განმავლობაში. ფაქტიურად **j** სიმბოლო – არის მოდელში ხელსაწყოს (მოწყობილობის) ნომერი. მაგალითად, **FC12** აღნიშნავს მე-12 ნომრის მქონე მოწყობილობის დაკავებულობის მრიცხველს, ანალოგიურად **FC3** – მესამე მოწყობილობის დაკავებულობის

მრიცხველს და ა.შ. **FC$** სახელი – სახელწოდებით ჩაწერილი 5.1 ცხრილის მეორე ელემენტი წარმოადგენს სიმბოლური სახელის მქონე მოწყობილობის დაკავებულობის რიცხვს. მაგალითად, ჩანაწერი **FC$JOE** აღნიშნავს **JOE** მოწყობილობის დაკავებულობის რიცხვს; **FC$KATHY** აღნიშნავს **KATHY** სახელწოდების მქონე მოწყობილობის დაკავებულობის რიცხვს და ა.შ. აღნიშნული მაგალითებიდან გამომდინარეობს, რომ დოლარის ნიშანი იწერება სტანდარტული რიცხვითი ატრიბუტის ჯგუფურ სახელსა და მოცემული ჯგუფის შესაბამისი წევრის სიმბოლურ სახელს შორის. 5.1-5.3 ცხრილებში ჩამოთვლილი ყველა სტანდარტული რიცხვითი ატრიბუტი წარმოადგენს მთელ რიცხვს. ზოგი ატრიბუტი წარმოადგენს მთელ რიცხვს, მაგალითად, მოწყობილობის დაკავებულობის მრიცხველი, რიგის მიმდინარე მდგომარეობა, მრავალარხიან მოწყობილობაში შესასვლელთა რიცხვი და ა.შ. სხვა ატრიბუტებს შესაძლოა ჩვეულებრივად გააჩნდეთ წილადი ნაწილი. მაგალითად, მრავალარხიანი მოწყობილობის საშუალო შემცველობა, რიგში ყოფნის საშუალო დრო.

GPSS-ში მიღებულია შენახულ იქნას სტანდარტული რიცხვითი ატრიბუტების მნიშვნელობათა მხოლოდ მთელი ნაწილები. მთელი რიცხვით ხელსაწყოს (მოწყობილობის) დატვირთვის მნიშვნელობას ან მრავალარხიანი მოწყობილობის ჩაწერის ცდის დროს მივიღებდით 0-ს (ან ყველაზე ცუდ, „გადაგვარების“ შემთხვევაში 1-ს), ვინაიდან დატვირთვის წილი მოიცემა ერთეულებში. იმისათვის, რომ თავიდან იქნას აცილებული ეს მდგომარეობა ხელსაწყოებისა და მრავალარხიან

მოწყობილობათა დატვირთვის განმსაზღვრელ რიცხვით ატრიბუტებს გააჩნიათ ათასების წილებში მოცემული მნიშვნელობები. დავუშვათ, რომ, თუკი მოდელირების დროის 58, 2%-ის განმავლობაში მოწყობილობა იმყოფება მუშაობის პროცესში, მაშინ მისი დატვირთვა უტოლდება 0, 582. ათასის წილებში მოცემულ მოწყობილობის დატვირთვას გააჩნია სახე 582.

ყურადღება უნდა მიექცეს 5.1–5.3 ცხრილებში ჩამოთვლილ სტანდარტულ რიცხვით ატრიბუტთაგან რამდენიმეს, განსაკუთებით მათ, რომლებიც მიეკუთვნება მრავალარხიან მოწყობილობებსა და რიგებს. გავიხსენოთ, რომ ყველა სტანდარტული რიცხვითი ატრიბუტი წარმოადგენს **GPSS**-ის მიერ ავტომატურად შეგროვილი და მოდელის სხვადასხვა მოწყობილობების, მრავალარხიან მოწყობილობათა და რიგებისათვის დაბეჭდილი სტატისტიკური მონაცემების მთელ ნაწილს. მოწყობილობებისთვის სტრა-ს მნიშვნელობები ემთხვევა თანამოსახელე ხელსაწყოებისთვის განსაზღვრულ ანალოგიურ-სტატისტიკურ მონაცემებს.

ასეთ შესაბამისობას აქვს ადგილი მრავალარხიანი მოწყობილობებისა და რიგებისათვის, გამონაკლისს წარმოადგენს მხოლოდ **ENTER** და **LEAVE, QUEUE** და **DEPART** ბლოკებში არანულოვანი **B** ოპერანდის გამოყენების შემთხვევა. ამ შემთხვევაში მრავალარხიან მოწყობილობათა და რიგებისათვის სტანდარტული რიცხვითი ატრიბუტები უნდა ინტერპრეტირ-დებოდეს იმის გათვალისწინებით, რომ მრავალარხიან მოწყობილობაში ტრანზაქტები იკავებს არხებს, ხოლო რიგებში

ჩვენ მივიჩნევთ მისი შემადგენელის ელემენტებს და არა ტრანზაქტებს.

ცხრილი 5.1

სტანდარტული რიცხვითი ატრიბუტები მოწყობილობებისთვის

დასახელება[1]	მ ნ ი შ ვ ნ ე ლ ო ბ ა
Fj ან **F$** სახელი	0, თუ მოწყობილობა არ არის დაკავებული; 1, თუ თავისუფალია
FCj ან **FC$** სახელი	მოწყობილობის დაკავებულობის რიცხვი
FRj ან **FR$** სახელი	მეათასედ ნაწილებში გამოსახული მოწყობილობის დატვირთვა
FTj ან **FT$** სახელი	მოწყობილობაზე ტრანზაქტების დაყოვნების საშუალო დროის მნიშვნელობის მთელი ნაწილი

ცხრილი 5.2

სტანდარტული რიცხვითი ატრიბუტები მრავალარხიანი მოწყობილობებისთვის

დასახელება[1]	მ ნ ი შ ვ ნ ე ლ ო ბ ა
Rj ან **R$** სახელი	მოწყობილობის შეუვსებელი ნაწილის ტევადობა
Sj ან **S$** სახელი	მოწყობილობის შევსებელი ნაწილის ტევადობა
SAj ან **SA$** სახელი	მოწყობილობის შუამდე შევსების მთელი ნაწილი
SCj ან **SC$** სახელი	მრავალარხიან მოწყობილობაში შესასვლელთა რიცხვის მრიცხველი. მოწყობილობისთვის **ENTER** ბლოკის ყოველი შესრულების დროს მრიცხველის მნიშვნელობა იზრდება აღნიშნული ბლოკის **B** ოპერანდის სიდიდის მნიშვნელობამდე
SMj ან **SM$** სახელი	მოწყობილობის მაქსიმალურად დაკავებული ტევადობა. იმახსოვრებს **Sj** (ან **S$** სახელი) მაქსიმალურ მნიშვ-ნელობას
SRj ან **SR$** სახელი	ათასეულების წილით გამოსახული მრავალარხიანი მოწყობილობის დატვირთვა
STj ან **ST$** სახელი	მოწყობილობაში ტრანზაქტის ყოფნის საშუალო დროის (ხანგრძლივობის) მთელი ნაწილი. იგი გამოითვლება ყველა სახისათვის.

ცხრილი 5.3

სტანდარტული რიცხვითი ატრიბუტები რიგებისთვის

დასახელება[1]	მ ნ ი შ ვ ნ ე ლ ო ბ ა
Qj ან **Q$** სახელი	რიგის სიგრძის მიმდინარე მნიშვნელობა (მიმდინარე შემცველობა)
QAj ან **QA$** სახელი	რიგის სიგრძის საშუალო მნიშვნელობის მთელი ნაწილი
QCj ან **QC$** სახელი	რიგში შესასვლელთა რაოდენობა. ყოველი შესრულების დროს **QUEUE** ბლოკში კონკრეტული რიგისთვის **QCj** (ან **QC$** სახელი) მნიშვნელობა იზრდება აღნიშნული ბლოკის **B** ოპერანდის სიდიდემდე.
QMj ან **QM$**სახელი	რიგის სიგრძის მაქსიმალური მნიშვნელობა. იგი არის მაქსიმალური მნიშვნელობა, რომელიც მიიღო **Qj** ან **Q$** სახელმა.
QTj ან **QT$** სახელი	რიგში ყოფნის საშუალო დროის მთელი ნაწილი ყველა სახისათვის, ანუ **QCj** (ან **QC$** სახელი). ეს ნიშნავს, რომ ნულოვანი შესასვლელები ასევე ითვალისწინება.
QXj ან **QX$** სახელი	ყველა სახისათვის რიგში ყოფნის საშუალო დროის მთელი ნაწილი **„QCj** მინუს **QZj“** (ან **QC** სახელი – **QZ$** სახელი), ანუ ნულოვანი შესასვლელების გამორიცხვით.
QZj ან **QZ$** სახელი	რიგში ნულოვანი შესასვლელების რიცხვი. ეს გახლავთ შესასვლელთა რიცხვი, რომელთა რიგში ყოფნის დრო უტოლდებოდა ნულს.

1j წარმოადგენს ჩვენი ინტერესის სფეროში მოქცეული კონკრეტული მოწყობილობის ნომერს. თუ სახელი მოცემულია სიმბოლურად, მაშინ იგი იწერება პოზიციაში „სახელი“.

შემთხვევით რიცხვთა **RNj** გენერატორები წარმოადგენს ჩვეულებრივ სტანდარტულ რიცხვით ატრიბუტებს. **GPSS**-ის დისკტერულ და უწყვეტ ფუნქციებს შესაძლოა გააჩნდეთ არამთელი მნიშვნელობები. საერთო შემთხვევაში ფუნქციის ჭეშმარიტი მნიშვნელობა შეიცავს წილად ნაწილს. წარმოადგენს თუ არა ფუნქციის მნიშვნელობას მთლიანი ან წილადი რიცხვი დამოკიდებულია კონტექსტისაგან.

1. თუ ფუნქცია გამოიყენება **GENERATE** და **ADVANCE** ბლოკების **B** ოპერანდის სახით, მაშინ გამოიყენებენ ჭეშმარიტ მნიშვნელობას. (არსებობს კიდევ ორი კონტექსტი, რომლებშიც შეიძლება გამოყენებულ იქნეს ფუნქციის ჭეშმარიტი მნიშვნელობა, მაგრამ ისინი ჯერ არ განხილულა ჩვენს მიერ).
2. სხვა ნებისმიერ კონტექსტში იყენებენ მხოლოდ ფუნქციის მთელ მნიშვნელობებს.

   დაბოლოს, როგორც **GPSS**-ის სტანდარტული რიცხვითი ატრიბუტები, ასევე შეიძლება განხილულ იქნას კონსტანტები. კონსტანტის ჯგუფური სახელი არის – **K**. ამ ჯგუფის ნომერს წარმოადგენს თავად კონსტანტა. ასე, მაგალითად, **K3** აღნიშნავს კონსტანტას 3, **K561** წარმოადგენს კონსტანტას 561 და ა.შ.

GPSS-ში კონსტანტების გამოყენების დროს **K** სიმბოლოს გამოყენება არ არის აუცილებელი.

5.2. ტრანზაქტების პარამეტრების მნიშვნელობათა ცვლილება.

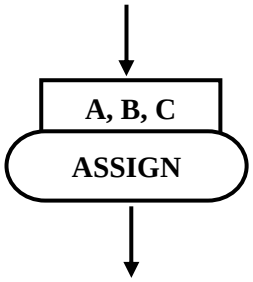
5.2.1. *ASSIGN* ბლოკი

ტრანზაქტების პარამეტრების მნიშვნელობები შესაძლებელია მივუთითოთ ან შევცვალოთ **ASSIGN** ბლოკში ტრანზაქტის შევლისას ანუ **ASSIGN** ბლოკი ტრანზაქტების პარამეტრების მნიშვნელობათა დაწესების ძირითად საშუალებას წარმოადგენს. ყოველ ტრანზაქტს შეიძლება გააჩნდეს პარამეტრის ნებისმიერი რიცხვი. ტრანზაქტების გენერაციის მომენტში ყველა მისი პარამეტრი ნულოვანია (მხოლოდ ისინი რომლებიც მოდელში გამოიყენება).

ASSIGN ბლოკის ჩაწერის ფორმატს აქვს შემდეგი სახე:

ASSIGN A, B, [C]

ხოლო გრაფიკულად ასე გამოისახება



ნახ. 5.1. **ASSIGN** ბლოკი **A** და **B** ოპერანდებით

A ოპერანდით მოიცემა პარამეტრის ნომერი, რომელსაც შემდგომში ენიჭება მნიშვნელობა. **A** ოპერანდი შესაძლოა იყოს სახელი, დადებითი მთელი რიცხვი, ბრჭყალებში ჩასმული გამოსახულება, სტანდარტული რიცხვითი ატრიბუტი, სტანდარტული რიცხვითი ატრიბუტის პარამეტრი და შემდეგ ნიშნები + (თუ საჭიროა გაზრდა) ან – (თუ საჭიროა შემცირება).

B ოპერანდი განსაზღვრავს მნიშვნელობას, რომელიც აუცილებელია დაემატოს, გამოაკლდეს ან რომლითაც აუცილებელია შეცვლილ იქნას **A** ოპერანდის მიერ მითითებული პარამეტრის მნიშვნელობა. თუკი ასეთი პარამეტრი არ არსებობს, იგი ითვლება 0-ის ტოლად. **B** ოპერანდი შესაძლოა იყოს ისეთი, როგორც **A** ოპერანდი. გარდა აღნიშნულისა, იგი შეიძლება აღმოჩნდეს რიცხვი და სტრიქონი.

C ოპერანდი აწესებს ფუნქცია-მოდიფიკატორის ნომერს. **C** ოპერანდის გამოყენების დროს **B** ოპერანდის მნიშვნელობა მრავლდება ფუნქცია-მოდიფიკატორის მნიშვნელობაზე. მიღებული ნამრავლი იქცევა მნიშვნელობად, რომელიც ცვლის **A** ოპერანდის მიერ მითითებული პარამეტრის მნიშვნელობას. უნდა აღინიშნოს, რომ **C** ოპერანდი განსაზღვრავს ფუნქციის ნომერს ან მის სახელს (არ არის აუცილებელი გამოყენებულ იქნას სტანდარტული რიცხვითი ატრიბუტები **FN** ან **FN$** მის წინ). თუკი გამოიყენება **FN$,** მაგალითად **FN3,** გამოითვლება **GPSS**-ის ნომერი 3 ფუნქცია. მიღებული შედეგი გამოიყენება **GPSS**-ის მეორე ფუნქციის განსასაზღვრად, რომელიც ასევე გამოითვლება და მისი მნიშვნელობა მრავლდება **B** ოპერანდის მნიშვნელობაზე.

მოვიყვანოთ ბლოკ **ASSIGN**-ის ჩაწერის მაგალითები:

ASSIGN 1, 754. 3

ASSIGN 4+ , Q5
ASSIGN 3- , 5. 85, 7
ASSIGN Name “Plan”
ASSIGN Tr1-, (Normal (32,Sredn, SrKvOtkl), ExpDis

პირველ მაგალითში 1-ელ პარამეტრს ენიჭება 754. 3. მეორე მაგალითში მე-4 პარამეტრის მნიშვნელობას ემატება მიმდინარე რიგის სიგრძის მე-5 ნომრის მნიშვნელობა. მესამე მაგალითში მე-3 პარამეტრის მნიშვნელობას აკლდება 5.85-ის ნამრავლი ნომერი 7 ფუნქციის გამოთვლილ მნიშვნელობაზე. მეოთხე მაგალითში **Name** სახელით ცნობილ პარამეტრს ენიჭება სტრიქონი **Plan**. მეხუთე მაგალითში გამოითვლება ფრჩხილებში ჩასმული გამოსახულება, გამრავლდება **ExpDis** სახელწოდებით მოცემულ ფუნქციაზე და მიღებული ნამრავლი გამოაკლდება **Tr1** სახელით ცნობილი პარამეტრის მნიშვნელობას.

5.2.2. PLUS ბლოკი

GPSS World-ში ტრანზაქტების პარამეტრების მნიშვნელობები აგრეთვე შესაძლოა შეიცვალოს **PLUS** ბლოკით. **PLUS** ბლოკი გამოთვლის გამოსახულებას და ის შესაძლებელია ჩაწეროს მის პარამეტრში. **PLUS** ბლოკს გააჩნია შემდეგი ფორმატი:

PLUS A, [B]

სადაც **A** ოპერანდი არის გამოსახულება. ის შეიძლება იყოს სახელი, რიცხვი,
სტრიქონი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი.

B ოპერანდი – ტრანზაქტის პარამეტრის ნომერი, რომელშიც შეინახება შედეგი. იგი შეიძლება იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი. მაგალითად:

PLUS (Exponential (23, 0, 32. 7)+47.74), Sed

PLUS ბლოკში ტრანზაქტის შესვლისას გამოითვლება **A** ოპერანდით მოცემული ფრჩხილებში ჩასმული გამოსახულება, შემდეგ გამოთვლილი შედეგი მიენიჭება **Sed** სახელით მითითებულ პარამეტრს. თუ ასეთი პარამეტრი არ არსებობს, იგი იქმნება.

5.2.3. INDEX ბლოკი

ტრანზაქტის პარამეტრის მნიშვნელობის შეცვლა შესაძლებელია აგრეთვე **INDEX** ბლოკის საშუალებით. აღნიშნულ ბლოკს გააჩნია შემდეგი ზოგადი სახე:

INDEX A, B

სადაც **A** ოპერანდი არის ტრანზაქტის პარამეტრის ნომერი. ის შესაძლებელია იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი.

B ოპერანდი – რიცხვითი მნიშვნელობა, რომელიც მიემატება **A** ოპერანდით მოცემულ მნიშვნელობას. იგი გახლავთ იმავე ტიპის, როგორც **A** ოპერანდი. მიღებული შედეგი შეიტანება ტრანზაქტის პირველ პარამეტრში. მაგალითად:

INDEX Slag1, (Normal (21, Mat, SreOtk)+X$post)

ტრანზაქტის შესვლის დროს გამოითვლება **B** ოპერანდით მითითებული ფრჩხილებში ჩასმული გამოსახულება და მიემატება **Slag1** სახელით მითითებული პარამეტრის მნიშვნელობას. შეკრების შედეგი მიენიჭება ტრანზაქტის ნომერ პირველ პარამეტრს. თუ ასეთი პარამეტრი არ არსებობს, მაშინ იგი იქმნება. აგრეთვე თუ **A** ოპერანდით მითითებული პარამეტრი არ არსებობს, მაშინ გამოიტანება შემდეგი სახის შეტყობინება: „მიმართვა არარსებულ პარამეტრთან“.

5.2.4. MARK ბლოკი

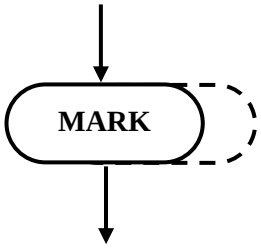
ხშირად საჭიროა იმის ცოდნა თუ რა დროა საჭირო ტრანზაქტის ერთი წერტილიდან მეორე წერტილში გადასაადგილებლად. ამისათვის გამოიყენება **MARK** ბლოკი,

რომელსაც აქტიურ ტრანზაქტში ან მის პარამეტრში შეაქვს აბსოლუტური სამოდელო დროის მნიშვნელობა.

MARK ბლოკის ფორმატს გააჩნია შემდეგი სახე:

MARK [A]

ხოლო გრაფიკულად ასე გამოისახება:



ნახ.5.2. **MARK** ბლოკი **A** ოპერანდით

სადაც, **A** ოპერანდი განსაზღვრავს ტრანზაქტის პარამეტრის ნომერს, რომელშიც ჩაიწერება სამოდელო დროის აბსოლუტური მნიშვნელობა. იგი შესაძლებელია იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა-ს პარამეტრი. მაგალითად:

MARK

MARK SHESVLA

პირველ მაგალითში **A** ოპერანდი არ გამოიყენება. ამ ბლოკში ფარულად შესულ ტრანზაქტს უწესდება სისტემაში შესვლის დრო, რომელიც აბსოლუტური სამოდელო დროის ტოლია.

მეორე მაგალითში **A** ოპერანდი გამოიყენება, ამიტომ აბსოლუტური სამოდელო დროის მნიშვნელობა (სტრა **AC1**) შეიტანება **SHESVLA** სახელის მქონე შესული ტრანზაქტის პარამეტრში. თუ აღნიშნული პარამეტრი არ არსებობს, იგი იქმნება.

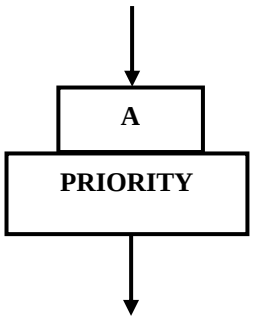
შენიშვნა: **ASSIGN, PLUS, INDEX, MARK** და **LOOP** ბლოკებში ოპერანდების ჩაწერის დროს, რომლებიც წარმოადგენს ტრანზაქტის პარამეტრს, სტრა **P** არ გამოიყენება.

5.3. ტრანზაქტების პრიორიტეტის დონის ცვლილება. PRIORITY ბლოკი

მოდელში ტრანზაქტის შესვლისას მისი პრიორიტეტი განისაზღვრება **GENERATE** ბლოკის შესაბამისი **E** ოპერანდის მნიშვნელობით. აქამდე ჩვენს მიერ განხილულ მოდელებში ტრანზაქტის პრიორიტეტის მნიშვნელობა მისი მოდელში შესვლის (გადაადგილების) დროს მუდმივი და პრიორიტეტის აღნიშნული საწყისი მნიშვნელობის ტოლი რჩებოდა. მიუხედავად ამისა, მოდელირების პროცესში არსებობს პრიორიტეტის დონის დინამიური შეცვლის შესაძლებლობა. ამ გარემოებას ტრანზაქტის მიმდინარე მოვლენათა ჯაჭვში განთავსების თვალსაზრისით გააჩნია ძალზე დიდი მნიშვნელობა, რაც თავის მხრივ განსაზღვრავს თანამიმდევრობას, რომელშიც ტრანზაქტები შეირჩევა მოდელში მოძრაობის აღსადგენად. ამგვარად, იმის შესაძლებლობა, დინამიურად შეიცვალოს ტრანზაქტების პრიორი-

ტეტის დონე, **GPSS**-ში მოდელირების თვალსაზრისით იძენს სულ დიდ მნიშვნელობას.

PRIORITY ბლოკში ტრანზაქტების შესვლის დროს იცვლება მისი პრიორიტეტის დონე. 5. 3 ნახაზზე გამოსახულია **PRIORITY** ბლოკი **A** ოპერანდთან ერთად. (აღნიშნულ ბლოკში ასევე შესაძლებელია მოცემულ იქნეს **B** ოპერანდიც). ზუსტად **A** ოპერანდის მნიშვნელობა კი მიიჩნევა ტრანზაქტის **PRIORITY** ბლოკში შემავალი პრიორიტეტის ახალი დონის სახით. სავსებით ნათელია ის გარემოება, რომ ახალი მნიშვნელობა შესაძლოა აღმოჩნდეს ძველი მნიშვნელობის ტოლი. ამ შემთხვევაში არავითარი ცვლილება არ მოხდება.



ნახ. 5.3. **PRIORITY** ბლოკი **A** ოპერანდით

PRIORITY ბლოკი არასოდეს არ კრძალავს ტრანზაქტის შესვლას; მისი შესვლის შემთხვევაში სრულდება შემდეგი მოქმედებები.

1. წესდება ტრანზაქტის პრიორიტეტის ახალი დონე;
2. მიმდინარე მოვლენათა ჯაჭვში ტრანზაქტი პრიორიტეტების შესაბამისი კლასის ტრანზაქტებს შორის იკავებს ბოლო ადგილს;

3. ინტერპრეტატორი აგრძელებს ტრანზაქტის მოდელში გადაადგილებას იქამდე, ვიდრე არსებობს ამის შესაძლებლობა;
4. დაბოლოს, როდესაც უკვე შეუძლებელია ტრანზაქტის შემდგომი გადაადგილება, ინტერპრეტატორი ახორციელებს მიმდინარე მოვლენათა ჯაჭვის ხელახალ გადამოწმებას. ამისათვის არსებობს გარკვეული მიზეზები: **PRIORITY** ბლოკში პრიორიტეტის მნიშვნელობა შესაძლოა აღმოჩნდეს მცირე. შედეგად ტრანზაქტიც და ის ადგილიც ჯაჭვში რომლისგანაც გაგრძელებულ უნდა იქნას გადამოწმება ირევა ჯაჭვის ბლოკში.

აღნიშნული წერტილიდან თუკი მოხდებოდა ჯაჭვის გადამოწმების განახლება, ამ შემთხვევაში ტრანზაქტების ნაწილი შესაძლოა გატარებული (შეშვებული) აღმოჩენილიყო. თავიდან ჯაჭვის გადამოწმების განახლება იძლევა იმის გარანტიას, რომ არც ერთი ტრანზაქტი გაშვებული არ იქნება.

ტრანზაქტის პრიორიტეტის დონე წარმოადგენს **GPSS**-ის სტანდარტულ რიცხვით ატრიბუტს. აღნიშნული ატრიბუტის სახელწოდება გახლავთ **PR (PRIORITY LEVEL –** *პრიორიტეტის დონე***).** ტრანზაქტის პრიორიტეტზე მითითება ამგვარი სახით წარმოგვიდგება დასრულებულად, ვინაიდან ტრანზაქტს გააჩნია პრიორიტეტის მხოლოდ ერთი მნიშვნელობა. აქტიური ტრანზაქტის პრიორიტეტის მნიშვნელობა რომელიმე ბლოკის ოპერანდის სახით მისი გამოყენების დროს იქცევა **PR** ატრიბუტის მნიშვნელობად. ზემოთ მინიშნებული იყო იმ გარემოებაზე, რომ **PR**-ის მნიშვნელობად შესაძლოა იქცეს მთელი რიცხვი 0-დან 127-მდე

დიაპაზონში. **PR** ატრიბუტი შესაძლოა გამოყენებულ იქნას შესაბამისად, ბლოკის, ოპერანდის, ფუნქციის არგუმენტის და ა.შ. სახით.

ახლა განვიხილოთ **PRIORITY** ბლოკის გამოყენების შესაძლებლობა დროის კვანძების წარმოშობის შემთხვევაში. მოვლენათა წარმოშობის თანამიმდევრობა ზუსტად განისაზღვრება მიმდინარე მოვლენათა ჯაჭვში ტრანზაქტების ფარდობითი განლაგებით. აღნიშნული ფარდობითი განლაგება შესაძლოა იცვლებოდეს პრიორიტეტების დონეების ცვალებადობის დროს. დავუშვათ, რომ წარმოიშვა სიტუაცია, როდესაც ორი მოვლენა ხდება ერთდროულად და აუცილებელია გაგვაჩნდეს უნარი ზემოქმედება მოვახდინოთ აღნიშნულ მოვლენათა დამუშავების თანამიმდევრობაზე. ზემოხსენებული კი ხორციელდება გარკვეული მოვლენის მქონე ტრანზაქტისათვის გაცილებით უფრო მაღალი პრიორიტეტის დაწესებით (მინიჭებით), რომლის დამუშავებაც უნდა დაიწყოს გაცილებით უფრო ადრეულ სტადიაზე.

აღნიშნული შემთხვევის უფრო დეტალურად განხილვის მიზნით განვიხილოთ ამოცანა პუასონის შესასვლელი ნაკადით და მოწყობილობა ექსპონენციალური განაწილებით. აღნიშნულ მოდელში ავტომობილი მომსახურებისათვის რჩება მხოლოდ იმ შემთხვევაში, თუკი სადგომზე არის ადგილი. იბადება კითხვა, როგორ მოვიქცეთ იმ შემთხვევაში, როდესაც მოვლენა – „ახალი ავტომობილის შემოსვლა“ და „მომსახურების დასასრული“ წარმოიშვება ერთდროულად, ხოლო სადგომზე ყველა ადგილი დაკავებულია. იმ შემთხვევაში თუკი თავიდან ადგილი ექნება

ავტომობილის შემოსვლის მოვლენის დამუშავებას, ტრანზაქტი, რომელიც წარმოადგენს ავტომობილის სიმბოლოს, აღმოაჩენს, რომ მრავალარხიანი მოწყობილობა **SPACE** შევსებულია და უცებ გადადის ჭდეზე **BYBYE**. თუკი ადგილი აქვს საპირისპირო მოვლენას, თავდაპირველად დამუშავებული იქნება მომსახურების დასრულების მოვლენა, მაშინ კი მომლოდინე ტრანზაქტებიდან ერთ-ერთი – ავტომობილი შეძლებს გადასვლას (შესვლას) **„LEAVE SPACE“** ბლოკში იქამდე, ვიდრე შემოსული ავტომობილი არ იქნება მომსახურებული. ეს ნიშნავს, რომ მისთვის სადგომზე მოიძებნება ადგილი, ხოლო იგი დაელოდება მომსახურებას.

არსებობს გარკვეული საფუძველი იმისათვის, რომ ზემოთ განხილულ ამოცანაში წარმოდგენილი მოვლენა, ანუ მომსახურების დასრულების მოვლენა დამუშავებულ იქნას მომდევნო ახალი ავტომობილის შემოსვლამდე გაცილებით უფრო ადრე. თითოეული ტრანზაქტთაგანი მოდელში შედის ფარულად დაწესებული პრიორიტეტის დონის მნიშვნელობით, ანუ იგი 0-ის ტოლია. თუ ჩვენ **PRIORITY** ბლოკს მოდელში მოვათავსებთ სადღაც ავტომობილის შემოსვლის შემდეგ, მაგრამ იქამდე, ვიდრე შესაბამისი ტრანზაქტი შევა **ADVANCE** ბლოკში, დროის კვანძის წარმოშობის შემთხვევაში წამმზომის კორექციის ფაზის შესრულების პროცესში, მომსახურების დამამთავრებელი ტრანზაქტი დასმული იქნება მიმდინარე მოვლენათა ჯაჭვში სულ ახლახანს შემოსული ტრანზაქტის წინ.

ყოველივე ზემოხსენებულის მომდევნო გადამოწმების ფაზის პროცესში მომსახურების დასრულების მოვლენა აუცილებლად იქნება დამუშავებული შემოსვლის მოვლენამდე ადრე, ხოლო

ასეთი თანამიმდევრობის მისაღწევად კი გახლდათ მიმართული ჩვენი მუშაობა. აღნიშნული ბლოკი, მაგალითად, შეიძლება მოვათავსოთ **GENERATE** ბლოკის შემდგომ. რა თქმა უნდა, მისი მოთავსება შესაძლებელია სხვა ადგილებშიც.

PRIORITY ბლოკი შეიძლება გამოყენებულ იქნას მრავალი შემთხვევისათვის, და არა მხოლოდ ერთდროულად წარმოშობილ მოვლენათა დამუშავების თანამიმდევრობით მართვისათვის.

5.4. ელემენტების შერჩევა მათი მდგომარეობის მიხედვით. SELECT ბლოკი

რამდენიმე მოწყობილობისა და რიგის მქონე მომსახურების სისტემაში შემავალი მოთხოვნები უბრალო შემთხვევაში ასრულებენ ორიდან ერთ-ერთ მოქმედებას.

1. თუ ერთ-ერთი მოწყობილობა ამ მომენტში თავისუფალია, მოთხოვნა იკავებს აღნიშნულ მოწყობილობას.
2. თუკი ყველა მოწყობილობა დაკავებულია, მოთხოვნა უერთდება გაცილებით უფრო მოკლე რიგს და რჩება მასში იქამდე, ვიდრე არ იქნება მომსახურებული.

მაგალითად, თუკი ერთი ან რამდენიმე მოწყობილობა თავისუფალია, მოთხოვნას შეუძლია მიიღოს გადაწყვეტილება დარჩეს იმავე მოწყობილობის რიგში რომელიც მის მიერ არის შერჩეული, მაგრამ რომელიც იმავდროულად იმ მომენტისათვის დაკავებულია; უმოკლესი რიგისთვის უბრალო მიერთებისა და მასში ლოდინის ნაცვლად, მოთხოვნას შეუძლია სხვა რიგებში

გადასვლა თუ იგი გრძნობს, რომ ძველ რიგში მას დიდხანს ლოდინი მოუწევს, ხოლო სხვა რიგი კი მოძრაობს გაცილებით სწრაფად.

აღნიშნულ პარაგრაფში ჩვენ განვიხილავთ მოთხოვნის ქცევის ყველაზე მარტივ შემთხვევას. პირველ ყოვლისა მოქმედება 1-ის შესაბამისად უნდა აღმოვაჩინოთ ამ დროისათვის თავისუფალი მოწყობილობის მოძებნის მეთოდი. მეორეს მხრივ, მოქმედება 2-ის შესაბამისად, ჩვენ განვიხილავთ როგორ აღმოვაჩინოთ, თუ რიგთაგან რომელი წარმოადგენს უმოკლესს. დაბოლოს, აღვწერთ მოწყობილობის არჩევის პროცესის მოდელირების მეთოდს.

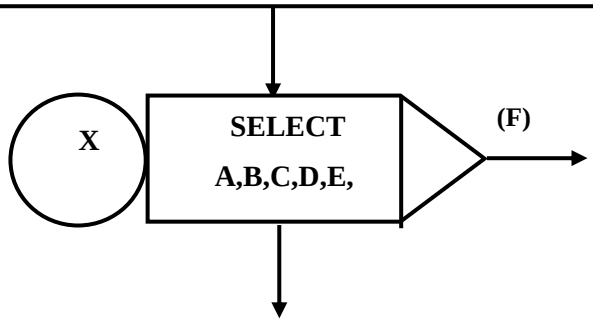
5.4.1. SELECT ბლოკი დამოკიდებულების ოპერატორის რეჟიმში

GPSS-ში არის ბლოკი, რომელიც შეიძლება გამოყენებულ იქნას განსაზღვრული ჯგუფის პირობის დასაკმაყოფილებლად. აღნიშნულ ბლოკში ტრანზაქტის ელემენტების სიმრავლის გადასათვალიერებლად იმის გამოსარკვევად, თუ აკმაყოფილებს ერთ-ერთი მათგანი განსაზღვრულ რიცხვით პირობას (შესვლის დროს), ეს უკანასკნელი ახორციელებს განსაზღვრული ჯგუფის ელემენტების მდგომარეობის გადათვალიერებას. გადათვალიერება სრულდება ელემენტების ნომრების ზრდის თანამიმდევრობით (რიგის მიხედვით). თუკი ნაპოვნია ისეთი ელემენტი, რომელიც აკმაყოფილებს დადგენილ პირობას, გადათვალიერება წყდება. წინააღმდეგ შემთხვევაში გადათვალიერება წყდება მხოლოდ იმის განსაზღვრის შემდეგ, როდესაც სიმრავლის არც ერთი წევრი არ

აკმაყოფილებს აღნიშნულ პირობას. ნებისმიერ შემთხვევაში, გადათვალიერების დასრულების შემდეგ (იმ წუთასვე ან როდესაც ამის საშუალებას მისცემს მომდევნო ბლოკი) ტრანზაქტი გადაადგილდება მოდელში.

ქვემოთ მოყვანილია ბლოკის მიერ შესრულებულ გადათვალიერებათა რამდენიმე მაგალითი:

1. გადათვალიერდეს 1, 2 და 3 მოწყობილობები იმისათვის, რომ გარკვეულ იქნას იმყოფება თუ არა რომელიმე მათგანი თავისუფალ მდგომარეობაში (ანუ უნდა განისაზღვროს არსებობს კი თუნდაც ერთი მოწყობილობა, რომლისთვისაც **F** ატრიბუტი ნულის ტოლია).
2. გადაიხედოს მრავალარხიანი მოწყობილობები იმის გასარკვევად, თუ არსებობს თუნდაც ერთი მოწყობილობა, რომლის დატვირთვაც 25%-ზე მცირეა (ანუ უნდა განისაზღვროს არსებობს კი თუნდაც ერთი მოწყობილობა, რომლის **SR** ატრიბუტი 250-ზე მცირეა).
3. გადაიხედოს მე-13, 14, 15, 16 და 17 რიგები იმის გასარკვევად, თუ არსებობს ერთ-ერთი მათგანი, რომლის ყოფნის საშუალო დრო, თუკი გამოვრიცხავთ ნულოვან შესასვლელებს არ არის 3-ზე მცირე (ანუ განსაზღვრულ უნდა იქნას თუ არსებობს თუნდაც ერთ-ერთი აღნიშნული რიგებიდან, რომლის **X** ატრიბუტი გადააჭარბებდა ან აღმოჩნდებოდა 3-ის ტოლი). ასეთი გადათვალიერების განსახორციელებლად იყენებენ **SELECT** (არჩევა) ბლოკს.



ნახ. 5.4. SELECT ბლოკი დამხმარე ოპერატორითა და **A, B, C, D, E** და **F** ოპერანდებით

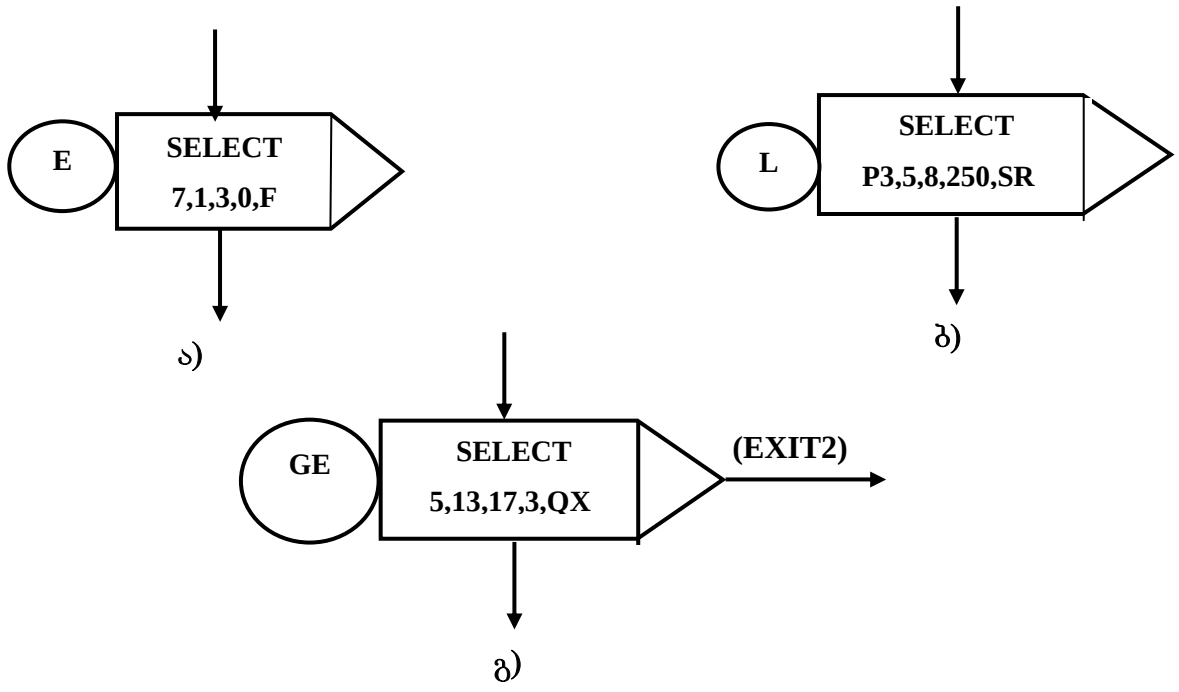
აღნიშნული ბლოკი თავის ოპერანდებთან ერთად ნაჩვენებია 5.4 ნახაზზე. **E** ოპერანდი აწესებს სტანდარტული რიცხვითი ატრიბუტის ჯგუფურ სახელს (მაგალითად, **F, SR, X**, ისევე როგორც ზემოთ მოყვანილ მაგალითებში). **B** და **C** ოპერანდები მიუთითებენ ჯგუფის წევრების შესაბამისად მინიმალურ და მაქსიმალურ ნომრებს. ისინი მიეკუთვნება განსაზღვრულ ჯგუფებს, რომელთა წევრებიც მოცემულია (მაგალითად, 1 და 3, 5 და 8, 13 და 17, როგორც ზემოთ მოყვანილ მაგალითებში**). D** ოპერანდი მიუთითებს მონაცემის მნიშვნელობას, რომლითაც უნდა მოხდეს ოპერანდის მიერ დაწესებული ატრიბუტის მნიშვნელობის შედარება (მაგალითად, 0, 250, 3 როგორც ზემოთ მოყვანილ მაგალითებში).

რიგის დამოკიდებულება სტრა-ს და **D** ოპერანდის მნიშვნელობას შორის წესდება **X** ჩანაწერით 5.4 ნახაზზე წარმოდგენილი დამხმარე ოპერატორის მეშვეობით. დამხმარე ოპერატორმა შესაძლოა მიიღოს **G, GE, L, LE, E** და **NE** მნიშვნელობები ისეთ თანაფარდობათა აღსანიშნავად როგორც: მეტია, მეტია ან ტოლი, ნაკლებია, ნაკლებია ან ტოლი, ტოლი და განსხვავებულია.

SELECT ბლოკის **A** ოპერანდი მიუთითებს ტრანზაქტის პარამეტრის ნომერს. თუ გადასინჯვის დროს აღმოჩნდება სიმრავლის წევრი, რომელიც აკმაყოფილებს დადგენილ პირობას, ამ წევრის ნომერი გადაიწერება აღნიშნული პარამეტრის მნიშვნელობის სახით. ამის შემდგომ ტრანზაქტი ტოვებს **SELECT** ბლოკს.

რათქმა უნდა შესაძლებელია მოხდეს ისე, რომ ჯგუფის არც ერთმა წევრმა არ დააკმაყოფილოს დადგენილი პირობა. თუკი ამასთან **SELECT** ბლოკის არააუცილებელი **F** ოპერანდი არ გამოიყენება, ტრანზაქტის მითითებული პარამეტრის მნიშვნელობად გვევლინება ნული და ტრანზაქტი **SELECT** ბლოკიდან გადადის მომდევნო ბლოკში.

იმ შემთხვევაში თუ **F** ოპერანდი განსაზღვრულია და გადათვალიერებამ არ უჩვენა დადებითი შედეგი, ტრანზაქტი გამოდის ბლოკიდან მაგრამ შედის არა მომდევნო ბლოკში, არამედ იმ ბლოკში, რომლის სახელიც მითითებულია **F** ოპერანდის სახით. როდესაც ადგილი აქვს ასეთ გამოსვლას, ტრანზაქტის პარამეტრის მნიშვნელობა არ არის ნულის ტოლი. ამ შემთხვევაში **SELECT** ბლოკი არ ახდენს გავლენას პარამეტრის მნიშვნელობაზე. 5.5 ნახაზზე ნაჩვენებია გადათვალიერების ზემოთ წარმოდგენილი სამი ტიპის შესაბამისი **SELECT** ბლოკის გამოყენების სამი მაგალითი.



ნახ.5.5. **SELECT** ბლოკის გამოყენების მაგალითები

5.5 ა ნახაზზე ნაჩვენებია ოპერანდები რომლებიც განკუთვნილია 1, 2 და მე-3 მოწყობილობათა გადასათვალიერებლად, იმის განსასაზღვრად, არსებობს თუ არა მათ შორის ისეთი ოპერანდი, რომლისთვისაც **F** ატრიბუტი ნულის ტოლი იქნებოდა. ბლოკში არ არის გათვალისწინებული დამატებითი გამოსასვლელი. შედეგად, ტრანზაქტი გაივლის რა **SELECT** ბლოკს, ეცდება შევიდეს მომდევნო ბლოკში. თუკი გადათვალიერება წარმატებით დასრულდება ტრანზაქტი გამოვა აღნიშნული მოწყობილობის ნომრის თანაბარი (1, 2 ან 3) პარამეტრის მნიშვნელობით 7, სადაც პირველი მათგანისათვის აღმოჩენილია, რომ მისი **F** ატრიბუტი ნულის ტოლია. თუკი გადათვალიერებამ არ მოგვცა დადებითი შედეგი, ტრანზაქტის მე-7-ე პარამეტრი ნულოვანი მნიშვნელობით ტოვებს **SELECT** ბლოკს.

დავუშვათ, რომ წინა მაგალითში სამიდან ორ მოწყობილობას გადათვალიერების შესრულების მომენტში გააჩნია ნულის ტოლი **F** ატრიბუტი. მაშინ მე-7-ე პარამეტრში ჩაწერილი მოწყობილობის ნომრად აღმოჩნდება უმცირესი აღნიშნული მოწყობილობების ნომრებიდან. თუკი მაგალითად, მე-2 და 3 მოწყობილობებისათვის ატრიბუტი ნულის ტოლია, მაშინ მე-7-ე პარამეტრის მნიშვნელობა იქნება 2-ის ტოლი.

SELECT ბლოკში ტრანზაქტის შესვლის დროს (ნახ. 5. 5 ბ) გადაიხედება 5, 6, 7 და 8 მრავალარხიანი მოწყობილობები იმის განსასაზღვრად, თუ რომელ მათგანს გააჩნია 250 ათასეულის წილზე ნაკლები დატვირთვა. თუკი გადათვალიერება წარმატებით დასრულდა მაშინ 5, 6, 7 და 8 რიცხვი ჩაწერილი იქნება პარამეტრში, რომლის ნომერიც განსაზღვრულია **P3** ტრანზაქტით, ხოლო თავად ტრანზაქტი გადადის მომდევნო ბლოკში. **SELECT** ბლოკში კვლავაც არ არის განსაზღვრული **F** ოპერანდი. ამიტომ, თუ გადათვალიერება არ იქნება წარმატებული, შესაბამისი პარამეტრის მნიშვნელობა იქნება ნულის ტოლი.

5. 5 გ ნახაზზე ნაჩვენებია **SELECT** ბლოკიდან გამოსვლა. აღნიშნულ ბლოკში ტრანზაქტის შესვლის დროს გადაიხედება რიგები ნომრებით 13-დან 17-მდე და განისაზღვრება არის თუ არა რომელიმე მათგანი, რომლის ატრიბუტიც გახლავთ **QX,** რომელიც 3-ზე მეტია ან ტოლი. თუკი ერთ-ერთი აღნიშნულ რიგთაგანი აკმაყოფილებს წაყენებულ პირობას, მისი ნომერი ჩიწერება ტრანზაქტის მე-5 პარამეტრში, რომელიც შემდგომში გამოდის **SELECT** ბლოკიდან და გადადის მომდევნო ბლოკში. თუკი აღნიშნულ რიგთაგან არც ერთი არ აკმაყოფილებს

წაყენებულ პირობას, ტრანზაქტი გადადის ბლოკში რომლის სახელიც მითითებულია **F** ოპერანდის მიერ, ანუ ბლოკში **EXIT 2**. ამასთან, პარამეტრი 5 არ იცვლის თავის მნიშვნელობას.

დაბოლოს, აღვნიშნოთ **SELECT** ბლოკის შემდეგი თავისებურებანი. პირველყოვლისა, რომ **A, B, C** და **D** ოპერანდების მნიშვნელობები შესაძლოა განისაზღვრებოდეს პირდაპირ ან ირიბად. 5.4 ნახახზე მოყვანილ მაგალითებში ოპერანდის ირიბ (არაპირდაპირ) გამოხატულებას ადგილი ქონდა მხოლოდ **A** ოპერანდში (ნახ. 5. 5 ბ).

მეორეს მხრივ, გადათვალიერებისათვის განსაზღვრულ პირობას გააჩნია სახე "სტრა (დამოკიდებულების ოპერატორი) **D** ოპერანდის მნიშვნელობა". მაშასადამე, ნახ. 5. 4 ბ ნახაზზე წარმოდგენილ პირობად გვევლინება "**SR** მნიშვნელობა ნაკლებია 250" და არა პირობა " 250 ნაკლებია **SR** მნიშვნელობაზე".

მესამე შემთხვევაში, **B** ოპერანდის მიერ განსაზღვრული ნომრიდან **C** ოპერანდის მიერ განსაზღვრულ ნომრამდე მოცემული ჯგუფის ყველა წევრი უნდა იქნას გადასინჯული. **SELECT** ბლოკის ამგვარი გამოყენების დროს არ არის შესაძლებლობა, მაგალითად, გადასინჯულ იქნას 3, 4, 7, 10 და 12 ნომრების მქონე მოწყობილობები, რომელთა **F** ატრიბუტის მნიშვნელობა იქნებოდა ნულის ტოლი.

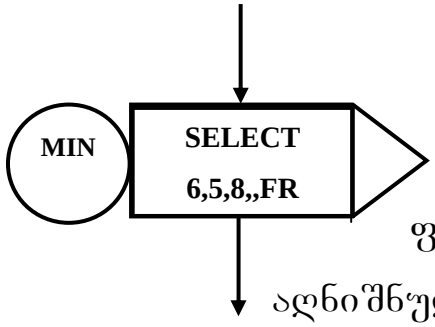
5.4.2. SELECT ბლოკი MIN და MAX რეჟიმებში

SELECT ბლოკის გამოყენების აღწერილი რეჟიმი საშუალებას იძლევა გამორკვეულ იქნას არსებობს თუ არა განსაზღვრული ჯგუფის წევრებს შორის რაიმე პირობის დამაკმაყოფილებელი ელემენტი. აღნიშნული ბლოკი ასევე შეიძლება გამოყენებულ იქნას ჯგუფის წევრის მოძებნის რეჟიმში, რომლის განსაზღვრული ატრიბუტის მნიშვნელობა წარმოადგენს მინიმალურს ან/და მაქსიმალურს. ასეთი რეჟიმი საშუალებას იძლევა პასუხი იქნეს გაცემული მაგალითად, შემდეგი სახის შეკითხვებზე:

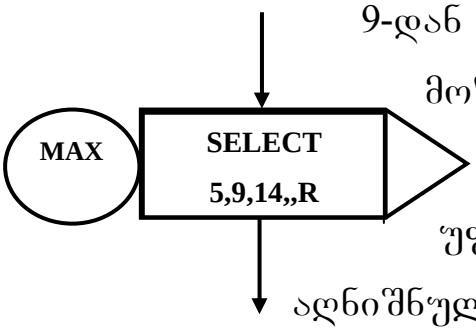
1. რიგთაგან რომელს გააჩნია მიმდინარე შიგთავსის ყველაზე მცირე მნიშვნელობა?
2. მოწყობილობათაგან რომელს გააჩნია ყველაზე მცირე დატვირთვა?
3. მრავალარხიან მოწყობილობათაგან რომელს გააჩნია ყველაზე მეტი თავისუფალი არხი?

ყველა ზემოაღნიშნული შემთხვევისათვის საერთოა ერთი კითხვა: “ნომრების აღნიშნულ დიაპაზონში მოცემული ჯგუფის რომელ წევრს გააჩნია შესაბამისი ატრიბუტის მინიმალური ან/და მაქსიმალური მნიშვნელობა?”. ასეთ კითხვებზე პასუხის გასაცემად **SELECT** ბლოკი შეიძლება გამოყენებულ იქნას **MIN** ან **MAX** რეჟიმებში. **SELECT** ბლოკის გამოყენების ასეთი რეჟიმისათვის დამხმარე ოპერატორების როლს თამაშობს **MIN** და **MAX** სიტყვები. თავად ბლოკი და მისი ოპერანდები გამოსახულია 5.6 ნახაზზე. **A, B, C** და **E** ოპერანდები იგივე რჩება როგორ ამას

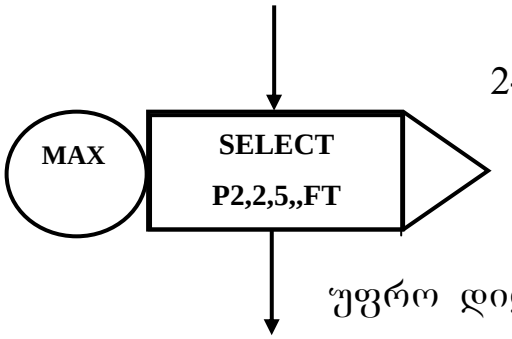
ადგილი ჰქონდა **SELECT** ბლოკის დამოკიდებულების ოპერატორის რეჟიმში. ერთად აღებული, ისინი ადგენენ თუ რომელი ატრიბუტი უნდა იქნეს შემოწმებული (**E** ოპერანდი), ჯგუფის წევრების რომელი ნომრები დაექვემდებარება შემოწმებას (**B** და **C** ოპერანდები) და სად უნდა იქნას მოთავსებული შემოწმების შედეგი (**A** ოპერანდი). ვინაიდან ატრიბუტმა უნდა უპასუხოს რომელიმე აბსოლუტურ პირობას, **D** ოპერანდის მნიშვნელობა არ განისაზღვრება. დამხმარე ოპერატორის მნიშვნელობად გვევლინება **MIN** ან **MAX**, იმისაგან დამოკიდებულებით, ატრიბუტის რომელი მნიშვნელობა გვაინტერესებს ჩვენ. დაბოლოს, ვინაიდან ელემენტების ერთ-ერთ წევრს გააჩნია მინიმალური ან/და მაქსიმალური მნიშვნელობა არ არის აუცილებელი გათვალისწინებულ იქნას **MIN** ან **MAX** რეჟიმში **SELECT** ბლოკიდან დამატებითი გამოსავალი. სხვაგვარად რომ ვთქვათ, ბლოკს ამ რეჟიმში არ გააჩნია არააუცილებელი **F** ოპერანდი.

ა) 

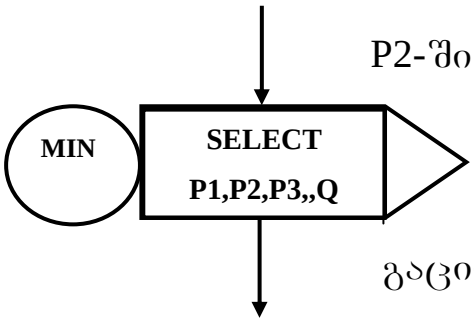
5-დან 8-მდე ნომრების მქონე რომელ მოწყობილობას გააჩნია მინიმალური ფარდობითი (შედარებითი) დატვირთვა? აღნიშნული მოწყობილობის ნომერი მოვათავსოთ მე-6 პარამეტრში.

ბ) 

9-დან 14-მდე ნომრების მქონე მრავალარხიანი მოწყობილობებიდან რომელს გააჩნია დარჩენილი ტევადობის გაცილებით უფრო დიდი ზომა? აღნიშნული მოწყობილობის ნომერი მოვათავსოთ მე-5 პარამეტრში.

გ) 

2-დან 5-მდე ნომრების მქონე რომელ მოწყობილობას გააჩნია დაკავებულობის დროის გაცილებით უფრო დიდი მნიშვნელობა? აღნიშნული მოწყობილობის ნომერი მოვათავსოთ პარამეტრში რომლის ნომერიც მითითებულია მე-2 პარამეტრის მნიშვნელობით.

დ) 

P2-ში მდებარე მნიშვნელობიდან P3-ში მდებარე მნიშვნელობამდე ნომრების მქონე რიგთაგან რომელს გააჩნია გაცილებით უფრო მცირე მიმდინარე შემცველობა? რიგის ნომერი მოვათავსოთ პარამეტრში, რომლის ნომერიც იმყოფება P1-ში.

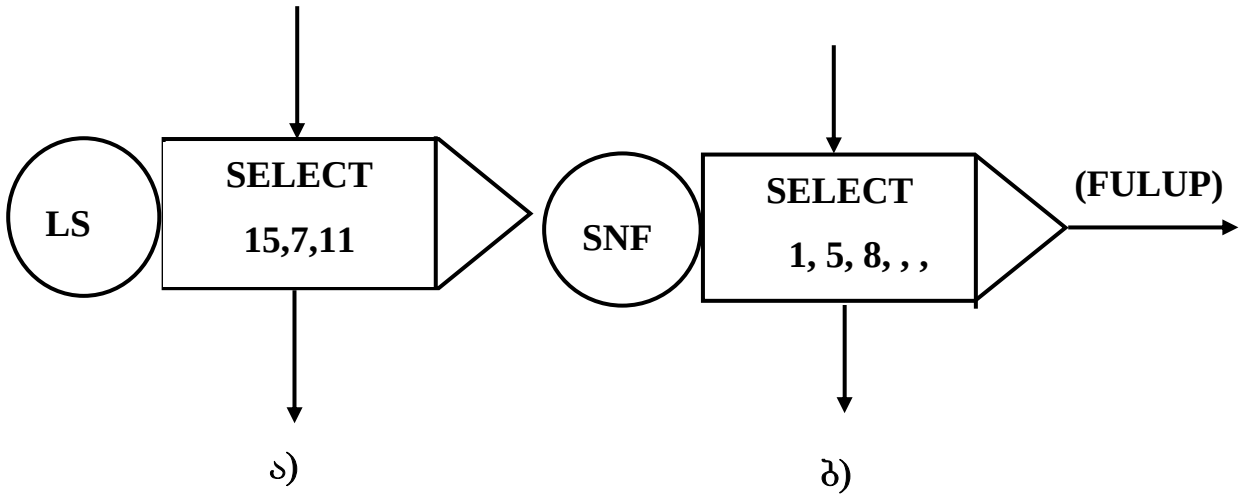
ნახ.5.6. **SELECT** ბლოკის
გამოყენების მაგალითები **MIN** ან **MAX** რეჟიმებში

5.6 ნახაზზე მოყვანილია **MIN** ან **MAX** რეჟიმებში **SELECT** ბლოკის გამოყენების ოთხი მაგალითი და მოცემულია განმარტება თითოეული ამ ბლოკის გამოსაყენებლად. აღვნიშნოთ, კერძოდ, რომ 5.6 ნახაზზე მოცემულ გ და დ მაგალითებში ნაჩვენებია **A** ოპერანდის მნიშვნელობათა ირიბი განსაზღვრებანი. გარდა ამისა, 5.6 ნახაზზე წარმოდგენილ დ მაგალითში გამოყენებულია **B** და **C** ოპერანდების მნიშვნელობათა ირიბი განსაზღვრა.

5.4.3. SELECT ბლოკის გამოყენება ლოგიკურ რეჟიმში

5.7 ნახაზზე მოყვანილია **SELECT** ბლოკის ლოგიკურ რეჟიმში გამოყენების ორი მაგალითი. ნახ. 5.7 ა ნახაზზე მითითებულია ლოგიკური მნიშვნელობები 7-დან 11-ის ჩათვლით, იმის გასარკვევად, იმყოფება თუ არა თუნდაც ერთ-ერთი მათგანი რეჟიმში „დაყენებულია.“ ლოგიკური გადამრთველის პირობის დამაკმაყოფილებელი ნომერი უნდა მოთავსდეს შემრჩევი ტრანზაქტის მე-15 პარამეტრში, რომელიც გადადის რიგის მიხედვით მომდევნო ბლოკში. თუკი 7-დან 11-ის ჩათვლით არსებულ ლოგიკურ გადამრთველთაგან არც ერთი არ იმყოფება „დაყენებულია“ რეჟიმში, ამ შემთხვევაში შემრჩევი ტრანზაქტის მე-15 ნომერი უტოლდება ნულს, ხოლო ამის შემდეგ ტრანზაქტი გადაინაცვლებს მომდევნო ბლოკში. ნახ. 5.7 ბ ნახაზზე მითითებულია მრავალარხიანი მოწყობილობები დიაპაზონში 5-დან 8-ის ჩათვლით მათ შორის თუნდაც ერთი შეუვსებელი მოწყობილობის აღმოჩენის მიზნით. თუ შეუვსებელი მრავალ-

არხიანი მოწყობილობა იქნა აღმოჩენილი, მისი ნომერი შეიტანება შემრჩევი ტრანზაქტის პირველ პარამეტრში, რომელიც შემდეგ გადადის მომდევნო ბლოკში. ხოლო იმ შემთხვევაში, თუ მრავალარხიანი მოწყობილობები 5-დან 8-ის ჩათვლით დიაპაზონიდან შევსებულია, მაშინ პირველი პარამეტრი რჩება უცვლელი, ხოლო შემრჩევი ტრანზაქტი გადადის **FULUP** სახელის მქონე ბლოკში.



ნახ.5.7. **SELECT** ბლოკის გამოყენების მაგალითები ლოგიკურ რეჟიმში: ა – ალტერნატიული გამოსასვლელი არაა განსაზღვრული; ბ – ალტერნატიული გამოსასვლელი განსაზღვრულია

5.5. ცხრილების გამოყენება GPSS World-ში

GPSS WORLD-ში ცხრილის გამოყენებისათვის აუცილებელია შემდეგი ორი მოქმედების შესრულება:

1) თითოეული ცხრილი, რომელიც მოდელში გამოიყენება წინასწარ უნდა იყოს აღწერილი;

2) თითეული აღწერილი ცხრილისათვის საჭიროა მნიშვნელობების განსაზღვრა.

5.5.1. სტატისტიკური ცხრილები

ერთ მოდელში შესაძლებელია რამდენიმე ცხრილის გამოყენება. თითოეული ცხრილი თავდაპირველად მის გამოყენებამდე უნდა იყოს აღწერილი. ცხრილის სახელი შესაძლებელია იყოს რიცხვითი ან სიმბოლური. მოდელირების პროცესში ტრანზაქტები მოდელის ერთი წერტილიდან მეორეში გადაადგილდებიან. ტრანზაქტის გადაადგილების დროები შესაძლებელია წარმოდგენილ იქნეს როგორც შემხვევითი რიცხვების გადაადგილების დროების მნიშვნელობათა (არგუმენტების) სიმრავლე. ამ სიმრავლის ელემენტები (შემთხვევითი რიცხვები) შესაძლოა მოხვდეს მნიშვნელობების რომელიმე დიაპაზონში. მნიშვნელობათა ამ დიაპაზონს ეწოდება *სიხშირული ინტერვალები* ანუ *სიხშირული კლასები.*

სტანდარტული რიცხვითი ატრიბუტების (ტრანზაქტის მოდელში ყოფნისა და მის თითოეულ ნაწილში დაყოვნების დრო, რიგის სიგრძე, მრავალარხიანი მოწყობილობის შემცველობა და ა.შ.) არგუმენტების საშუალო მნიშვნელობის, სტანდარტული გადახრის, დამოკიდებული სიხშირის განაწილებისათვის გამოიყენება **TABLE** ან **QTABLE** ელექტრონული ცხრილები.

TABLE ცხრილის აღწერის ბრძანებას გააჩნია შემდეგი ფორმატი:

Name TABLE A, B, C, D

ბრძანება განსაზღვრავს არგუმენტს, აგრეთვე სიხშირულ ინტერვალთა რიცხვს და სიგანეს, სადაც **Name** ჭდე განსაზღვრავს ცხრილის სახელს;

A ოპერანდით მოცემულია ცხრილის არგუმენტის სახელი, რომლის მნიშვნელობები ცხრილში ტაბულირდება. ეს ოპერანდი შესაძლოა იყოს სახელი, ფრჩხილებში ჩასმული გამოსახულება ან სტანდარტული რიცხვითი ატრიბუტი;

B ოპერანდით განისაზღვრება პირველი ინტერვალის ზედა საზღვარი;

C ოპერანდით სიხშირული (ყველა შუალედური) ინტერვალების სიგანე;

D ოპერანდით სიხშირული ინტერვალების საერთო რიცხვი (იგი აუცილებლად უნდა იყოს დადებითი მთელი რიცხვი);

B, C და **D** ოპერანდების მნიშვნელობები შესაძლებელია იყოს მთელი ტიპის კონსტანტები. **B** ოპერანდი შესაძლოა იყოს უარყოფითი რიცხვი, თავისთავად **C** და

D ოპერანდებიც იქნება უარყოფითი. მოდელირების პროცესში ცხრილის საზღვრების მნიშვნელობა და ინტერვალების რიცხვი არ შეიძლება შეიცვალოს.

მაგალითად:

VrRem TABLE TIME, 10, 5, 6

ცხრილის არგუმენტის სახელია **TIME** (ოპერანდი **A**). ცხრილის პირველი საზღვარია 10 (ოპერანდი **B**). ცხრილში შუალედური ინტერვალების სიგანე უდრის 5-ს (ოპერანდი **C**). ეს ნიშნავს, რომ მეორე საზღვარი იქნება 15 (ე.ი 10+5), მესამე
მესამე – 20 (ე.ი. 15+5) და ა.შ. უკანასკნელი საზღვარი იქნება 30. ნამდვილ მნიშვნელობათა ღერძი გაყოფილია ექვს თანამიმდევრულ ინტერვალად (ოპერანდი **D**). ეს ნიშნავს, რომ მარცხენა და მარჯვენა ინტერვალების გარდა არსებობს ოთხი შუალედი.

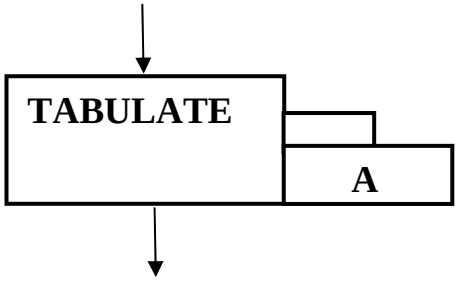
5.5.2. TABULATE ბლოკი

არგუმენტის მნიშვნელობები ცხრილში მოხვდება მას შემდეგ, რაც ტრანზაქტები შევა **TABULATE** ბლოკში, რომელსაც გააჩნია **TABLE** ბლოკში აღწერილი სახელი.

TABULATE ბლოკს გააჩნია შემდეგი ფორმატი:

TABULATE A , [B]

ხოლო გრაფიკულად შემდეგნაირად გამოისახება:



ნახ. 5.8. **TABULATE** ბლოკი **A** ოპერანდით

სადაც **A** ოპერანდი მიუთითებს ცხრილის სახელს (სიმბოლურს ან რიცხვითს), რომლის არგუმენტის მნიშვნელობები ცხრილში ტაბულირდება.

B ოპერანდით (წონითი კოეფიციენტი) მოცემულია ერთეულების რიცხვი. **B** ოპერანდის არ არსებობის შემთხვევაში მისი მნიშვნელობა იგულისხმება 1-ის ტოლად.

A და **B** ოპერანდები შესაძლოა იყოს სახელი, ფრჩხილებში ჩასმული გამოსახულება, სტანდარტული რიცხვითი ატრიბუტი ან სტანდარტული რიცხვითი ატრიბუტის პარამეტრი. ამასთან ერთად, **A** ოპერანდი შესაძლებელია იყოს მხოლოდ დადებითი მთელი რიცხვი, ხოლო **B** ოპერანდი – დადებითი რიცხვი. მაგალითად:

```
CXRILI  TABLE  P$ReaLvs, 7.2, 5.5, 10
...
TABULATE  CXRILI
```

TABLE ბრძანება აღწერილია **CXRILI** სახელით. ცხრილის არგუმენტები წარმოადგენს **P$ReaLvs** სტრა-ს პირველი ინტერვალის ზედა საზღვრით 7.2, სიგანით 5.5 და ინტერვალების რიცხვით 10. **P$ReaLvs** არგუმენტის თითოეული ტაბულირებული მნიშვნელობა ნაკლებია ან ტოლი 7.2, რომელიც ცხრილის სიხშირული კლასის სიხშირეს ზრდის ერთით. თუ ცხრილის არგუმენტი არ მოხვდა პირველ სიხშირულ კლასში, მაშინ ეს კლასი განისაზღვრება არგუმენტის მნიშვნელობის გაყოფით **TABLE** ბრძანების **C** ოპერანდზე. მაგალითად: **P$ReaLvs** ტოლია 30. 25. მაშინ 30. 25 / 5. 5 და მეექვსე სიხშირული კლასი

გაიზრდება 1 სიხშირით. თუ ცხრილის **A** არგუმენტი მეტია **B + C ≠ D = 8.2 + 5.5 ≠ 10 = 63.2,** მაშინ ბოლო (მეათე) სიხშირული კლასი 1 ერთეულით შეიცვლება.

მაგალითი 5.1. სარემონტო განყოფილებაში მომსახურების ერთი არხით შედის ორი ტიპის გაუმართავი კავშირის საშუალებები. I და II ტიპის კავშირის საშუალებები რემონტდება სარემონტო განყოფილებაში ერთი და იგივე ხელოსნის მიერ. I ტიპის კავშირის საშუალებების შესვლის დროის ინტერვალი განაწილებულია თანაბარალბათურად 20±10 სთ, ხოლო II ტიპის კავშირის საშუალებების – 15±8 სთ. შესული კავშირის საშუალებები რემონტდება შემდეგი პირობის მიხედვით: „პირველი მოვიდა – პირველი მომსახურდა“. I ტიპის კავშირის საშუალებების გარემონტებაზე იხარჯება 6±2 სთ, მეორე ტიპის – 8±4 სთ დრო.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს სარემონტო განყოფილების ფუნქციონირებას 3 დღე-ღამის განმავლობაში (72 სთ). დროის ერთეულია 1 წთ.

მოდელის შედგენამდე გაეცანით შემდეგ იდენტიფიკატორებს:

- **Rem1** – სარემონტო განყოფილება მომსახურების ერთი არხით;
- **RemQ** – გაუმართავი კავშირის საშუალებების საერთო რიგი რემონტში ყოფნის დროს;
- **RemQ1** – I ტიპის გაუმართავი კავშირის საშუალებების რიგი;

- **RemQ2** – II ტიპის გაუმართავი კავშირის საშუალებების რიგი;
- **VrRem** – ცხრილის სახელი, რომელშიც ტაბულირდება რემონტში გაუმართავი კავშირის საშუალებების ყოფნის საერთო დრო;
- **VrRem1** – ცხრილის სახელი, რომელშიც ტაბულირდება I ტიპის გაუმართავი კავშირის საშუალებების რემონტში ყოფნის დრო;
- **VrRem2** – ცხრილის სახელი, რომელშიც ტაბულირდება II ტიპის გაუმართავი კავშირის საშუალებების რემონტში ყოფნის დრო;

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*          ცხრილის განსაზღვრა

VrRem1 TABLE  M1, 420, 180, 5
VrRem2 TABLE  M1, 420, 180, 5
VrRem  TABLE  M1, 420, 180, 5

*          I ტიპის კავშირის საშუალებების რემონტის
                  იმიტაციის სეგმენტი

           GENERATE  600,300   ; I ტიპის კავშირის საშუალებების
                               წყარო
           QUEUE     RemQ      ; საერთო რიგში ჩადგომა
           QUEUE     RemQ1     ; I ტიპის კავშირის საშუალებების
                               რიგში ჩადგომა
           SEIZE     Rem1      ; სარემონტო განყოფილების
                               დაკავება
           DEPART    RemQ      ; საერთო რიგის დატოვება
           DEPART    RemQ1     ; I ტიპის კავშირის საშუალებების
                               რიგის დატოვება
```

```
          ADVANCE   360, 120  ; რემონტის იმიტაციაზე
                                დახარჯული დრო
          RELEASE   Rem1      ; სარემონტო განყოფილების
                                განთავისუფლება
          TABULATE  VrRem1    ; ცხრილი I ტიპის კავშირის
                                საშუალებების შესახებ
          TRANSFER  ,Wde1     ; I ტიპის გარემონტებული
                                კავშირის საშუალებები

*              II ტიპის კავშირის საშუალებების რემონტის
                        იმიტაციის სეგმენტი

          GENERATE  600, 300  ; II ტიპის კავშირის საშუალებების
                                წყარო
          QUEUE     RemQ      ; საერთო რიგში ჩადგომა
          QUEUE     RemQ2     ; II ტიპის კავშირის საშუალებების
                                რიგში ჩადგომა
          SEIZE     Rem1      ; სარემონტო განყოფილების
                                დაკავება
          DEPART    RemQ      ; საერთო რიგის დატოვება
          DEPART    RemQ2     ; II ტიპის კავშირის
                                საშუალებების რიგის დატოვება
          ADVANCE   360, 120  ; რემონტის იმიტაციაზე
                                დახარჯული დრო
          RELEASE   Rem1      ; სარემონტო განყოფილების
                                განთავისუფლება
          TABULATE  VrRem2    ; ცხრილი II ტიპის კავშირის
                                საშუალებების შესახებ
Wde1      TABULATE  VrRem     ; ცხრილი ორივე ტიპის კავშირის
                                საშუალებების შესახებ
          TERMINATE           ; ორივე ტიპის გარემონტებული
                                კავშირის საშუალებები

*                    მოდელირების დროის სეგმენტი

          GENERATE  4320      ; მოდელირების დრო
          TERMINATE 1         ; დასრულება
```

არსებობს ისეთი შემთხვევები, როდესაც შემთხვევითი სიდიდეების დასაკვირვებელი მნიშვნელობები დროის ერთსა და იმავე მომენტში ცხრილში შეტანილ უნდა იქნას არა ერთხელ, არამედ ორჯერ ან მეტი რაოდენობით. ამისათვის კი ერთ-ერთ ხერხს წარმოადგენს ცხრილში **TABULATE** ბლოკის ორჯერ გამოყენება. მსგავსი სიტუაციების აღწერისათვის გაცილებით უფრო მოხერხებულ ხერხს **GPSS World**-ში წარმოადგენს **TABULATE** ბლოკი არააუცილებელი ოპერანდით, რომელსაც როგორც ზემოთ იყო აღნიშნული ეწოდება *წონითი კოეფიციენტი.* ეს ოპერანდი გვიჩვენებს რიცხვს, თუ **TABULATE** ბლოკის შესრულების შემდეგ რამდენჯერ ტაბულირდება ცხრილში მოცემული სიდიდე. აქედან გამომდინარე, შესაძლებელია ერთ დასაკვირვებელ სიდიდეს დაუწესდეს გაცილებით უფრო მაღალი წონა, ვიდრე სხვას. ცხრილები, რომლებშიც წონის მიხედვით შესაძლებელია ამგვარი სხვაობის დაწესება უწოდებენ *წონით ცხრილებს.*

თუ **B** ოპერანდი არ გამოიყენება, მაშინ **TABULATE** ბლოკის შესრულებისას დაკვირვების ქვეშ მყოფი სიდიდე შედის შესაბამის ცხრილში მხოლოდ ერთხელ. **TABULATE** ბლოკი ძალზე ხშირად გამოიყენება **B** ოპერანდის გარეშე. ამ შემთხვევას ეწოდება *არააწონილი ცხრილი.* თუ ცხრილი წარმოადგენს წონითს, მაშინ აღნიშნული ცხრილის **D** ოპერანდის პირველი სიმბოლო უნდა იყოს **W** ასო. წონითი ცხრილის გამოყენება მიზანშეწონილია ისეთი შემთხვევითი სიდიდეების განხილვისას, როგორიცაა: „რიგის სიგრძე“ და „რიგში ყოფნის დრო“. რიგში ყოფნის დროის

განაწილების შესაფასებლად ასევე შესაძლებელია ცხრილის გამოყენება **QTABLE** რეჟიმში.

TABULATE ბლოკში **B** ოპერანდის რანგში წონითი კოეფიციენტის გამოყენების მაგალითი:

TABULATE CXRILI, 3

ამ შემთხვევაში უკანასკნელი დაემატება სიხშირული კლასის სიხშირის მნიშვნელობას. აგრეთვე, წონითი კოეფიციენტები გამოიყენება საშუალო და სტანდარტული გადახრის დროს რათა თანაბარალბათურად მოხდეს **TABULATE** ბლოკში ტრანზაქტების შესვლა. **TABLE** ბრძანება **TABULATE** ბლოკთან ერთად გამოიყენება ნებისმიერი სტრა-ს ტაბულაციისათვის.

TABLE ცხრილის გარდა შესაძლებელია **QTABLE** ცხრილის გამოყენება, რომლის საშუალებით შესაძლებელია მხოლოდ ტრანზაქტის რიგში ყოფნის დროის განაწილების ტაბულაცია. **QTABLE** ცხრილის აღწერის ფორმატი **TABLE** ცხრილის მსგავსია. განსხვავება მდგომარეობს იმაში, რომ **A** ოპერანდი მიუთითებს რიგის სახელს, ხოლო **B** და **C** ოპერანდების დანიშნულება იგივე გახლავთ რაც **TABLE** ბრძანებით აღწერის დროს. **B** ოპერანდი შესაძლებელია იყოს ნულის ტოლი ან დადებითი რიცხვი, ხოლო **D** ოპერანდი მიუთითებს თვით ცხრილის სახელს. მოდელში ასეთი ტიპის ცხრილის შექმნისას აუცილებელია იგი წინასწარ აღიწეროს **QTABLE** ბრძანების საშუალებით და მას გააჩნია შემდეგი ფორმატი:

Name QTABLE A, B, C, D

მაგალითად:

CXRILI QTABLE Sigrdze, 17.2, 4.3, 7

სადაც **Sigrdze** – მოწყობილობასთან რიგის სახელია და არა სტრა, როგორც **TABLE** ბრძანების შემთხვევაში.

QUEUE და **DEPART** ბლოკებში ტრანზაქტების გავლისას მისი გავლის დრო ფიქსირდება და ცხრილის სიხშირული ინტერვალის მრიცხველს სადაც მოხდა აღნიშნული დროის დაფიქსირება ემატება 1. ცხრილში ერთდროულად გროვდება ინფორმაცია ლოდინის დროის საშუალო მნიშვნელობის და საშუალო კვადრატული გადახრის გამოსათვლელად. უნდა აღინიშნოს, რომ **QTABLE** სბრძანების შემთხვევაში (ტრანზაქტის **QUEUE** და **DEPART** ბლოკებში შესვლისას) ინფორმაცია ცხრილში ავტომატურად შეიტანება და **TABULATE** ბლოკის გამოყენება არ არის მიზანშეწონილი. ორივე ბრძანების შემთხვევაში მოდელირების დამთავრებისას ცხრილში შეკრებილი ინფორმაცია გამოიტანება სტანდარტული ანგარიშის სახით.

VI თავი
მოდელების აგება არითმეტიკული ცვლადების გამოყენებით

არითმეტიკული ცვლადი **GPSS** ენაში წარმოადგენს მომხმარებლის მიერ განსაზღვრულ სტანდარტულ რიცხვით ატრიბუტს.

6.1. შენარჩუნებული უჯრედები

GPSS-ში გამოიყენება მეხსიერების უჯრედები, რომელთა საწყისი მნიშვნელობებიც შეიძლება მითითებულ იქნას მოდელირებამდე და რომელთაც მოდელირების დროს შეიძლება მივმართოთ მოდელის ნებისმიერი ადგილიდან. აღნიშნულ უჯრედებს ეწოდებათ *შენარჩუნებული უჯრედები.* ისინი წარმოადგენენ სტანდარტულ რიცხვით ატრიბუტებს (სტრა). შენარჩუნებული უჯრედების მნიშვნელობები იცვლება მხოლოდ მომხმარებლის პირდაპირი მითითების დროს. მოდელირების დაწყებამდე შენარჩუნებული უჯრედების მნიშვნელობები ტოლია ნულის.

6.1.1. INITIAL ბრძანება და SAVEVALUE ბლოკი

მომხმარებლის სურვილისამებრ ცალკეულ შენარჩუნებულ უჯრედებს **INITIAL** ბრძანების მეშვეობით შესაძლოა მიენიჭოთ არანულოვანი საწყისი მნიშვნელობები. **INITIAL** ბრძანება გამოიყენება აქტიური ტრანზაქტის უქონლობის დროს და იმ დროს, როდესაც არ შეიძლება გამოვიყენოთ ტრანზაქტების პარამეტრების მნიშვნელობები. მას გააჩნია შემდეგი ფორმატი:

INITIAL A, [B]

შენარჩუნებული უჯრედის ინიციალიზაციის დროს **A** ოპერანდი შეიძლება იყოს

Xდადებითი მთელი რიცხვი, **X$**სახელი. ანუ ბრძანებაში შეიძლება მიეთითოს სახელი ან ნომერი, თუმცა, როგორც ოდნავ მოგვიანებით შეიტყობთ ამის განხორციელება არ არის შესაძლებელი ყველა ბრძანებაში.

B ოპერანდი – არის მისანიჭებელი საწყისი მნიშვნელობა ან **UNCPECIFIED** (არ არის განსაზღვრული). იგი შეიძლება იყოს სახელი, რიცხვი, სტრიქონი ან **UNCPECIFIED.** თუ **B** ოპერანდი არ გამოიყენება, მაშინ უჯრედის მნიშვნელობა ტოლია 1-ის. მაგალითად:

INITIAL X32, 3895.56

INITIAL XDef, QPod

INITIAL X$Cost, Zena1

INITIAL X$Time, „Result Number“

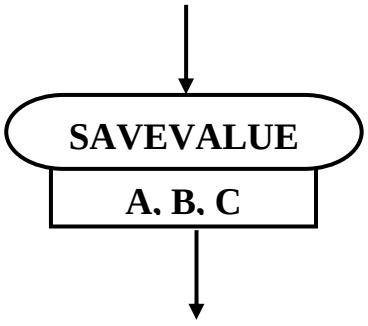
INITIAL X1

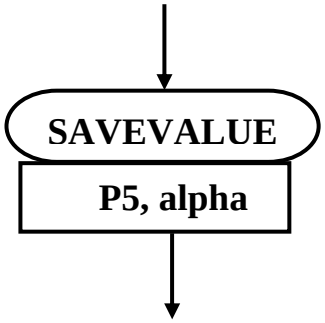
პირველ მაგალითში ნომერ 32 უჯრედში შეიტანება რიცხვი 3895.56, მეორეში – **Def** სახელით მითითებულ უჯრედში **- Pod** სახელის მქონე რიგის მიმდინარე სიგრძე, მესამეში – **Cost** სახელის მქონე უჯრედში – **Zena1** მომხმარებლის ცვლადის მნიშვნელობა, მეოთხეში – **Time** სახელის მქონე უჯრედში შეიტანება სტრიქონული კონსტანტა, ხოლო მეხუთეში – **B** ოპერანდი არ გამოიყენება, ამიტომ ნომერ 1 შენარჩუნებულ უჯრედს ენიჭება მნიშვნელობა 1.

მოდელირების პროცესში შენარჩუნებული უჯრედის მნიშვნელობა იცვლება **SAVEVALUE** ბლოკში ტრანზაქტის შესვლის დროს. **SAVEVALUE** ბლოკს გააჩნია შემდეგი ფორმატი:

SAVEVALUE A, B, [C]

გრაფიკულად **SAVEVALUE** ბლოკი ასე გამოისახება:





ნახ. 6.1. **SAVEVALUE** ბლოკი **A, B** და **C** ოპერანდებით

როდესაც ტრანზაქტი შედის **SAVEVALUE** ბლოკში, **B** ოპერანდში მდგარი სიდიდე იქცევა შენარჩუნებელი უჯრედის მნიშვნელობად, რომლის ნომერი ან სიმბოლური სახელი ჩაწერილია **A** ოპერანდში. მაგალითად, როგორც კი ტრანზაქტი შედის ამ ბლოკში, გამოითვლება **Alpha** ცვლადის მნიშვნელობა. მიღებული შედეგი ენიჭება შენარჩუნებულ უჯრედს, რომლის ნომერიც ჩაწერილია აქტიური ტრანზაქტის ნომერ 5 პარამეტრში. ამასთან, შენარჩუნებული უჯრედის ძველი მნიშვნელობა ნადგურდება. **A** ოპერანდი შეიძლება იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი, ხოლო **B** ოპერანდი გარდა ამისა შეიძლება იყოს რიცხვი ან სტრიქონი.

SAVEVALUE ბლოკი შეიძლება გამოყენებულ იქნას როგორც შეცვლის რეჟიმში, ასევე დაგროვების ან შემცირების რეჟიმებში. დაგროვების რეჟიმში შენარჩუნებული უჯრედის მომდევნო მნიშვნელობა იზრდება **B** ოპერანდში მდგარ მნიშვნელობამდე. შემცირების რეჟიმში კი – მცირდება **B** ოპერანდში მდგარ მნიშვნელობამდე. დაგროვების ან შემცირების რეჟიმები

განისაზღვრება **A** და **B** ოპერანდების გამყოფი მძიმის წინ + ან – შესაბამისი ნიშნის შემოღებით. მაგალითად:

```
SAVEVALUE  5+, X2
SAVEVALUE  Vad-, V$Hdl
SAVEVALUE  Ais, -659
SAVEVALUE  3, (25. 75#X$Num1+Q4)
SAVEVALUE  11, „Result number…“
```

A ოპერანდების მნიშვნელობები იცვლება მხოლოდ აღნიშნულ ბლოკებში ტრანზაქტების შესვლის დროს. პირველ მაგალითში, ნომერი 5 შენარჩუნებული უჯრედის მნიშვნელობა ტრანზაქტის **SAVEVALUE** ბლოკში შესვლის დროს იზრდება ნომერი 2 შენარჩუნებული უჯრედის მნიშვნელობამდე. მეორე მაგალითში, **Vad** სახელით ცნობილი შენარჩუნებული უჯრედის მნიშვნელობა მცირდება **Hdl** არითმეტიკული ცვლადის გამოთვლილ მნიშვნელობამდე. მესამე მაგალითში **Ais** სახელით ცნობილი შენარჩუნებული უჯრედის მნიშვნელობა შეიცვლდება რიცხვით -659. მეოთხე მაგალითში გამოითვლება ფრჩხილებში ჩასმული გამოსახულება და ენიჭება ნომერ 3 უჯრედს. მეხუთე მაგალითში ნომერი 11 უჯრედის შემცველობა შეიცვლება სტრიქონით. თუკი ასეთი სახელის მქონე უჯრედი არ არსებობს, იგი იქმნება.

შენარჩუნებულ უჯრედებს გააჩნიათ ერთადერთი სტრა**X** სახელით, რომლის მნიშვნელობასაც წარმოადგენს შესაბამისი შენარჩუნებული უჯრედის მიმდინარე მნიშვნელობა. როგორც მაგალითებიდან ჩანს, **SAVEVALUE** ბლოკში სტრა**X A** ოპერანდის

მითითების დროს არ გამოიყენება. სხვა ოპერანდების ჩაწერის დროს მითითება სტრა**X**-ზე აუცილებელია. აღნიშნულ სტრა-ზე მითითება გამოიყენება აგრეთვე გამოსახულებებში, ფუნქციებში და **INITIAL** ბრძანებებში.

6.1.2. შენარჩუნებული უჯრედის მატრიცები. **MATRIX** ბრძანება, **INITIAL** ბრძანება და **SAVEVALUE** ბლოკი

შენარჩუნებული უჯრედის ცნება **GPSS**-ში გამოიყენება მატრიცებთან მიმართებაშიც. ეს ობიექტები საშუალებას იძლევა მოწესრიგებულ იქნას მნიშვნელობები, რომლებიც უნდა შევინარჩუნოთ მატრიცების სახით. მატრიცა წინასწარ უნდა იქნას აღწერილი. მატრიცის აღწერის ბრძანებას გააჩნია შემდეგი ფორმატი:

Name MATRIX A, B, C, [D], [E], [F], [G]

Name ჭდე განსაზღვრავს მატრიცის სახელს და უნდა იყოს სახელი (არ შეიძლება იყოს რიცხვი). **A** ოპერანდი არ გამოიყენება, ვინაიდან დატოვებულია **GPSS**-ის გაცილებით უფრო მოძველებულ ვერსიებთან შეთავსების მიზნით. **B, C, D, E, F, G** ოპერანდები შეიძლება იყოს მხოლოდ მთელი დადებითი რიცხვები.

B ოპერანდის მიერ წესდება მატრიცის სტრიქონების რიცხვი – ელემენტების მაქსიმალური რაოდენობა პირველ განზომილებაში,

C ოპერანდის მიერ – მატრიცის სვეტების რიცხვი – ელემენტების მაქსიმალური რაოდენობა მეორე განზომილებაში, **D, E, F, G** ოპერანდები – აწესებენ ელემენტების მაქსიმალურ რაოდენობას შესაბამისად მესამე, მეოთხე, მეხუთე და მეექვსე განზომილებებში. მაგალითად, ბრძანება

TraSr MATRIX , 5, 7

განსაზღვრავს მე-5 სტრიქონის და მე-7 სვეტის შემცველ **TraSr** სახელის მქონე მატრიცას.

მატრიცებს გააჩნიათ ერთადერთი სტრა**MX** სახელწოდებით, რომლის მეშვეობითაც შესაძლებელია მატრიცის ნებისმიერ ელემენტთან მიმართვა. სტრიქონების და სვეტების მნიშვნელობათა სახით შესაძლებელია გამოყენებულ იქნას მხოლოდ სახელები, მთელი რიცხვები და ტრანზაქტების პარამეტრები. მაგალითად:

MX7(13, 5)
MX$Tab2(P4, P$Nal)
MX$Spis (User, P3)

პირველ მაგალითში განისაზღვრება მე-13 სტრიქონის და მე-5 სვეტის გადაკვეთაზე მყოფი ნომერი 7 მატრიცის ელემენტი. მეორე მაგალითში **Tab2** სახელის მქონე მატრიცის ელემენტის განსაზღვრისათვის გამოიყენება ტრანზაქტის ნომერი 4 პარამეტრის და **Nal** სახელის მქონე პარამეტრის მნიშვნელობა, ხოლო მესამე მაგალითში – მომხმარებლის ცვლადის სახელი და

ტრანზაქტის ნომერი 3 პარამეტრი. მომხმარებლის ცვლადს **EQU** ბრძანებით წინასწარ უნდა მიენიჭოს მთელი დადებითი მნიშვნელობა.

ყურადღება უნდა მიექცეს შემდეგ გარემოებას. პირველ მაგალითში მატრიცას გააჩნია არა სახელი, არამედ ნომერი, თუმცა როგორც ზემოთ იქნა აღნიშნული, მატრიცის განსაზღვრის ბრძანებაში **Name** ჭდის სახით შესაძლოა გამოყენებულ იქნას მხოლოდ სახელი. მატრიცისათვის სახელის ნაცვლად ნომრის მისანიჭებლად მატრიცის განსაზღვრის ბრძანების ჩაწერამდე გამოყენებულ უნდა იქნას **EQU** ბრძანება. მაგალითად:

Skl EQU 7
Skl MATRIX , 10, 7

მოცემულ მაგალითში მატრიცის **Skl** სახელი შეცვლილია ნომერი 7-ით, რომლის მიხედვითაც ამჟამად არის **MX7** და უნდა მივმართოთ მატრიცას. თუ **MATRIX** ბრძანებაში **Skl** სახელის ნაცვლად ჩაწერილი იქნება რიცხვი 7, ტრანსლაციის დროს წარმოებს შეჩერება შეცდომის მიხედვით – „გამოტოვებულია ოპერატორის ჭდე“.

მოდელირების ან **CLEAR ON** ბრძანების შესრულების დაწყებამდე მატრიცის ყველა ელემენტის მნიშვნელობები ნულის ტოლია. მოდელის შემქმნელის (პროგრამისტის) სურვილისამებრ ცალკეულ ან ყველა ელემენტს შეიძლება მიენიჭოს არანულოვანი მნიშვნელობა, ან **INITIAL** ბრძანებით ისინი შესაძლოა გადაყვანილ იქნას განუსაზღვრელ მდგომარეობაში. მაგალითად:

```
INITIAL   MX5 (2, 6), -648. 237
INITIAL   MX$Ntf (4, 7), 48
INITIAL   MX$Avt (13, 24), Koef
INITIAL   Gros, 48
INITIAL   Pult
INITIAL   MainResult, UNCPECIFIED
```

პირველ მაგალითში მეორე სტრიქონისა და მეექვსე სვეტის გადაკვეთაზე მყოფ ნომერი 5 მატრიცის ელემენტს ენიჭება საწყისი მნიშვნელობა -648. 237, მეორე მაგალითში — **Ntf** სახელის მქონე მატრიცის (4, 7) ელემენტს — მნიშვნელობა 48, მესამე მაგალითში — **Avt** სახელის მქონე მატრიცის (13, 24) ელემენტს — **Koef** მომხმარებლის ცვლადის მნიშვნელობა. მეოთხე მაგალითში ოპერანდის ნაცვლად მითითებულია **Gros** მატრიცის სახელი, ამიტომ აღნიშნული მატრიცის ყველა ელემენტს ენიჭება მნიშვნელობა 48. მეხუთე მაგალითში **A** ოპერანდის ნაცვლად ასევე დგას **Pult** მატრიცის სახელი, მაგრამ **B** ოპერანდი არ გამოიყენება, ამიტომ მატრიცის ყველა ელემენტს ენიჭება მნიშვნელობა 1. მეექვსე მაგალითში **B** ოპერანდის ნაცვლად ჩაწერილია საკვანძო სიტყვა **UNCPECIFIED**, ამიტომ წისასწარ შექმნილი **MainResult** მატრიცის ყველა ელემენტი გადაყვანილი იქნება განუსაზღვრელ მდგომარეობაში. აღნიშნული გადაყვანით მატრიცა მზად არის ექსპერიმენტში გამოსაყენებლად, რომელიც შეიძლება იყოს გამოტოვებული, ანუ ეს გახლავთ ექსპერიმენტის მსვლელობის დროს მიუღებელი მონაცემები. მატრიცის ელემენტის განუსაზღ-

ვრელი მდგომარეობა მიუთითებს იმაზე, რომ მონაცემები არ არის და თუ მატრიცის ელემენტები გადაყვანილი იქნებოდა ნულში, ეს აღიქმებოდა როგორც ნულის ტოლი საექსპერიმენტო მონაცემების მიღება, რაც არ შეესაბამება სინამდვილეს.

ექსპერიმენტის ჩატარების შემდეგ მატრიცა მიღებული შედეგებით გადაეცემა დისპერსიული ანალიზის ბიბლიოთეკურ **(ANOVA)** პროცედურას, რომელიც ამუშავებს **UNCPECIFIED** მდგომარეობაში მყოფ ელემენტებს, როგორც ნულოვანი მნიშვნელობების ნაცვლად გამოტოვებულ მონაცემებს.

MATRIX ოპერატორი მიმდინარე მოდელში ქმნის მატრიცას. მატრიცა არ შეიძლება წაშლილ იქნას მიმდინარე მოდელიდან. თუ **MATRIX** ოპერატორი წაშლილია შესასრულებელი მოდელიდან, მაშინ მიმდინარე მოდელში მატრიცასთან კავშირი რჩება.

მატრიცა შესაძლებელია განმეორებით იქნას განსაზღვრული იმავე ჭდის მქონე **MATRIX** ოპერატორის მიერ. გამოყენების წინ მატრიცა უნდა განისაზღვროს **MATRIX** ოპერატორში. მატრიცა შეზღუდულია მეხსიერების მაქსიმალური მოცულობით, რომელიც დგინდება მოდელის აგებათა ჟურნალის მიერ.

მოდელირების პროცესში მატრიცებში მნიშვნელობათა ჩაწერასა და აგრეთვე მატრიცებში ჩაწერილ მნიშვნელობათა შემცირებას ან (და) გაზრდას ემსახურება **MSAVEVALUE** ბლოკი. მისი ჩაწერის ფორმატს გააჩნია შემდეგი სახე:

MSAVEVALUE A, B, C, D

A ოპერანდში წესდება მატრიცის სახელი ან ნომერი. **A** ოპერანდის მარჯვენა განაპირა (კიდური) სიმბოლო შეიძლება იყოს + ნიშანი (დაგროვების რეჟიმი) ან – ნიშანი (გამოკლების რეჟიმი). **B** ოპერანდში წესდება მატრიცის სტრიქონის ნომერი, ხოლო **C** ოპერანდში კი სვეტის ნომერი. **D** ოპერანდი განსაზღვრავს მნიშვნელობას, რომლითაც უნდა ინახებოდეს, მას ემატებოდეს ან (და) აკლდებოდეს. ყველა ოპერანდი თავისთავად წარმოადგენს აუცილებლობას. ისინი შეიძლება იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა, სტრა*პარამეტრი. მაგალითად:

MSAVEVALUE 2, 13, P21, 782. 34

MSAVEVALUE X$Bba-, 8, 23, V$Hit

MSAVEVALUE Rta+, 11, (User# 2+3), M1

MSAVEVALUE (Rra+Q$Rem) +, (Rra-5), 12, P$Nht

MSAVEVALUE 4, Striqoni, Sveti “Sigdrze Mars”

პირველ მაგალითში **MSAVEVALUE** ბლოკში ტრანზაქტის შესვლის დროს მე-13 სტრიქონსა და პარამეტრ 21-ში შემავალი ტრანზაქტის შემადგენლობაში მყოფ სვეტის გადაკვეთაზე განლაგებული ნომერი 2 მატრიცის ელემენტის მნიშვნელობა იცვლება რიცხვით 782. 34.

მეორე მაგალითში მე-8 სტრიქონისა და 23-ე სვეტის გადაკვეთაზე მყოფი მატრიცის ელემენტის მნიშვნელობა, რომლის ნომერსაც შეიცავს **Bba** სახელის მქონე შენარჩუნებული უჯრედი

მცირდება **Hit** არითმეტიკული ცლადის გამოთვლილი მნიშვნელობით.

მესამე მაგალითში სვეტის ნომერი იმყოფება როგორც **(User # 2+3)** ფრჩხილებში ჩასმული გამოსახულების გამოთვლის შედეგი, ხოლო შემდეგ ამგვარად განსაზღვრული **Rta** მატრიცის ელემენტის მნიშვნელობა იზრდება მოდელში ტრანზაქტის ყოფნის **M1** ფარდობითი დროის სიდიდით.

მეოთხე მაგალითში მატრიცის ნომერი და ელემენტის სტრიქონის ნომერი განისაზღვრება როგორც **(Rra + Q$Rem)** და შესაბამისად **(Rra-5)** ფრჩხილებში ჩასმულ გამოსახულებათა გამოთვლის შედეგები. ამის შემდეგ ნაპოვნი ელემენტის მნიშვნელობა იზრდება **Nht** სახელის მქონე აქტიური ტრანზაქტის პარამეტრში არსებული სიდიდით.

მეხუთე მაგალითში მომხმარებლის **Striqoni** და **Sveti** ცვლადებით განსაზღვრულ, მე-4 სტრიქონისა და სვეტის გადაკვეთაზე არსებულ მატრიცის ელემენტს ენიჭება სტრიქონი „**Sigrdze Mars**“.

თუკი მომხმარებლის ცვლადებს **EQU** ბრძანებით წინასწარ არ მიენიჭებათ შესაბამისი მნიშვნელობები, წარმოიშვება „შეჩერება შეცდომის დროს“.

„შეჩერება შეცდომის დროს“ წარმოიშობა იმ შემთხვევაში, თუ **SAVEVALUE** ბლოკში ტრანზაქტის შესვლის დროს არ იქნება აღმოჩენილი მატრიცა საჭირო სახელით ან ნომრით, ან არ აღმოჩნდება სტრიქონისა და სვეტის შესაბამისი ნომრების მქონე ელემენტები.

SAVEVALUE ბლოკში სტრა**MX A** ოპერანდის მითითების დროს, ისევე როგორც სტრა**X** ანალოგიურ ბლოკში იმავე ოპერანდის მითითებისას როგორც წესი – არ გამოიყენება. მითითება აღნიშნულ სტრა-ზე ასევე გამოიყენება გამოსახულებებში, ფუნქციებში, **INITIAL** ოპერატორში.

მატრიცის აღწერის ბრძანების ზემოთ განხილული ფორმატიდან გამომდინარეობს, რომ **GPSS**-ში მატრიცას შეუძლია გააჩნდეს ექვსამდე განზომილება. თუმცა როგორც **MSAVEVALUE** ბლოკის ფორმატიდან გამომდინარეობს, იგი საშუალებას იძლევა გამოყენებულ იქნას მხოლოდ ერთი ან ორგანზომილებიანი მატრიცები, ანუ მასში ხელმისაწვდომია მხოლოდ პირველი ორი განზომილება. დანარჩენი ინდექსები (მაჩვენებლები) უდრის ერთს.

თუკი აუცილებელია გამოყენებულ იქნას ორზე გაცილებით მეტი განზომილების მქონე მატრიცა, საჭიროა შეიქმნას ერთი ან გაცილებით უფრო მეტი **PLUS**-პროცედურა. მათ გაჩნიათ დაშვება მატრიცების ყველა ელემენტებთან.

MATRIX ბრძანებით განსაზღვრული მატრიცები წარმოადგენს გლობალურ მატრიცებს და დასაშვებია თითქმის ყველა **PLUS**-პროცედურისათვის. **PLUS**-პროცედურის შესრულების დროს პარალელურად შესაძლებელია შექმნილ იქნეს ხილვადობის ლოკალური არის მქონე დროებითი მატრიცები.

VII თავი

სისტემების მოდელების აგება ერთარხიანი მოწყობილობებისათვის წყვეტისა და მიუწვდომლობის რეჟიმებში ფუნქციონირებისას

7.1. ერთარხიანი მოწყობილობის წყვეტა. **PREEMPT** და **RETURN** ბლოკები

თუ ერთარხიანი მოწყობილობის შესასვლელზე არსებობსრიგი, მაშინ მისი განთავისუფლების შედეგად წარმოებს ტრანზაქტების შერჩევა მის დასაკავებლად:

- **FIFO** შესვლის თანრიგში – თანაბარი პრიორიტეტით;
- **GENERATE** ბლოკში **E** ოპერანდით მითითებული პრიორიტეტის გათვალისწინებით.

მაღალი პრიორიტეტის მქონე მორიგი ტრანზაქტი მისი პრიორიტეტის მიუხედავად ელოდება წინა ტრანზაქტის მომსახურების დასრულებას. პრიორიტეტი გათვალისწინებულია მხოლოდ საცდელი რიგის შემთხვევაში. მასში ტრანზაქტები დგებიან პრიორიტეტულ თანრიგში.

ერთარხიანი მოწყობილობა ასეთ რეჟიმში ფუნქციონირებს მეორე თავში განხილული **SEIZE** და **RELEASE** ბლოკების ორგანიზებით.

ხშირად შესაძლებელია ისეთი შემთხვევების მოდელირება, როდესაც მორიგმა ტრანზაქტმა შესაძლებელია დაიკავოს ერთარხიანი მოწყობილობა მას შემდეგ, რაც წინა ტრანზაქტმა შეწყვიტა მომსახურება. ასეთ შეწყვეტას ეწოდება ერთარხიანი მოწყობილობის „დაკავება“ და მოდელირდება **PREEMPT** (დაკავება, დაპყრობა) ბლოკით. მას გააჩნია ჩაწერის შემდეგი ფორმატი:

PREEMPT A, [B], [C], [D], [E]

სადაც **A** ოპერანდი არის დაკავებული ერთარხიანი მოწყობილობის სახელი ან ნომერი. როდესაც ერთარხიანი მოწყობილობა თავისუფალია, მაშინ **PREEMPT** ბლოკი ისევე მუშაობს როგორც **SEIZE** ბლოკი. თუ ერთარხიანი მოწყობილობა დაკავებულია მაშინ **PREEMPT** ბლოკი ფუნქციონირებს ან პრიორიტეტულ რეჟიმში ან წყვეტის რეჟიმში.

B ოპერანდით განისაზღვრება რეჟიმები:

- **PR –** პრიორიტეტული;
- ფარულად (როდესაც **B** ოპერანდი არ გამოიყენება) – წყვეტის.

პრიორიტეტულ რეჟიმში ფუნქციონირებისას **B** ოპერანდი ტოლია **PR. C** ოპერანდი არ გამოიყენება, ანუ შეწყვეტილი ტრანზაქტი მომდევნო ბლოკში არ გადავა. ამიტომ შეწყვეტის

შემდეგ იგი გარკვეული დროით ავტომატურად შეინახება ერთარხიანი მოწყობილობის დაყოვნების სიაში.

პრიორიტეტულ რეჟიმში წინა მომსახურებული ტრანზაქტის მომსახურების შეწყვეტა ანუ ერთარხიანი მოწყობილობის „დაკავება“ შესაძლებელია მხოლოდ იმ ტრანზაქტისათვის, რომელსაც გააჩნია მაღალი პრიორიტეტი. თუ ერთარხიანი მოწყობილობის დაკავების პრეტენდენტი ტრანზაქტის პრიორიტეტი ტოლია ან დაბალია (ნაკლებია) მომსახურებული ტრანზაქტის პრიორიტეტზე, ის მოთავსდება ერთარხიანი მოწყობილობის სიის ზოლში თავისი პრიორიტეტის გათვალისწინებით.

იმ ტრანზაქტებისათვის რომელთა მომსახურება შეწყვეტილია განისაზღვრება **C, D** და **E** ოპერანდებით:

C ოპერანდი – იმ ბლოკის სახელი ან ნომერი სადაც უნდა გადავიდეს შეწყვეტილი ტრანზაქტი.

E ოპერანდი – **RE** მნიშვნელობისას განსაზღვრავს შეწყვეტილი ტრანზაქტის წაშლას.

D ოპერანდი – შეწყვეტილი ტრანზაქტის პარამეტრის ნომერი, რომელშიც ჩაიწერება მომსახურების დასრულებამდე დარჩენილი დრო.

A, B, C და **D** ოპერანდები შესაძლებელია იყოს დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი.

დაკავებული ერთარხიანი მოწყობილობის გასანთავისუფლებლად გამოიყენება **RETURN** (დაბრუნების) ბლოკი. მისი ჩაწერის ფორმატს გააჩნია შემდეგი სახე:

RETURN A

სადაც **A** ოპერანდი არის განთავისუფლებული ერთარხიანი მოწყობილობის სახელი ან ნომერი. მაგალითად:

RETURN Rem1

აღნიშნულ მაგალითში **Rem1** არის დაკავებული ერთარხიანი მოწყობილობა.

7.1.1 პრიორიტეტულ რეჟიმში წყვეტა

მაგალითი 7.1. განვიხილოთ **PREEMPT** ბლოკის ფუნქციონირება პრიორიტეტულ რეჟიმში, ანუ **B** ოპერანდს გააჩნია **PR** პრიორიტეტი. (**C** ოპერანდი არ გამოიყენება, ანუ შეწყვეტილი ტრანზაქტი არ მიემართება სხვა ბლოკისაკენ. ამიტომ, შეწყვეტის შედეგად მომსახურების დასრულებამდე ის მოთავსდება ერთარხიანი მოწყობილობის დაყოვნების სიაში).

აღნიშნული მაგალითის შესაბამისი მოდელი შემდეგნაირად გამოიყურება:

```
*                    GPSS WORLD SIMULATION

            GENERATE   5, , , 3                ;ტრანზაქტების წყარო
            PRIORITY   XN1   ; ნომრის თანახმად პრიორიტეტის მინიჭება
            PREEMPT    Rem1, PR      ; პრიორიტეტულ რეჟიმში ეამ-ის
                                        დაკავება
            ADVANCE    11                  ; მომსახურება
            PREEMPT    Rem1                ; ეამ-ის განთავისუფლება
            TEST  E    PR, 1, Wde2         ; პრიორიტეტი უდრის 1-ს?
Wde1        TERMINATE  1                   ; დიახ, მაშინ გაანადგურეთ
Wde2        TEST  E    PR, 2, We4          ; პრიორიტეტი უდრის 2-ს?
Wde3        TERMINATE  1                   ; დიახ, მაშინ გაანადგურეთ
Wde4        TERMINATE  1                   ; მესამე პრიორიტეტის მქონე
                                             ტრანზაქტის განადგურება
```

GENERATE ბლოკი სამოდელო დროის მომენტში გენერირებს მხოლოდ სამ ტრანზაქტს 5, 10 და 15 (აღვნიშნოთ **t**-თი) და წყვეტს აქტიურობას. ყოველ მორიგ ტრანზაქტს გააჩნია გაცილებით მაღალი პრიორიტეტი, ვიდრე წინას შესაბამისად, 1, 2 და 3 **SAVEVALUE** და **PRIORITY** ბლოკების გამოყენებით.

პირველი ტრანზაქტი 1 პრიორიტეტით **t** = 5 მომენტში იკავებს **Rem1** ერთარხიან მოწყობილობას. **t** = 10 მომენტში გენერირდება მეორე ტრანზაქტი 2 პრიორიტეტით. ის წყვეტს პირველი ტრანზაქტის მომსახურებას რომელსაც მომსახურებების დასრულებამდე დარჩა 5 + 11 - 10 = 6 სამოდელო დროის

ერთეული. პირველი ტრანზაქტი **C** ოპერანდის არ არსებობისას მოთავსდება **Rem1** ერთარხიანი მოწყობილობის დაყოვნების სიაში. **t** = 5 მომენტში გენერირდება მესამე ტრანზაქტი 3 პრიორიტეტით. ის წყვეტს მეორე ტრანზაქტის მომსახურებას, რადგან მას გააჩნია გაცილებით დაბალი პრიორიტეტი (**PR** = 2). მომსახურების დასრულებამდე მეორე ტრანზაქტს დარჩა 10 + 11 – 15 = 6 სამოდელო დროის ერთეული. აგრეთვე ისიც მოთავსდება **Rem1** ერთარხიანი მოწყობილობის დაყოვნების სიაში პირველი ტრანზაქტის წინ, რადგან მისი პრიორიტეტი დაბალია და უდრის 1-ს.

ამგვარად, პრიორიტეტის შესაბამისად მომსახურდება მესამე, მეორე და პირველი ტრანზაქტები. ამის გაკეთება შესაძლებელია თუ შესრულდება ბიჯით მოდელირება: **TERMINATE** ბლოკში ტრანზაქტები თანამიმდევრობით შევლენ **Wde1, Wde2** და **Wde3** მითითებული ჭდეებით შესაბამისად. „მომსახურებამდე“ დროის შეწყვეტისას შეწყვეტილი ტრანზაქტი ავტომატურად დამახსოვრდება და შემდეგ მომსახურების აღდგენისას გამოიყენება. მოცემულ მაგალითში ამის გაკეთება შესაძლებელია სამივე ტრანზაქტის მომსახურების დროის ჯამის მიხედვით: 5 + 11 + (5 + 6) + (5 + 6) = 38 სამოდელო დროის ერთეული. მესამე ტრანზაქტის მომსახურება დასრულდება **t** = 15 + 11 = 26, მეორე **t** = 26 + 6 = 32, პირველი **t** = 32 + 6 = 38 სამოდელო დროის ერთეულებში.

7.1.2. ერთარხიანი მოწყობილობის მდგომარეობის შემოწმება პრიორიტეტულ რეჟიმში ფუნქციონირებისას

ერთარხიანი მოწყობილობის მდგომარეობის შემოწმება პრიორიტეტულ რეჟიმში მიმდინარეობს **GATE** ბლოკით, აგრეთვე ბულის ცვლადების გამოყენებით და **TEST** ბლოკით.

განვიხილოთ ერთარხიანი მოწყობილობის მდგომარეობის შემოწმება **GATE** ბლოკით.

პირობის შემოწმება მიმდინარეობს ერთ-ერთი შემდეგი **X** პირობითი ოპერატორის მეშვეობით:

- **I – A** ოპერანდით მითითებული ერთარხიანი მოწყობილობის ფუნქციონირება შეწყვეტილია;
- **NI – A** ოპერანდით მითითებული ერთარხიანი მოწყობილობის ფუნქციონირება არ არის შეწყვეტილი.

მაგალითად:

```
GATE   I      Stan
GATE   NI     (V$Ras - 3)
GATE   I      Print, Wde
```

პირველ მაგალითში **GATE** ბლოკი გაატარებს ტრანზაქტს, როდესაც **Stan** ერთარხიანი მოწყობილობის ფუნქციონირება შეწყვეტილი იქნება. მეორე მაგალითში, ტრანზაქტი გადაადგილდება მომდევნო ბლოკში, როდესაც ერთარხიანი მოწყობილობის ფუნქციონირება შეწყვეტილი არ იქნება, რომლის

ნომერიც განისაზღვრება, როგორც ფრჩხილებში ჩასმული გამოსახულების **(V$Ras - 3)** გამოთვლის და მისი მთელ რიცხვამდე დამრგალების შედეგი. მესამე მაგალითში **Print** ერთარხიანი მოწყობილობის ფუნქციონირების შეწყვეტის შემთხვევაში ტრანზაქტი გადაადგილდება **Wde** ჭდით მითითებულ ბლოკში.

პირველ და მეორე მაგალითებში პირობის შეუსრულებლობის შემთხვევაში **GATE** ბლოკი შესასვლელში მუშაობს მტყუნების რეჟიმში. აქვე უნდა აღინიშნოს ისიც, რომ **B** ოპერანდის არარსებობის დროს შესაძლებელია მოდელირების მანქანური დროის გაზრდა. თუმცა გარკვეულ შემთხვევებში ასეთი რეჟიმი შესაძლოა გამოყენებულ იქნას.

7.2 ერთარხიანი მოწყობილობის მიუწვდომლობა

ერთარხიანი მოწყობილობის გაუმართავობისა და სხვა სიტუაციათა მოდელირებისათვის **GPSS World**-ში გამოიყენება ბლოკები, რომელთა საშუალებით შესაძლებელია ერთარხიანი მოწყობილობის მისაწვდომობის და მიუწვდომლობის რეალიზება. აღნიშნული ბლოკების გამოყენებისას ერთარხიანი მოწყობილობის სტატისტიკა არ დამახინჯდება (ანუ სხვაგვარად რომ ვთქვათ ხელი არ შეეშლება). მაგალითად, გაუმართავობის მოდელირებისათვის შეიძლება გამოყენებულ იქნას წყვეტის რეჟიმი (**PREEMPT**). ამასთან ერთად, ტრანზაქტები რომლებიც იწვევენ წყვეტას (ერთარხიანი მოწყობილობის უარის იმიტაცია)

მონაწილეობენ სტატისტიკაში ისევე როგორც მომსახურებადი ტრანზაქტები ერთარხიანი მოწყობილობის ფუნქციონირებისას. ეს კი არ არის მართებული, ვინაიდან მათი გავლენა იწვევს ერთარხიანი მოწყობილობის სტატისტიკის დამახინჯებას.

7.2.1. მიუწვდომლობის მდგომარეობაში გადასვლა და მისაწვდომლობის აღდგენა

ერთარხიანი მოწყობილობის მიუწვდომლობა მოდელირდება **FUNAVAIL** (**F** – სიმბოლო აღნიშნავს ერთარხიან მოწყობილობას, **UNAVAIL** – მიუწვდომლობას) ბლოკის საშუალებით. ამ ბლოკის გამოყენებით ერთარხიანი მოწყობილობის სტატისტიკა არ მახინჯდება. მის ფორმატს გააჩნია შემდეგი სახე:

FUNAVAILA, [B], [C], [D], [E], [F], [G], [H]

აღნიშნული ბლოკი უზრუნველყოფს ერთარხიანი მოწყობილობის მიუწვდომლობას, რომლის სახელი ან ნომერი მითითებულია **A** ოპერანდით. **FUNAVAIL** ბლოკის მიერ დამუშავებული ყველა ტრანზაქტი იყოფა სამ კლასად, რომლებიც განსაზღვრავს ოპერანდების დანიშნულებას:

- ტრანზაქტები, რომლებიც იკავებს ერთარხიან მოწყობილობას (**SEIZE** და **PREEMPT**) მისი მიუწვდომელ მდგომარეობაში გადაყვანის მომენტში (**B, C** და **D** ოპერანდები);
- უკვე შეწყვეტილი ტრანზაქტები იმყოფებიან შეწყვეტის სიაში (**E** და **F** ოპერანდები);
- ტრანზაქტები იმყოფებიან ჩამოშორებული შეწყვეტის სიაში და ერთარხიანი მოწყობილობის დაყოვნების სიაში (**G** და **H** ოპერანდები).

B ოპერანდით დასაკავებელი ერთარხიანი მოწყობილობის მიუწვდომელ მდგომარეობაში გადაყვანის მომენტში მოიცემა ტრანზაქტის დამუშავების რეჟიმები.

- **CO –** გაგრძელების რეჟიმი: ტრანზაქტის მიერ დასაკავებელი ერთარხიანი მოწყობილობის მიუწვდომელ დროში დამუშავების გაგრძელება;
- **RE –** წაშლის რეჟიმი: ტრანზაქტი, რომელსაც დაკავებული აქვს ერთარხიანი მოწყობილობა წაიშალოს და გადავიდეს **C** ოპერანდით მითითებულ ბლოკში.
- ფარულად – დამუშავების შეწყვეტა და ერთარხიანი მოწყობილობის შეწყვეტის სიაში მოთავსება; მისაწვდომლობის აღდგენის შემდეგ აღნიშ-ნულმა ტრანზაქტმა შესაძლებელია დაიკავოს ერთარხიანი მოწყობილობა და „მომსახურდეს“.

C ოპერანდი – არის ბლოკის ჭდე, სადაც წაშლის რეჟიმში უნდა გადავიდეს ტრანზაქტი, რომელმაც დაიკავა ერთარხიანი მოწყობილობა მისი მიუწვდომელ მდგომარეობაში გადაყვანის მომენტში.

D ოპერანდი – ტრანზაქტის პარამეტრის სახელი ან ნომერი, რომელმაც დაიკავა ერთარხიანი მოწყობილობა მისი მიუწვდომელ მდგომარეობაში გადაყვანის მომენტში; თუ ის წაიშლება (**RE** რეჟიმი), ანუ ამოირიცხება სიიდან, მაშინ ამ პარამეტრში ჩაიწერება ის დრო, რომელიც მომსახურების დასრულებამდე დარჩა წაშლილ ტრანზაქტს.

E ოპერანდით მოიცემა ტრანზაქტის დამუშავების რეჟიმები, რომლებიც იმყოფებიან წყვეტის სიაში მოწყობილობის მიუწვდომელ მდგომარეობაში გადაყვანის მომენტში, ანუ ისეთი ტრანზაქტები რომელთა მომსახურება შეწყდა ადრე მოცემულ ერთარხიან მოწყობილობაში:

- **CO** – გაგრძელების რეჟიმი: ერთარხიანი მოწყობილობის მუშაობის გაგრძელება მიუწვდომლობის დროს – ტრანზაქტები მომსახურდეს შეწყვეტის სიიდან;
- **RE** – წაშლის რეჟიმი: შეწყვეტის სიიდან ტრანზაქტები წაიშალოს და გადავიდეს ახალ ბლოკში, რომლის ჭდე ნაჩვენებია **F** ოპერანდით;
- ფარულად – დარჩნენ ადრე შეწყვეტილი ტრანზაქტები ერთარხიანი მოწყობილობის წყვეტის სიაში და აეკრძალოთ მისი დაკავება მიუწვდომლობის დროს.

F ოპერანდით მითითებულია ბლოკის ჭდე, სადაც უნდა გავიდეს ტრანზაქტი ერთარხიანი მოწყობილობის წყვეტის სიიდან (ისინი არ მოთავსდება სიაში), ამიტომ მათთვის შეუძლებელია მათ პარამეტრში იმ დროის შეტანა, რომელიც დარჩათ მათ მომსახურების დასრულებამდე.

F ოპერანდის გამოყენება შესაძლებელია მაშინ, როდესაც **E** ოპერანდი მითითებულია არაცხადად (ფარულად). ამ შემთხვევაში მომდევნო ბლოკში გადაადგილებული ტრანზაქტებისათვის შენარჩუნდება მომსახურების შეწყვეტა.

G ოპერანდით მოიცემა დამუშავების რეჟიმები, რომლებიც იმყოფებიან ერთარხიანი მოწყობილობის მისი მიუწვდომელ მდგომარეობაში გადაყვანის მომენტში გადადებულ შეწყვეტილ სიაში, ანუ დაყოვნების სიაში ელოდებიან შეწყვეტის შესრულებას:

- **CO** – გაგრძელების რეჟიმი: ერთარხიანი მოწყობილობის მუშაობის გაგრძელება მიუწვდომლობის დროს – ტრანზაქტები მომსახურდეს გადადებული შეწყვეტის სიიდან და დაყოვნების სიიდან;
- **RE** – წაშლის რეჟიმი: გადადებული შეწყვეტის სიიდან და დაყოვნების სიიდან ტრანზაქტები წაიშალოს და გადავიდეს ახალ ბლოკში, რომლის ჭდე ნაჩვენებია **4** ოპერანდით;
- ფარულად – ტრანზაქტები დარჩეს ერთარხიანი მოწყობილობის გადადებული შეწყვეტის სიაში და დაყოვნების სიაში აეკრძალოს მათ მისი დაკავება მიუწვდომლობის დროს.

H ოპერანდით ნაჩვენებია ახალი ბლოკის ჭდე, სადაც წაშლის რეჟიმში (**RE**) გადავა ტრანზაქტები გადადებული შეწყვეტის სიიდან და დაყოვნების სიიდან. როდესაც **G** ოპერანდი არ გამოიყენება, არ შეიძლება **A** ოპერანდის გამოყენება.

ერთარხიანი მოწყობილობის მიუწვდომლობა შენარჩუნდება იქამდე, ვიდრე ტრანზაქტი, რომელმაც გამოიწვია მიუწვდომელ მდგომარეობაში გადასვლა, არ შევა ბლოკში

FAVAIL A

FAVAIL ბლოკი ცვლის ერთარხიანი მოწყობილობის მდგომარეობას მისაწვდომლობით, ანუ აღადგენს ერთარხიან მოწყობილობაში ტრანზაქტების შესვლის ჩვეულ რეჟიმს.

A ოპერანდით მითითებული ყველა ტრანზაქტი, რომლებიც ელოდება ერთარხიანი მოწყობილობის მისაწვდომლობის მდგომარეობას, აქტიურდება და ცდილობს მომავალში დაიკავოს ის.

შენიშვნა 1: **B ... H** ოპერანდები მიეკუთვნება ზემოთ ნაჩვენებ სამ კლასს. სხვა ტრანზაქტები, რომელთაც სურთ ერთარხიანი მოწყობილობის შეწყვეტა, უკვე იმყოფებიან მიუწვდომელ მდგომარეობაში. აღნიშნულ კლასებში არ შედის **B ... H** ოპერანდები და მათთან არ გააჩნიათ არანაირი დამოკიდებულება.

შენიშვნა 2: ზემოთ წარმოდგენილი სამი კლასიდან ერთარხიანი მოწყობილობის გადასვლა მიუწვდომელ მდგომარეობაში და ტრანზაქტების დამუშავების გაგრძელების

ნების დართვა იძლევა არა მხოლოდ უარის, არამედ მომსახურების სხვადასხვა დისციპლინების იმიტაციის შესაძლებლობას.

7.2.2. ერთარხიანი მოწყობილობის მიუწვდომლობის და მისაწვდომლობის მდგომარეობის შემოწმება

ერთარხიანი მოწყობილობის მდგომარეობის შემოწმება მიუწვდომელ რეჟიმში შესაძლებელია **GATE** ბლოკით. (მისი ჩაწერის ფორმატი იხილეთ მეორე თავში).

პირობის შემოწმება მიმდინარეობს **X** პირობითი ოპერატორიდან ერთ-ერთით:

- **FNV – A** ოპერანდით მოცემული ერთარხიანი მოწყობილობა მიუწვდომელია;
- **FV – A** ოპერანდით მოცემული ერთარხიანი მოწყობილობა მისაწვდომია.

მაგალითად:

```
GATE      FNN      Stan
GATE   FV (FN$Rasp-X$SV)
GATE      FNN      Primvt, Wde1
```

პირველ მაგალითში **GATE** ბლოკი ტრანზაქტს შეუშვებს მაშინ, როდესაც **Stan** სახელით მითითებული მოწყობილობა მიუწვდომელი იქნება. მეორე მაგალითში ტრანზაქტი გადავა მომდევნო ბლოკში, როდესაც ერთარხიანი მოწყობილობა მისაწვდომია, რომლის ნომერი განისაზღვრება როგორც **FN$Rasp - X$SV** გამოსახულების გამოთვლის და მისი მთელ რიცხვამდე დამრგვალების შედეგი. მესამე მაგალითში ერთარხიანი მოწყობილობის მისაწვდომლობის დროს – **Primvt,** ანუ **GATE** ბლოკში პირობის არშესრულებისას ტრანზაქტი გადავა **Wde1** ჭდით მითითებულ ბლოკში.

7.3. სამანქანო დროის შემცირება და მომსახურების დისციპლინების ცვლილება მომხმარებლის სიების გამოყენების მეთოდით

მოდელში ტრანზაქტების გადაადგილება შესაძლებელია დაბლოკილ იქნას, მაგალითად, როგორც ზემოთ იქნა აღნიშნული, ერთარხიანი მოწყობილობების შემოწმებისას **GATE** და **TEST** ბლოკებით. ასეთი დაბლოკილი ტრანზაქტების დასათვალიერებლად და მათ გადასაადგილებლად ძალიან დიდი დრო იხარჯება. მანქანური დროის შესამცირებლად ასეთ დაბლოკილ ტრანზაქტებს მიზანმიმართულად ათავსებენ მომხმარებლის სიებში და ტოვებენ იქამდე, ვიდრე არ შესრულდება ისეთი პირობა, რომელიც მათ ნებას დართავს განახორციელონ მოძრაობა. ამ

შემთხვევაში არ მოქმედებს პრინციპი „პირველი მოვიდა – პირველი მომსახურდა“.

მომხმარებლის სიები თავის მხრივ წარმოადგენს რაღაც ბუფერს, რომელშიც დროებით თავსდებიან ტრანზაქტები. მომხმარებლის სიაში ტრანზაქტები შედის და გამოდის არა ავტომატურად, არამედ მომხმარებლის ნების დართვის თანახმად მოდელის ლოგიკის შესაბამისად სპეციალური ბლოკების მეშვეობით.

ტრანზაქტების მომხმარებლის სიაში შეტანას ემსახურება **LINK** ბლოკი (სიაში შეტანა), რომელიც შეიძლება გამოყენებულ იქნას ორ რეჟიმში: პირობით და უპირობო რეჟიმებში.

7.3.1. მომხმარებლის სიაში ტრანზაქტების შესვლა უპირობო რეჟიმში. LINK ბლოკი

უპირობო რეჟიმში **LINK** ბლოკს გააჩნია შემდეგი ფორმატი:

[სახელი] LINK A, B

A ოპერანდი მიუთითებს მომხმარებლის სიის სახელს ან ნომერს, რომელშიც უპირობოდ მოთავსდება **LINK** ბლოკში შესული ტრანზაქტი;

B ოპერანდი მიუთითებს თუ მომხმარებლის სიაში რომელ ადგილზე უნდა მოთავსდეს შესული ტრანზაქტი. შესაძლო მნიშვნელობები:

- **FIFO –** ტრანზაქტი მოთავსდება სიის ბლოკში;
- **LIFO –** ტრანზაქტი მოთავსდება სიის დასაწყისში;
- **PR –** ტრანზაქტები დალაგდება კლებადობის პრიორიტეტის მიხედვით;
- **P –** ტრანზაქტები მოთავსდება იმ ტრანზაქტების უკან, რომელთა შესაბამისი მნიშვნელობები მცირეა (პარამეტრის მნიშვნელობათა ზრდის თანრიგში);
- **M1 –** ტრანზაქტები მოთავსდება მოდელში ფარდობითი დროის ზრდის თანრიგში.

B ოპერანდის რანგში შესაძლებელია ზემოთ ჩამოთვლილი სტრა-ს (არითმეტიკული ცვლადი, ფუნქცია, ფრჩხილებში ჩასმული გამოსახულება) გარდა სხვა სტრა-ს გამოყენება. ამ შემთხვევაში გამოითვლება აქტიური ტრანზაქტის **B** ოპერანდი და ყველა დანარჩენი ტრანზაქტები რომლებიც მომხმარებლის სიაში იმყოფება რიგის დასაწყისიდან დაწყებული. ამის შემდეგ მიმდინარეობს ტრანზაქტების დაწყობა კლებადობის მიხედვით (მომხმარებლის სიაში გამოთვლილი მნიშვნელობის მიხედვით). მაგალითად, ბლოკი

LINK 3, FIFO

ბლოკში მათი შესვლისას ტრანზაქტები მოთავსდება მომხმარებლის სიის ბოლოში 3 ნომრით. ბლოკში

LINK Dar, P$Pol

შესული ტრანზაქტი თავსდება მომხმარებლის სიაში **Dar** სახელით და პარამეტრის მნიშვნელობები მათი ზრდადობის მიხედვით დალაგდება **Pol** სახელით.

პირობა, რომლის მიხედვითაც ტრანზაქტები თავსდება მომხმარებლის სიაში უპირობო რეჟიმში, მოწმდება მოდელის შემდგენელის მიერ მითითებული მეთოდით. მაგალითად, მომხმარებლის სიაში ტრანზაქტების შესვლისას ერთარხიანი მოწყობილობის დაკავების შემთხვევაში შესაძლებელია შემდეგი ჩაწერა:

. . .

GATE NU Rem1,Wde

SEIZE Rem1

. . .

Wde LINK Dar, FIFO

თუ **Rem1** ერთარხიანი მოწყობილობა დაკავებულია, მაშინ **GATE** ბლოკიდან ტრანზაქტი გადადის **Wde** ჭდით მითითებულ **LINK** ბლოკში და ტრანზაქტი შედის მომხმარებლის სიის ბოლოში **Dar** სახელით.

მომხმარებლის სიის მოდელის იგივე ფრაგმენტი გამოიყურება შემდეგნაირად:

. . .

```
          GATE  U  Rem1,     Wde
          LINK     Dar, FIFO
                   . . .
Wde       SEIZE    Rem1
                   . . .
```

აქ მოწმდება დაკავებულია თუ არა **Rem1** ერთარხიანი მოწყობილობა. თუ ერთარხიანი მოწყობილობა დაკავებულია, მაშინ ტრანზაქტი გადადის **LINK** ბლოკში და **Dar** სახელით მოთავსდება ერთარხიანი მოწყობილობის სიაში. თუ ერთარხიანი მოწყობილობა არ არის დაკავებული, მაშინ ტრანზაქტი გადადის **Wde-**ით მითითებულ **SEIZE** ბლოკში და იკავებს თავისუფალ ერთარხიან მოწყობილობას.

აღსანიშნავია, რომ მომხმარებლის სია არ არის შეზღუდული, ანუ მასში შესაძლებელია გარკვეული რაოდენობის ტრანზაქტების მოთავსება.

რეალური სისტემების მოდელირებისას მომხმარებლის სია შეიძლება გამოყენებულ იქნას იმიტაციისათვის. ქვემოთ მოყვანილ მაგალითში რეალიზებულია ტევადობის შეზღუდვა.

```
Tev    EQU      10
       . . .

       GATE     NU  Rem1, Wde1
       SEIZE        Rem1
```

```
      . . .
Wde1  TEST   L   CH$Dar, Tev, Wde2
      LINK       Dar, FIFO
```

თუ ერთარხიანი მოწყობილობა **Rem1** დაკავებულია, მაშინ **GATE** ბლოკი არ უშვებს ტრანზაქტებს **SEIZE** ბლოკში. იგი მიემართება **Wde1** ჭდით მითითებულ **TEST** ბლოკში, რომელიც დგას **LINK** ბლოკის წინ. თუ **Dar** სახელით მითითებული მომხმარებლის სიის მიმდინარე მნიშნელობა ნაკლებია **TEST** სახელით მოცემულ ტევადობაზე, მაშინ ტრანზაქტი მოხვდება მომხმარებლის სიაში, ხოლო წინააღმდეგ შემთხვევაში გადადის **Wde2** ჭდით მითითებულ ბლოკში. იგივე ფრაგმენტი შესაძლებელია შემდეგნაირად გამოიყურებოდეს:

```
 Tev  EQU    10
. . .
      GATE     U   Rem1, Wde1
      TEST L   CH$Dar, Tev, Wde2
      LINK         Dar, FIFO
Wde1  SEIZE        Rem1
```

თუ თუ **Rem1** ერთარხიანი მოწყობილობა დაკავებულია, მაშინ ტრანზაქტი შედის **TEST** ბლოკში. თუ **Dar** სახელით მითითებული მომხმარებლის სიის მიმდინარე მნიშვნელობა ნაკლებია **Tev** სახელით მოცემულ ტევადობაზე, მაშინ ტრანზაქტი შედის მომხმარებლის სიაში, წინააღმდეგ შემთხვევაში კი – მიემრთება

Wde2 ჭდით მითითებული ბლოკისაკენ. თუ ერთარხიანი მოწყობილობა არ არის დაკავებული, მაშინ ტრანზაქტი გადადის **Wde1** –ით მითითებულ **SEIZE** ბლოკში და იკავებს თავისუფალ **Rem1** ერთარხიან მოწყობილობას.

7.3.2. ტრანზაქტების გამოსვლა მომხმარებლის სიიდან. UNLINK ბლოკი

მომხმარებლის სიიდან ერთი ან რამდენიმე ტრანზაქტის გამოსვლას და მათ მოთავსებას მიმდინარე მოვლენის სიაში ემსახურება **UNLINK** ბლოკი (სიიდან გამოსვლა). მას გააჩნია შემდეგი ფორმატი:

[სახელი] UNLINK X A, B, C, [D], [E], [F]

სადაც **A** ოპერანდი მიუთითებს მომხმარებლის სიის სახელს ან ნომერს.

B ოპერანდი – ბლოკის ჭდეს, რომელშიც მოთავსდება მომხმარებლის სიიდან გამოდევნილი ტრანზაქტები.

C ოპერანდი მიუთითებს გამოდევნილი ტრანზაქტების რიცხვს. **ALL** საკვანძო (მომსახურე) სიტყვა გამოიყენება სიაში მოთავსებული ყველა ტრანზაქტების გამოსვლისათვის. ფარულად იგულისხმება, რომ როდესაც არ გამოიყენება **C** ოპერანდი, მაშინ იხმარება **ALL.**

D და **E** ოპერანდები **X** პირობით ოპერატორთან ერთად განსაზღვრავს მომხმარებლის სიიდან ტრანზაქტების გამოსვლის წესს და პირობას. **X** ოპერატორს გააჩნია იგივე მნიშვნელობები, რაც **TEST** ბლოკის შემთხვევაში. იმ შემთხვევაში, როდესაც **X** პირობითი ოპერატორი უნდა იქნას გამოყენებული და არ არის ნაჩვენები, მისი მნიშვნელობა ფარულად **E**–ს ტოლია. თუ **D** და **E** ოპერანდები არ გამოიყენება და **X** ოპერატორი არ არის მითითებული, ამ შემთხვევაში ტრანზაქტები გამოდის სიის დასაწყისიდან, ხოლო გამოსასვლელი ტრანზაქტების რიცხვი აუცილებლად უნდა მიეთითოს **C** ოპერანდით.

D ოპერანდი შესაძლებელია იყოს:

- ბულის ცვლადი;
- ტრანზაქტის პარამეტრის ნომერი;
- **BACK** მომსახურე სიტყვა.

თუ **D** ოპერანდი წარმოადგენს ბულის ცვლადს, მაშინ **E** ოპერანდი და **X** ოპერატორი არ გამოიყენება. ბულის ცვლადი გამოითვლება ტრანზაქტზე დამოკიდებულებით, რომელიც იმყოფება მომხმარებლის სიაში. თუ შედეგი არ არის ნულის ტოლი, ანუ გამოსვლის პირობა სრულდება, მაშინ ტრანზაქტი გამოვა. როგორც ცნობილია, გამოსული ტრანზაქტების რიცხვი განისაზღვრება **C** ოპერანდით.

თუ **D** ოპერანდით მითითებულია **BACK** მომსახურე სიტყვა, აგრეთვე **E** ოპერანდი და **X** ოპერატორი არ გამოიყენება, მაშინ

სიის ბოლოდან გამოსული ტრანზაქტების რიცხვი განისაზღვრება აუცილებლად **C** ოპერანდით.

თუ **D** ოპერანდი არ არის ბულის ცვლადი და არც **BACK** მომსახურე სიტყვა, მაშინ აუცილებლად უნდა მიეთითოს **E** ოპერანდი და და **X** პირობითი ოპერატორი. **D** ოპერანდი გამოითვლება მომხმარებლის სიაში მოთავსებულ ტრანზაქტზე დამოკიდებულებით და პარამეტრის ნომრის რანგში გამოიყენება მნიშვნელობა, რომელიც შედარდება **E** ოპერანდის გამოთვლილ შედეგს.

თუ **D** ოპერანდი მითითებულია, როგორც პარამეტრი, ხოლო **E** ოპერანდი არ გამოიყენება, მაშინ მომხმარებლის სიიდან ტრანზაქტის პარამეტრის მნიშვნელობა შედარდება გამოსასვლელი ტრანზაქტის ასეთივე პარამეტრის მნიშვნელობას. თუ ისინი ტოლია, მაშინ ტრანზაქტი გამოვა მომხმარებლის სიიდან. გამოსასვლელი ტრანზაქტების რიცხვი განისაზღვრება **C** ოპერანდით.

F ოპერანდით მითითებულია ბლოკის სახელი, სადაც გადავა **UNLINK** ბლოკიდან გამოსული ტრანზაქტი, თუ მომხმარებლის სიიდან არ არის გამოსული არც ერთი ტრანზაქტი.

თუ **F** ოპერანდი არ გამოიყენება, მაშინ გამოსული ტრანზაქტი გადადის შემდეგ ბლოკში. მაგალითად, ბლოკი:

UNLINK 4, Apd, 1

მომხმარებლის სიის დასაწყისიდან გამოდის 4 ნომრით მითითებული ერთი ტრანზაქტი და გადადის **Apd** ჭდით მითითებულ ბლოკში. ბლოკი:

UNLINK Dar, Mars, 1, BACK

მომხმარებლის სიის ბოლოდან გამოდის **Dar** სახელით მითითებული ერთი ტრანზაქტი და გადადის **Mars** ჭდით მითითებულ ბლოკში. ბლოკი:

UNLINK E P$Wiw, App1, ALL, App2, P$App2, App3

მომხმარებლის სიიდან გამოდის **Wiw** სახელით მითითებული ყველა ტრანზაქტი და გადადის **App1** ჭდით მითითებულ ბლოკში, **App2** სახელით მითითებულ პარამეტრის შემცველობა გამომავალი ტრანზაქტის იმავე სახელის მქონე პარამეტრის შემცველობის ტოლია. თუ ასეთი პარამეტრი სიაში არ აღმოჩნდება, გამომავალი ტრანზაქტი გადავა **App3** ჭდის მქონე ბლოკში, წინააღმდეგ შემთხვევაში იგი გადავა მომდევნო ბლოკში.

7.3.3. ტრანზაქტების შესვლა მომხმარებლის სიაში პირობით რეჟიმში. LINK ბლოკი

მომხმარებლის სიაში პირობით რეჟიმში ტრანზაქტების შესვლისათვის გამოიყენება **LINK** ბლოკი. მისი ჩაწერის ფორმატს გააჩნია შემდეგი სახე:

[სახელი] LINK A, B, [C]

C ოპერანდი მიუთითებს ჭდეს, სადაც უნდა გადავიდეს აქტიური ტრანზაქტი იმ შემთხვევაში, თუ მომხმარებლის სიის ინდიკატორი დგას 0-ში (ანუ გამორთულია). მომხმარებლის ყოველ სიას გააჩნია თავისი ინდიკატორი. იმ შემთხვევაში, როდესაც **C** ოპერანდი არ გამოიყენება, ანუ **LINK** ბლოკი მუშაობს უპირობო რეჟიმში, მაშინ ინდიკატორი დგას 1-ში (ანუ ჩართულია) და **LINK** ბლოკში შესული ყველა ტრანზაქტი მოთავსდება მომხმარებლის სიაში.

LINK ბლოკის გამოყენებით მოდელის ფრაგმენტი შემდეგნაირად გამოიყურება:

```
. . .
        LINK      Nak, M1
Wde1    SEIZE     Can
        ADVANCE   V$UK
        RELEASE   Can
```

```
                UNLINK      Nak, Wde1, 1
                . . .
```

როდესაც **LINK** ბლოკი მოთავსებულია **Can** სახელით მითითებული ერთარხიანი მოწყობილობის წინ, მაშინ მოდელი არ მუშაობს. **LINK** ბლოკში შესული ყველა ტრანზაქტი მოთავსდება **Nak** სახელით მითითებულ მომხმარებლის სიაში და არც ერთი ტრანზაქტი არ დაიკავებს **Can** სახელით მითითებულ ერთარხიან მოწყობილობას. შესაბამისად **UNLINK** ბლოკი არ იმუშავებს.

ანალოგიურ ფრაგმენტში **LINK** ბლოკში **C** ოპერანდის გამოყენებისას მოდელი შემდეგნაირად გამოიყურება:

```
                . . .
                LINK        Nak,  M1, Wde1
          Wde1  SEIZE       Can
                ADVANCE     V$UK
                RELEASE     Can
                UNLINK      Nak, Wde1, 1
                . . .
```

მომხმარებლის სიის ინდიკატორი იმართება **LINK** და **UNLINK** ბლოკებით. იმ შემთხვევაში, როდესაც **LINK** ბლოკში **C** ოპერანდის გამოყენებისას **UNLINK** ბლოკი აღმოაჩენს, რომ მომხმარებლის სია ცარიელია, ინდიკატორი დადგება 0-ში. **LINK** ბლოკში შესული პირველი ტრანზაქტი ნახავს რა გამორთულ მდგომარეობაში მყოფ ინდიკატორს, ის არ მოთავსდება მომხმარებლის სიაში, არამედ გადავა **Wde1** ჭდით მითითებულ ბლოკში, რის შემდეგ **LINK** ბლოკის ინდიკატორი დადგება 1-ში.

ამის შემდეგ ტრანზაქტს შეუძლია შევიდეს **LINK** ბლოკში გაცილებით უფრო ადრე, ვიდრე წინა ტრანზაქტი დაასრულებს მომსახურებას, ან მომსახურების დასრულების შემდეგ.

თუ მომდევნო ტრანზაქტი **LINK** ბლოკში შევიდა იმაზე ადრე, ვიდრე **Can** სახელით მითითებული ერთარხიანი მოწყობილობის მომსახურება დასრულდება წინა ტრანზაქტისათვის, მაშინ იგი მოთავსდება მომხმარებლის სიაში.

დავუშვათ, რომ **Can** სახელით მითითებული ერთარხიანი მოწყობილობის მომსახურება დასრულდა იმაზე გაცილებით ადრე, ვიდრე შემდეგი ტრანზაქტი შევა **LINK** ბლოკში. მომსახურებული ტრანზაქტი, რომელიც ასევე გამოსულია, **UNLINK** ბლოკში შესვლისას აღმოაჩენს, რომ მომხმარებლის სია თავისუფალია, ამ შემთხვევაში **UNLINK** ბლოკი მომხმარებლის სიის ინდიკატორს დააყენებს 0-ში. ამიტომ, შემდეგი ტრანზაქტი კვლავ მიემართება **Wde1** ჭდით მითითებულ ბლოკში და არ მოთავსდება მომხმარებლის სიაში. ასე რომ **LINK** ბლოკი პირობით რეჟიმში და **UNLINK** ბლოკი მართავენ მომხმარებლის სიაში ტრანზაქტების მოთავსებას: თუ სია თავისუფალია ტრანზაქტი მასში არ მოთავსდება. იგი მიემართება **C** ოპერანდით მითითებულ ბლოკში.

LINK და **UNLINK** ბლოკების საშუალებით მომხმარებელს შეუძლია დინამიკაში დაყოვნების სიებისგან დამოუკიდებლად მოახდინოს პირადი სიების ფორმირება, რომლებიც ავტომატურად იმართება **GPSS World**-ის საშუალებით.

შენიშვნა: **LINK** და **UNLINK** ბლოკების გამოყენებისას **QUEUE-DEPART** ბლოკები რიგის შესახებ სტატისტიკური ინფორმაციის შესაგროვებლად არ გამოიყენება. ასე, რომ სტატისტიკური ინფორმაციის შესახებ მონაცემთა მიღება შესაძლებელია მომხმარებლის სიების მეშვეობით.

VIII თავი
სისტემების მოდელების აგება მრავალარხიანი მოწყობილობებისა და გადამრთველებისათვის

8.1. მრავალარხიანი მოწყობილობის მიუწვდომელ მდგომარეობაში გადასვლა და მისაწვდომლობის აღდგენა

მრავალარხიანი მოწყობილობის მიუწვდომლობა მოდელირდება **SUNAVAIL** ბლოკის (**S** სიმბოლო აღნიშნავს მრავალარხიან მოწყობილობას, **UNAVAIL** – მიუწვდომლობას) მეშვეობით. მისი ჩაწერის ფორმატს გააჩნია შემდეგი სახე:

SUNAVAIL A

A ოპერანდი არის მრავალარხიანი მოწყობილობის სახელი ან ნომერი. ის შეიძლება იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა, ან სტრა*პარამეტრი.

მაგალითად:

SUNAVAIL Miu

იმ შემთხვევაში, როდესაც ტრანზაქტი შედის აღნიშნულ ბლოკში, **Miu** სახელით მითითებული მრავალარხიანი მოწყობილობა ხდება მიუწვდომელი. თუ მიუწვდომელ მდგომარეობაში გადასვლისას მრავალარხიან მოწყობილობაში იმყოფება ტრანზაქტები ანუ მრავალარხიანი მოწყობილობის შემცველობა არ უდრის ნულს, მაშინ ასეთი ტრანზაქტების მომსახურება გაგრძელდება, ვიდრე მიმდინარე შემცველობა არ გახდება ნულის ტოლი. ტრანზაქტები, რომლებიც ცდილობენ დაიკავონ მრავალარხიანი მოწყობილობა მისი მიუწვდომელ მდგომარეობაში ყოფნის დროს, არ შედიან **ENTER** ბლოკში და თავსდებიან მრავალარხიანი მოწყობილობის დაყოვნების სიაში.

მიუწვდომელ მდგომარეობაში ყოფნა გრძელდება იქამდე, ვიდრე ტრანზაქტი არ შევა **SAVAIL** ბლოკში. მის ფორმატს გააჩნია შემდეგი სახე:

SAVAIL A

A ოპერანდი მიუთითებს მრავალარხიანი მოწყობილობის ნომერს ან სახელს. იგი იმავე ტიპისაა, როგორც **SUNAVAIL** ბლოკის შემთხვევაში.

მაგალითი 8.1: თუ მრავალარხიანი მოწყობილობის მისაწვდომლობის მდგომარეობაში გადასვლის მომენტში მისი დაყოვნების სიაში აღმოჩნდება ტრანზაქტები, მათ საშუალება ეძლევათ დაიკავონ მრავალარხიანი მოწყობილობა შესაბამისი

დისციპლინის მიხედვით **„first-fit-with-skip“** („პირველი მოვიდა – პირველი მომსახურდა“). ტრანზაქტები, რომლებიც უარს აცხადებს დაიკავოს მრავალარხიანი მოწყობილობა, რჩება დაყოვნების სიაში.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                    * * * * * * * * * * * * * * * * * * * * *
*                    *   CPSS WORLD SIMULATION   *
*                    * * * * * * * * * * * * * * * * * * * * *

*          მრავალარხიანი მოწყობილობის განსაზღვრა

  Arxi     STORAGE              3

*          ტრანზაქტების შესვლის და იმიტაციის სეგმენტი

           GENERATE    , , , 3      ; ტრანზაქტების შესვლა
           ENTER       Arxi         ; მრავალარხიანი მოწყობილობის
                                      დაკავება
           ADVANCE     10           ; მომსახურების დრო
           LEAVE       Arxi         ; მრავალარხიანი მოწყობილობის
                                      განთავისუფლება
           TERMINATE                ; მომსახურებული ტრანზაქტი

*          მიუწვდომლობის იმიტაციის სეგმენტი

           GENERATE    5, , , 1     ; ტრანზაქტ-ინიციატორი იმყოფება
                                      მიუწვდომელ მდგომარეობაში
           SUNAVAIL    Arxi         ; მიუწვდომელ მდგომარეობაში
                                      გადასვლა
           ADVANCE     6            ; მისაწვდომლობის აღდგენა
           SAVAIL      Arxi         ; მისაწვდომლობის მდგომარეობაში
                                      გადასვლა
           TERMINATE   1            ; დასრულება
```

პირველი სეგმენტის **GENERATE** ბლოკი გენერირებს სამ ტრანზაქტს **t = 0** მომენტში, რომლებიც იკავებს **Arxi** მრავალარხიან მოწყობილობას, რომელიც განსაზღვრულია **STORAGE** ბრძანებით.

მეორე სეგმენტის **GENERATE** ბლოკი **t = 5** დროის მომენტში გენერირებს ტრანზაქტს, რომელიც შედის **SUNAVAIL** ბლოკში და **Arxi** სახელით მითითებული მრავალარხიანი მოწყობილობა გადადის მიუწვდომლობის მდგომარეობაში.

8.2. მრავალარხიანი მოწყობილობის მდგომარეობის შემოწმება

8.2.1. მრავალარხიანი მოწყობილობის მდგომარეობის შემოწმება GATE ბლოკით

მრავალარხიანი მოწყობილობის, ისევე როგორც ერთარხიანი მოწყობილობის მდგომარეობა მოწმდება **GATE** ბლოკის მიერ, რომელსაც გააჩნია იგივე ფორმატი:

GATE X A, [B]

განსხვავება მდგომარეობს **X** პირობითი ოპერატორის მნიშვნელობებში, რომლებიც შეიძლება იყოს შემდეგი სახის:

SE – ოპერანდით მითითებული მრავალარხიანი მოწყობილობა ცარიელია;

SF – ოპერანდით მითითებული მრავალარხიანი მოწყობილობა შევსებულია;

SNE – ოპერანდით მითითებული მრავალარხიანი მოწყობილობა არ არის ცარიელი;

SNF – ოპერანდით მითითებული მრავალარხიანი მოწყობილობა არ არის შევსებული;

SNV – ოპერანდით მითითებული მრავალარხიანი მოწყობილობა მიუწვდომელია;

SV – ოპერანდით მითითებული მრავალარხიანი მოწყობილობა მისაწვდომია.

მაგალითად:

GATE SNF Can, Wde5

თუ **Can** სახელწოდების მქონე მრავალარხიანი მოწყობილობა არ არის შევსებული, ანუ გაგვაჩნია თავისუფალი არხები (მეხსიერების ელემენტები), **GATE** ბლოკში დაწესებული პირობა სრულდება და ტრანზაქტი გადაადგილდება მომდევნო ბლოკში. თუკი მრავალარხიანი მოწყობილობა შეივსება, ტრანზაქტი მიმართული იქნება ბლოკისაკენ **Wde5** ჭდით.

მაგალითი 8.2: მრავალარხიანი მოწყობილობა გადადის მიუწვდომელ მდგომარეობაში. აღნიშნულ მდგომარეობაში მისი გადასვლის მომენტისათვის მრავალარხიან მოწყობილობაში მყოფი ტრანზაქტების მომსახურება გაგრძელდება, მაგრამ ხელახლა შემოსული ტრანზაქტები არ თავსდება მრავალარხიანი მოწყობი-

ლობის დაყოვნების სიაში. მრავალარხიანი მოწყობილობის მისაწვდომლობის აღდგენა წარმოებს მას შემდეგ, როგორც კი იგი ცარიელი აღმოჩნდება.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*       მრავალარხიანი მოწყობილობის განსაზღვრა

        Sist    STORAGE     7

*               ტრანზაქტების შემოსვლის და მომსახურების
                იმიტაციის სეგმენტი

                GENERATE        2, , ,5     ; ტრანზაქტების წყარო
                GATE   SV  Sist, Wde1       ;  მისაწვდომია მამ?
                GATE   SNF   Sist, Wde1     ; დიახ, თუ მამ არ არის
                                            შევსებული, მაშინ გადადი Wde1-ზე
                ENTER  Sist,3               ; დავიკავოთ თუკი არის მამ-ის
                                              თავისუფალი არხები
                ADVANCE   5                 ; მომსახურების დრო
                LEAVE       Sist, 3         ; გავანთავისუფლოთ მამ-ის არხები
                TERMINATE       1           ; მომსახურებული ტრანზაქტები
Wde1    TERMINATE 1                         ; დაკარგული ტრანზაქტები
```

* მრავალარხიანი მოწყობილობის მიუწვდომლობის იმიტაციის სეგმენტი

```
          GENERATE  , , ,1          ; მიუწვდომლობის
                                       ტრანზაქტ-ინიციატორი
Wde2      ADVANCE   7               ; მისაწვდომ მდგომარეობაში
                                       ყოფნის დრო
          SUNAVAIL  Sist            ; მიუწვდომლობის
                                       მდგომარეობაში გადასვლა
          GATE  SE  Sist            ; ცარიელია მამ?
          ADVANCE   1               ; დიახ, მაშინ მისაწვდომობის
                                       აღდგენა
          SAVAIL    Sist            ; მისაწვდომობის
                                       მდგომარეობაში გადასვლა
          TRANSFER  ,Wdet2          ; გადასვლა მომდევნო ციკლზე
```

t = 0 დროს სეგმენტი 2-ის **GENERATE** ბლოკში გენერირდება ერთი (**XN1 = 2**) ტრანზაქტი, რომელიც შედის **ADVANCE** დაყოვნების ბლოკში.

t = 2 დროს **GENERATE** ბლოკის I სეგმენტ პირველი (**XN1 = 1**) ტრანზაქტი გაივლის **GATE** ბლოკის პირველ და მეორე ბლოკებს და იკავებს მამ-ის **Sist**7 არხიდან სამ არხს. აღნიშნული ტრანზაქტის მომსახურება დასრულდება **t = 2 + 5 = 7** დროს.

t = 4 დროს მეორე **(XN1** = 3) ტრანზაქტი ასევე გაივლის **GATE** ბლოკის I სეგმენტის ორივე ბლოკს და იკავებს მამ-ის მომდევნო **Sist** სამ არხს. თავისუფალი რჩება ერთი არხი. მეორე ტრანზაქტის მომსახურება დასრულდება **t = 2 + 5 = 7** დროს.

t = 6 დროს I სეგმენტის მესამე **(XN1 = 4)** ტრანზაქტი გაივლის **GATE** ბლოკის პირველ ბლოკს და შევა აღნიშნული ბლოკის მეორე ბლოკში. ვინაიდან მამ **Sist**-ს გააჩნია ერთი თავისუფალი არხი, ანუ იგი არ არის შევსებული, პირობა სრულდება **GATE** ბლოკის მეორე ბლოკში და ტრანზაქტი მიმართული იქნება **ENTER** ბლოკში, მაგრამ ერთი თავისუფალი არხის არსებობა არ არის საკმარისი მოთხოვნის დასაკმაყოფილრბლად, საჭიროა სამი არხი, ამიტომ მესამე ტრანზაქტი თავსდება მამ-ის **Sist** დაყოვნების სიაში.

t = 7 დროს **SUNAVAIL** ბლოკის მეშვეობით მამ **Sist** გადადის მიუწვდომელ მდგომარეობაში. ამავე დროს დასრულდება პირველი ტრანზაქტის მომსახურება, მაგრამ მესამე ტრანზაქტი არ დაიკავებს მამ-ს და ისევე დარჩება დაყოვნების სიაში.

ვინაიდან მამ **Sist**-ში იმყოფება ერთი ტრანზაქტი, ანუ იგი არ არის ცარიელი, ამ შემთხვევაში, მიუწვდომელ მდგომარეობაში გადასვლის გამომწვევი ტრანზაქტი შეფერხებული იქნება II სეგმენტის **GATE** ბლოკის მიერ იმიტომ, რომ მასში დაწესებული პირობა არ სრულდება.

t = 8 დროს I სეგმენტის მეოთხე (**XN1 = 5**) ტრანზაქტი მამ **Sist**-ის მიუწვდომლობის შედეგად **GATE** ბლოკის პირველი ბლოკის

მიერ **Wde1** ჭდით მიემართება **TERMINATE** ბლოკისკენ და ნადგურდება.

t = 9 დროს დასრულდება მეორე ტრანზაქტის მომსახურება. მამ-ი დაცარიელდება, II სეგმენტის პირობა **GATE** ბლოკში შესრულდება და მიუწვდომლობის დამაგენერირებელი ტრანზაქტი შევა დაყოვნების **ADVANCE** ბლოკში.

t = 10 დროს I სეგმენტის მეხუთე (**XN1 = 6**) ტრანზაქტი მამ **Sist**-ის მიუწვდომლობის შედეგად **GATE** ბლოკის პირველი ბლოკის მიერ **Wde1** ჭდით მიემართება **TERMINATE** ბლოკისკენ და ნადგურდება.

ასევე **9 + 1 = 10** დროს **SAVAIL** ბლოკი მამ-ს გადაიყვანს მიუწვდომელ მდგომარეობაში. მესამე ტრანზაქტი მამ-ის დაყოვნების სიიდან დაიკავებს მამ-ის სამ არხს. მესამე (**XN1 = 4**) ტრანზაქტის მომსახურება დასრულდება **t = 10 + 5 = 15** დროს.

როგორც მაგალითიდან ჩანს, **GATE** ბლოკი საშუალებას იძლევა განსაზღვრულ იქნას მხოლოდ მრავალარხიანი მოწყობილობის შეუვსებლობის მდგომარეობა, ანუ თავისუფალი არხების არსებობა, მაგრამ საკმარისია ისინი თუ არა მოთხოვნის დასაკმაყოფილებლად იგი არ განსაზღვრავს. ამიტომ თუკი **ENTER** ბლოკში **B** ოპერანდი არ უდრის ერთს და მამ-ის ტევადობა უნაშთოდ არ იყოფა **B** ოპერანდის მნიშვნელობაზე, ტრანზაქტი გაივლის **GATE** ბლოკს და თავსდება მამ-ის დაყოვნების სიაში. **B** არ უდრის ერთს და **B** ოპერანდის

მნიშვნელობაზე მამ-ის ტევადობის მთელად გაყოფის, ან **B = 1** დროს მამ-ის დააყოვნების სია ყოველთვის ცარიელია.

8.2.2. მრავალარხიანი მოწყობილობის მდგომარეობის შემოწმება ბულის ცვლადების გამოყენებით და TEST ბლოკით

მრავალარხიანი მოწყობილობის მდგომარეობის შემოწმებისათვის **TEST** ბლოკს გააჩნია ჩაწერის იგივე ფორმატი, პირობითი ოპერატორები და მუშაობის რეჟიმები, როგორც ერთარხიანი მოწყობილობის შემთხვევაში.

ბულის ცვლადში შესაძლებელია გამოვიყენოთ შემდეგი ლოგიკური ოპერატორები, რომლებიც დაკავშირებულია მრავალარხიანი მოწყობილობასთან:

- **SF** = 1, თუ მრავალარხიანი მოწყობილობა შევსებულია, წინააღმდეგ შემთხვევაში – 0;
- **SE** = 1, თუ მრავალარხიანი მოწყობილობა ცარიელია, წინააღმდეგ შემთხვევაში – 0;
- **SV** = 1, თუ მრავალარხიანი მოწყობილობა მიღწევადია, წინააღმდეგ შემთხვევაში – 0.

აგრეთვე მრავალარხიანი მოწყობილობის სტრა:

- **S** – მრავალარხიანი მოწყობილობის დაკავებული არხების რიცხვი;

- **SA** – მრავალარხიანი მოწყობილობის დაკავებული არხების საშუალო მნიშვნელობა;
- **SC** – მრავალარხიანი მოწყობილობის გამოყენების მრიცხველი;
- **SR** – მრავალარხიანი მოწყობილობის გამოყენების კოეფიციენტი;
- **SM** – მრავალარხიანი მოწყობილობის დაკავებული არხების მაქსიმალური რიცხვი;
- **ST** – მრავალარხიანი მოწყობილობის ერთი არხის გამოყენების საშუალო დრო.

8.3. გადამრთველების მოდელირება

ისეთი მოწყობილობების მოდელირებისათვის როგორიცაა გადამრთველები, **GPSS**-ში გამოიყენება ლოგიკური გასაღები. იგი იმყოფება ორი მდგომარეობიდან ერთ-ერთში:

- ჩართულია (**ON** ან 1);
- გამორთულია (**OFF** ან 0).

გასაღების მდგომარეობის მიხედვით იცვლება ტრანზაქტების მოძრაობის მიმართულება.

ლოგიკური გასაღები მოდელირდება **LOGIC** ბლოკით. მას გააჩნია შემდეგი ფორმატი:

LOGIC X A

სადაც **A** ოპერანდი არის ლოგიკური გასაღების სახელი ან ნომერი. იგი შესაძლებელია იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი.

X არის ლოგიკური ოპერატორი. ლოგიკური გასაღების მდგომარეობა დგინდება მისი შემდეგი მნიშვნელობების მიხედვით:

- **S – A** ოპერანდით მოცემული ლოგიკური გასაღები ჩართულია;
- **R – A** ოპერანდით მოცემული ლოგიკური გასაღები გამორთულია;
- **I –** ლოგიკური გასაღების ინვენსირება, ანუ მისი მნიშვნელობა იცვლება საწინააღმდეგო მნიშვნელობით. მაგალითად, თუ იგი ჩართულია, მაშინ გამოირთვება ან პირიქით.

8.3.1. ლოგიკური გასაღების მდგომარეობის შემოწმება GATE ბლოკით

ლოგიკური გასაღების მდგომარეობის შემოწმებისათვის **GATE** ბლოკს გააჩნია ჩაწერის იგივე ფორმატი, პირობითი ოპერატორები და მუშაობის რეჟიმები, როგორც ერთარხიანი და მრავალარხიანი მოწყობილობების შემთხვევაში.

პირობითმა ოპერატორმა შეიძლება მიიღოს შემდეგი მნიშვნელობები:

- **LS** = 1 თუ **A** ოპერანდით მოცემული ლოგიკური გასაღები ჩართულია; 0 – თუ გამორთულია;
- **LR** = 1 თუ **A** ოპერანდით მოცემული ლოგიკური გასაღები გამორთულია; 0 – თუ ჩართულია.

8.3.2. ლოგიკური გასაღების მდგომარეობის შემოწმება TEST ბლოკით

ლოგიკური გასაღების მდგომარეობის შემოწმებისათვის **TEST** ბლოკს გააჩნია ჩაწერის იგივე ფორმატი, პირობითი ოპერატორები და მუშაობის რეჟიმები, როგორც ერთარხიანი და მრავალარხიანი მოწყობილობების შემთხვევაში.

ბულის ცვლადში გამოიყენება მხოლოდ სტრა **LS**. ბულის ცვლადში **LR** ჩაწერისას წარმოიშვება შეცდომა.

IX თავი

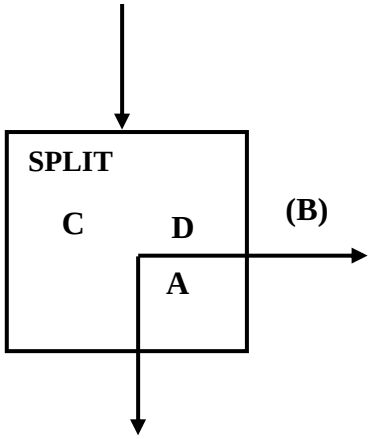
GPSS-ის გამოყენების თავისებურებანი რთული სისტემების მოდელირებისათვის

9.1 მოდელებში ტრანზაქტების ასლების შექმნის ბლოკის გამოყენება. SPLIT ბლოკი

როგორც ცნობილია, **GENERATE** ბლოკი წარმოადგენს ტრანზაქტების შექმნის და მათი მოდელში შეტანის ძირითად საშუალებას. **GENERATE** ბლოკის გარდა ტრანზაქტების შესაქმნელად გამოიყენება **SPLIT** ბლოკი, მაგრამ **GENERATE** ბლოკისაგან განსხვავებით **SPLIT** ბლოკი არ ქმნის დამოუკიდებელ ტრანზაქტებს, მხოლოდ გენერირებს მასში შესული ტრანზაქტების წარმომშობი ასლების მოცემულ რიცხვს. **SPLIT** ბლოკს გააჩნია შემდეგი ფორმატი:

SPLIT A, [B], [C], [D]

ხოლო გრაფიკულად შემდეგნაირად გამოისახება:



ნახ. 9.1. **SPLIT** ბლოკი **A, B, C** და **D** ოპერანდებით

A ოპერანდი მიუთითებს ერთი ოჯახის მიერ შექმნილი ასლების რიცხვს. ის შესაძლოა იყოს დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი. ყველა ასლი ფორმირდება **SPLIT** ბლოკში წარმოშობილი ტრანზაქტის შესვლის მომენტში.

B ოპერანდი – ბლოკის ნომერი, რომელშიც გადადის წარმომშობი ტრანზაქტების ასლები. **B** ოპერანდის მნიშვნელობა გამოითვლება თითოეული ასლისათვის ცალ-ცალკე.

SPLIT ბლოკის გავლის შემდეგ წარმომშობი ტრანზაქტი მიემართება მომდევნო ბლოკში, ხოლო ყველა ასლი კი – **B** ოპერანდით მითითებულ მისამართზე. ამგვარად, თუ **A** ოპერანდი **K**-ს ტოლია, მაშინ ბლოკიდან გამოვა **K** + 1 ტრანზაქტი. შემდეგ წარმომშობი ტრანზაქტი და მისი ასლი თანაბარუფლებიანი არიან და შესაძლებელია გაიარონ კვლავ **SPLIT** ბლოკების ნებისმიერი რიცხვი.

C ოპერანდით შესაძლებელია მითითებულ იქნას პარამეტრის ნომერი, რომელიც გამოიყენება ასლების თანამიმდევრობითი ნომრების (ნუმერაციის) მისანიჭებლად. თუ **C** ოპერანდი არ გამოიყენება, მაშინ **SPLIT** ბლოკიდან ფარულად გამოსული ტრანზაქტების პარამეტრებში მათი ნომრები არ არსებობს. **B** და **C** ოპერანდები იმავე ტიპისაა, რაც **A** ოპერანდი.

მაგალითები:

SPLIT 6

SPLIT 9,Wde12

SPLIT 4,Wde1, 5

პირველ მაგალითში **SPLIT** ბლოკიდან გამოდის **1 + 6 = 7** ტრანზაქტი. ექვსი ასლი და წარმომშობი ტრანზაქტი მიემართება მომდევნო ბლოკისაკენ, ვინაიდან **B** ოპერანდი არ გამოიყენება.

წარმომშობ პარამეტრში და ტრანზაქტების ასლში ნომრის ჩაწერა არ მოხდება, რადგან **C** ოპერანდი გამოტოვებულია. ასლებს გააჩნიათ იგივე პრიორიტეტი, პარამეტრების მნიშვნელობები და მოდელში შესვლის დრო, რაც მათ წარმომშობ ტრანზაქტს.

მეორე მაგალითში **SPLIT** ბლოკი გენერირებს **1 + 9 = 10** ტრანზაქტს. შემქმნელი ტრანზაქტი გადადის მომდევნო ბლოკში, ხოლო ცხრა ასლი – **Wde1** ჭდით მითითებულ ბლოკში. აგრეთვე არ მოხდება ტრანზაქტის პარამეტრში ნომრების ჩაწერა.

მესამე მაგალითში **SPLIT** ბლოკი შემქმნელი ტრანზაქტის შესვლისას გენერირებს ოთხ ასლს, რომლებიც მიემართება **Wde1** ჭდით მითითებული ბლოკისაკენ. თითოეულ ასლს გააჩნია თანაბარი პრიორიტეტი, მოდელში შესვლის დრო და პარამეტრების მნიშვნელობები (რაც მათ წარმომშობ ტრანზაქტს), გამონაკლისს წარმოადგენს 5 ნომრით მითითებული პარამეტრი.

5 ნომრით მითითებულ პარამეტრში ყოველი შემდეგი ასლისათვის ჩაწერილი იქნება რიგითი ნომერი. თუ წარმოშობი ტრანზაქტის 5 ნომრით მითითებული პარამეტრი წინასწარ არ არის განსაზღვრული, იგი იქმნება და მას ენიჭება ნულის ტოლი მნიშვნელობა.

განხილულ მაგალითში წარმომშობი ტრანზაქტის 5 ნომრით მითითებული პარამეტრის მნიშვნელობა 0-ის ტოლია. **SPLIT** ბლოკიდან გამოსვლის შემდეგ წარმომშობი ტრანზაქტის 5 ნომრით მითითებულ პარამეტრში ჩაიწერება 1, ხოლო 5 ნომრით მითითებულ პარამეტრის ასლებში შესაბამისად 2, 3, 4 და 5.

დავუშვათ, რომ წარმომშობი ტრანზაქტის 5 ნომრით მითითებული პარამეტრი წინასწარ არის განსაზღვრული და მისი მნიშვნელობა **n**-ის ტოლია. მაშინ, წარმომშობი ტრანზაქტის 5 ნომრით მითითებულ პარამეტრში და მის ასლებში **SPLIT** ბლოკიდან გამოსვლის შემდეგ ჩაიწერება შესაბამისად **n** + 1, **n** + 2, **n** + 3, **n** + 4 და **n** + 5.

წარმოგიდგენთ კიდევ ერთ მაგალითს, რომელშიც დემონსტრირებულია ტრანზაქტების ასლების მიმართულების

შესაძლებლობა მოდელში ბლოკების თანამიმდევრობითი განლაგებისას

SPLIT3, P$Mis, Mis

B ოპერანდით მითითებულია ბლოკი, რომელშიც უნდა შევიდეს ტრანზაქტების ასლები. **B** ოპერანდის მნიშვნელობა, ანუ ბლოკის ნომერი ხელახლა გამოითვლება თითოეული შექმნილი ასლისათვის. თუ **Mis** სახელით მითითებული პარამეტრი შეიცავს ბლოკის ნომერს, დავუშვათ **n**-ს, მაშინ პირველი ასლი მიემართება **n + 2** ბლოკისაკენ, მეორე – **n + 3** -კენ, მესამე – **n + 4** -კენ. აღნიშნული ნომრები კი ჩაიწერება **Mis** სახელით მითითებულ პარამეტრის ასლში. **Mis** სახელით მითითებული პარამეტრი **n + 1** მნიშვნელობით გადადის მომდევნო ბლოკში. **SPLIT** ბლოკის მიერ შექმნილი თითოეული ახალი ასლი ხდება ტრანზაქტების ოჯახის (ანსამბლის) წევრი, რომლებიც შექმნილია **GENERATE** ბლოკის საწყისი ტრანზაქტებით. ტრანზაქტები, რომლებიც მიეკუთვნება ერთ ოჯახს, ერთიანდებიან სიაში. თუ **SPLIT** ბლოკში არ წარმოებს წარმომშობი ტრანზაქტის და მისი ასლების ნუმერაცია, მაშინ ოჯახის შიგნით შეუძლებელია იმის გარჩევა, თუ რომელი ტრანზაქტია წარმომშობი. ოჯახში ტრანზაქტების რიცხვი წარმოსახვითია. **GENERATE** ბლოკის მიერ შექმნილი თითოეული ტრანზაქტი წარმოადგენს ცალკეულ ოჯახს

(ტრანზაქტის ნომერი ოჯახის ნომრის ტოლია, ანუ **GENERATE** ბლოკის მიერ რამდენჯერაც არის გენერირებული მოდელში ტრანზაქტები, იმდენი ოჯახია). ამგვარად, სისტემაში ოჯახის რიცხვი ატარებს წარმოსახვით ხასიათს და ოჯახი არსებობს იქამდე, ვიდრე მასში ერთი ტრანზაქტი მაინც იმყოფება.

9.2 მოდელებში ტრანზაქტების სინქრონიზებული მოძრაობის ორგანიზაცია

9.2.1. ASSEMBLE ბლოკი

ASSEMBLE (გაერთიანების) ბლოკი გამოიყენება ტრანზაქტების მოცემული რიცხვის გაერთიანებისათვის, რომლებიც მიეკუთვნება ერთ ოჯახს ერთ ტრანზაქტში (ანუ ახორციელებს მოცემული რიცხვის ტრანზაქტების ნაკრებს). შეკრების შემდეგ **ASSEMBLE** ბლოკიდან გამოდის მხოლოდ ერთი ტრანზაქტი, რომელიც გადადის მომდევნო ბლოკში. **ASSEMBLE** ბლოკს გააჩნია შემდეგი ფორმატი:

ASSEMBLE A

სადაც **A** ოპერანდი არის ერთი ოჯახის ტრანზაქტების რიცხვი, რომლებიც მონაწილეობს ნაკრებში (ნაკრების მრიცხველის საწყისი რიცხვი). ის შესაძლებელია იყოს დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა, ან

სტრა*პარამეტრი. ერთსა და იმავე **ASSEMBLE** ბლოკში შესაძლებელია რამდენიმე ოჯახის ტრანზაქტების ნაკრებთა ერთდროული არსებობა. ამგვარად, ტრანზაქტების შეკრება გრძელდება რაღაც სამოდელო დროის შუალედის განმავლობაში, ამიტომ **ASSEMBLE** ბლოკს გააჩნია სინქრონიზაციის სია. სინქრონიზაციის სიაში მოთავსდება **ASSEMBLE** ბლოკში თითოეული ოჯახის ნაკრებების პირველად შესული ტრანზაქტები. ისინი ელოდებიან თავისი ოჯახის ტრანზაქტების მოსვლას.

ASSEMBLE ბლოკში ტრანზაქტების შესვლისას მოწმდება: არის თუ არა სინქრონიზაციის სიაში ტრანზაქტების ასეთი ოჯახი. შესაძლებელია ორი შემთხვევის არსებობა. მოცემული ოჯახის ტრანზაქტები სინქრონიზაციის სიაში არსებობს ან არ არსებობს.

დავუშვათ, რომ ასეთი ოჯახის ტრანზაქტები სინქრონიზაციის სიაში არ არსებობს, ანუ ოჯახის ნაკრებებიდან პირველად შესული ტრანზაქტი არ არის. მაშინ გამოითვლება **A** ოპერანდი, დამრგვალდება მთელ რიცხვამდე და შემცირდება 1-ით. **S** - მიღებული შედეგის დამოკიდებულების მიხედვით შენახულ უჯრედში ოჯახის ნაკრების პირველი უჯრედისათვის სრულდება შემდეგი მოქმედებები:

- **S** < 0 – წარმოიქმნება შემდეგი სახის შეცდომა - „ნაკრების მრიცხველი უარყოფითია“;
- **S** = 0 – ტრანზაქტი ეცდება მომდევნო ბლოკში შესვლას (ანუ საჭიროა მხოლოდ ერთი ტრანზაქტის შეკრება);

- **S** ≥ 1 – ტრანზაქტი გამოაკლდება მიმდინარე მოვლენის სიას და მოთავსდება **ASSEMBLE** ბლოკის სინქრონიზაციის სიაში.

ახლა დავუშვათ, რომ სინქრონიზაციის სიაში უკვე არსებობს **ASSEMBLE** ბლოკში შესული იმავე ოჯახის ტრანზაქტი. ხოლო ნაკრების შენარჩუნებულ უჯრედში სინქრონიზაციის სიაში მოთავსებული ტრანზაქტის დროს შემცირდება 1-ით. როდესაც ამ ნაკრების მრიცხველი ხდება ნულის ტოლი, ანუ **ASSEMBLE** ბლოკში შევა ტრანზაქტების მოცემული რიცხვი, მაშინ მომლოდინე ტრანზაქტი გამოვა სინქრონიზაციის სიიდან. თუ ამ ტრანზაქტის მომსახურება არ იყო შეწყვეტილი არც ერთი მოწყობილობიდან, ის ეცდება გადავიდეს მომდევნო ბლოკში და წარმატების შემთხვევაში დაბრუნდება მიმდინარე მოვლენათა სიაში.

მაგალითად:

ASSEMBLE 7

აგროვებს ერთი ოჯახის 7 ტრანზაქტს, აქედან 6 გაქრება, ხოლო ერთი გადავა მომდევნო ბლოკში

ASSEMBLE *1

აგროვებს ტრანზაქტების რიცხვს, რომელიც უდრის **ASSEMBLE** ბლოკში ტრანზაქტების ოჯახის პირველად შესული 1 პარამეტრის მნიშვნელობას. ნაკრების მრიცხველის ასეთი წესით

მოცემა მოსახერხებელია მომხმარებლისათვის, როდესაც წინასწარ არის ცნობილი შესაგროვებელი ტრანზაქტების რიცხვი. ეს განისაზღვრება მოდელირების პროცესში. მაშასადამე, ყველა შესავსები ტრანზაქტის ნომერი ასეთ პარამეტრში ჩაიწერება ნაკრების მრიცხველის მნიშვნელობის სახით. ასე რომ არ არის ცნობილი თუ რომელი ტრანზაქტი მოვა პირველი და დააყენებს ნაკრების მრიცხველს.

შენიშვნა 1: შეწყვეტილმა ტრანზაქტმა უნდა დატოვოს **ASSEMBLE** ბლოკი იქამდე, ვიდრე მისი შეწყვეტილი მომსახურება არ დასრულდება. ამიტომ, რეკომენდებულია ისე აიგოს მოდელი, რომ ტრანზაქტები, რომელთა მომსახურება შეწყდა წაშლის გარეშე (მოწყობილობის განთავისუფლების გარეშე) არ შევიდეს **ASSEMBLE** ბლოკში.

შენიშვნა 2: **A** ოპერანდი **ASSEMBLE** ბლოკში შემდეგნაირად გამოიყურება:

ASSEMBLE 0· 7

ASSEMBLE -0· 7

ASSEMBLE 5· 7

ტრანსლაციის ეტაპზე შეცდომები არ აღმოჩნდება. პირველი ორი ჩანაწერის შემდეგ მოდელირების პროცესში აღმოჩნდება შემდეგი სახის შეცდომა: „ნაკრების მრიცხველი უარყოფითია“.

9.2.2. GATHER ბლოკი

GATHER (შეკრების) ბლოკი გამოიყენება ერთი ოჯახის მოცემული რაოდენობის ტრანზაქტების შესაკრებად, რომლებიც მოძრაობს ერთი და იმავე გზით. **GATHER** და **ASSEMBLE** ბლოკებს შორის განსხვავება მდგომარეობს იმაში, რომ **GATHER** ბლოკში ტრანზაქტები არ ქრება. შეკრების შემდეგ ყველა მიემართება მომდევნო ბლოკისაკენ. **GATHER** ბლოკს გააჩნია შემდეგი ფორმატი:

GATHER A

A ოპერანდი მიუთითებს ერთი ოჯახის ტრანზაქტების იმ რიცხვს, რომლებიც უნდა შეიკრიბოს მათი ერთი გზით მოძრაობისათვის (ნაკრების მრიცხველის საწყისი მნიშვნელობა). ეს ოპერანდი შესაძლოა იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა*პარამეტრი. **GATHER** ბლოკს ასევე შეუძლია ერთდროულად აწარმოოს რამდენიმე ოჯახის ტრანზაქტების შეკრება და გააჩნია სინქრონიზაციის სია.

დავუშვათ, რომ **GATHER** ბლოკში ტრანზაქტის შესვლის შემოწმების შედეგად აღმოჩნდა, რომ მისი სინქრონიზაციის სიაში არ აღმოჩნდა აღნიშნული ოჯახის ტრანზაქტები. ანუ, მოცემული ოჯახიდან შესული ტრანზაქტი წარმოადგენს პირველს. მაშინ გამოითვლება **A** ოპერანდი, დამრგვალდება მთელ რიცხვამდე,

შემცირდება 1-ით და მიღებული შედეგი დამახსოვრდება ტრანზაქტის უჯრედში – ნაკრების მრიცხველში. **S** შედეგის მიხედვით შესაძლებელია შემდეგი მოქმედებები:

- **S** < 0 – წარმოიქმნება შემდეგი სახის შეცდომა - „ნაკრების მრიცხველი უარყოფითია“;
- **S** = 0 – ტრანზაქტი ეცდება მომდევნო ბლოკში შესვლას (ანუ საჭიროა მხოლოდ ერთი ტრანზაქტის შეკრება);
- **S** ≥ 1 – ტრანზაქტი მოთავსდება **GATHER** ბლოკის სინქრონიზაციის სიაში და დაელოდება თავისი ოჯახის სხვა ტრანზაქტების შემოსვლას.

ახლა დავუშვათ, რომ სინქრონიზაციის სიაში შემოწმების შედეგად აღმოჩნდა, რომ **GATHER** ბლოკში შესული ტრანზაქტი მიეკუთვნება მოცემულ ოჯახს. იგი აუცილებლად მოთავსდება სინქრონიზაციის სიაში, ხოლო ნაკრების მრიცხველი შემცირდება 1-ით. როდესაც მისი მნიშვნელობა აღმოჩნდება ნულის ტოლი, ანუ **GATHER** ბლოკში შევიდა მოცემული ოჯახის ტრანზაქტების გარკვეული რიცხვი, ყველა შეკრებილი ტრანზაქტი გამოაკლდება სინქრონიზაციის სიას და მოთავსდე-ბა მიმდინარე მოვლენათა სიაში.

შენიშვნა 1: თუ ერთი ოჯახის შეკრებილ ტრანზაქტებს შორის აღმოჩნდა შეწყვეტილი ტრანზაქტები, მაშინ შეკრების შემდეგ ისინი არ გადავა ნაკრების მრიცხველში. შეწყვეტილი ტრანზაქტები იმყოფება **GATHER** ბლოკში, მაგრამ არა სინქრონიზაციის სიაში, ვიდრე ყველა შეწყვეტილი ტრანზაქტის მომსახურება არ დასრულდება. ამიტომ, მოდელში გათვალისწი-

ნებული უნდა იქნას, რომ შეწყვეტილი ტრანზაქტები, რომლებიც არ იქნა წაშლილი მოდელიდან იკავებს მოწყობილობას და ისინი არ შევა **GATHER** ბლოკში.

შენიშვნა 2: **GATHER** ბლოკის **A** ოპერანდი შემდეგნაირად გამოიყურება:

GATHER 0· 7

GATHER -0· 7

GATHER 5· 7

ტრანსლაციის ეტაპზე არ აღმოჩნდება შეცდომები. პირველი ორი ჩანაწერის შემდეგ მოდელირების პროცესში გამოვა შემდეგი სახის შეცდომა: „ნაკრების მრიცხველი უარყოფითია“.

9.2.3. MATCH ბლოკი

MATCH (სინქრონიზაციის) ბლოკი განკუთვნილია ერთი ოჯახის ტრანზაქტების სინქრონიზებული მოძრაობისათვის, რომლებიც გადაადგილდება სხვადასხვა გზით. სინქრონიზაციისათვის აუცილებელია **MATCH** ორი ბლოკის არსებობა, რომლებიც იმყოფება მოდელის შესაბამის ადგილებში და ეწოდებათ *დაკავშირებული (შეუღლებული) ტრანზაქტები.* **MATCH** ბლოკს გააჩნია შემდეგი სახე:

Name MATCH A

MATCH ბლოკის ყოველი **A** ოპერანდი მიუთითებს შეუღლებული ბლოკის ჭდეს ან ოპერანდის ნომერს. აღნიშნული ოპერანდი შესაძლებელია იყოს სახელი, დადებითი მთელი რიცხვი, ფრჩხილებში ჩასმული გამოსახულება, სტრა ან სტრა* პარამეტრი. მაგალითად:

Kont1 MATCH Kont 2
Kont2 MATCH Kont 1

მოდელში აღნიშნული ბლოკები მოთავსდება ცალ-ცალკე ტრანზაქტების პარალელური გზით გადასაადგილებლად. **MATCH** ბლოკში ტრანზაქტების შეღწევისას **A** ოპერანდი გამოითვლება და დამრგვალდება მთელ რიცხვამდე.

მიღებული მნიშვნელობების მიხედვით განისაზღვრება **MATCH** ბლოკის შეუღლებული ბლოკი. შედეგის არარსებობის შემთხვევაში გამოიტანება შეცდომა.

თუ შეუღლებული ბლოკები არსებობს, მაშინ მოწმდება მათში რომელიმე ოჯახის ტრანზაქტის არსებობა. თუ **MATCH** შეუღლებულ ბლოკში არ იმყოფება მოცემული ოჯახის არც ერთი ტრანზაქტი, მაშინ ბლოკში შესული ტრანზაქტი მოთავსდება სინქრონიზაციის სიაში და დაელოდება თავისი ოჯახის ტრანზაქტის შემოსვლას. **MATCH** შეუღლებულ ბლოკში ასეთი ტრანზაქტის მოცემული ოჯახის ორივე ტრანზაქტი გამოირიცხება

სინქრონიზაციის სიიდან და ერთდროულად გადაადგილდება **MATCH** ბლოკის მომდევნო ბლოკებში.

აგრეთვე, შესაძლებელია ტრანზაქტების რამდენიმე წყვილი სინქრონიზებულად გადაადგილდეს **MATCH** ბლოკის რამდენიმე წყვილში.

შენიშვნა 1: თუ სინქრონიზებული ტრანზაქტებიდან რომელიმე ტრანზაქტის მომსახურება შეწყვეტილი აღმოჩნდება, მაშინ იგი არ გამოვა აღნიშნული ბლოკიდან, ვიდრე არ დასრულდება ყველა შეწყვეტილი ტრანზაქტის მომსახურება.

ამიტომ, რეკომენდებულია ისე იქნას აგებული მოდელი, რომ ტრანზაქტები, რომელთა მომსახურებაც შეწყდა წაშლის გარეშე, არ შევიდეს **MATCH** ბლოკში.

შენიშვნა 2: **MATCH** ბლოკი შესაძლებელია შეუღლებული იყოს თავის თავთან, მაგალითად:

Six MATCH Six

ამ შემთხვევაში მისი მოქმედება **A** ოპერანდით მოცემული **GATHER** ბლოკის ანალოგიური აღმოჩნდება და უდრის 2-ს.

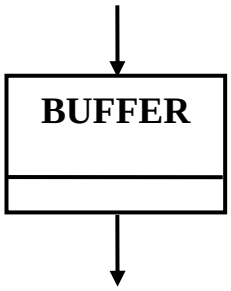
9.2.4. მიმდინარე მოვლენის ჯაჭვის დათვალიერების მართვა. BUFFER ბლოკი

ძალზე ხშირად აქტიური ტრანზაქტის გავლისას რომელიმე განსაზღვრული კატეგორიის ბლოკებში, ინტერპრეტატორი იწყებს მიმდინარე მოვლენის ჯაჭვის თავიდან შემოწმებას, მას შემდეგ რაც აღნიშნული ტრანზაქტი შეჩერდება. ქვემოთ მოცემულია წინა თავში აღწერილი ბლოკები, რომელთაც მივყავართ აღნიშნულ ეფექტამდე: **SEIZE, RELEASE, ENTER, LEAVE, LOGIC, PRIORITY.**

დათვალიერების განახლება არ წარმოებს, ვიდრე აქტიური ტრანზაქტი არ შეჩერდება რომელიმე ბლოკში. ტრანზაქტი არ წყვეტს თავის მოძრაობას, ვიდრე არ შევა **ADVANCE** ბლოკში ან **TERMINATE** ბლოკში, ან არ შეხვდება მაბლოკირებელ პირობას.

ამგვარად, დათვალიერება არ წარმოებს, ვიდრე აქტიური ტრანზაქტი არ გაივლის ერთ ან რამდენიმე დამატებით ბლოკს.

ხშირად წარმოიქმნება ისეთი შემთხვევები, როდესაც გარკვეულ წერტილამდე ტრანზაქტის მიღწევისას მომხმარებელმა უნდა დაიწყოს მიმდინარე მოვლენის ჯაჭვის შემოწმება, და არ დაელოდოს, ვიდრე აღნიშნული ტრანზაქტი გაივლის დამატებით ბლოკებს. აღნიშნულის განხორციელება შესაძლებელია **BUFFER** (დათვალიერების განახლება) ბლოკის მეშვეობით. იგი გრაფიკულად შემდეგნაირად გამოიყურება:



ნახ. 9.1. **BUFFER** ბლოკი

აღნიშნულ ბლოკს ოპერანდი არ გააჩნია.

როდესაც ტრანზაქტი შედის **BUFFER** ბლოკში, **GPSS**-ის ინტერპრეტატორი წყვეტს მის მოძრაობას და მაშინვე იწყებს მიმდინარე მოვლენის ჯაჭვის თავიდან შემოწმებას. ბუფერში მოთავსებული ტრანზაქტი (ანუ **BUFFER** ბლოკში შესული) რჩება მიმდინარე მოვლენის ჯაჭვში და ინარჩუნებს იმავე მდგომარეობას, რაც გააჩნდა აღნიშნულ ბლოკში შესვლამდე. ახალი დათვალიერების გასაგრძელებლად ბუფერში მოთავსებული ტრანზაქტი ინტერპრეტატორის მიერ თავიდან იქნება დამუშავებული და ამ მომენტიდან აღდგება მოდელში შემდგომი მოძრაობა.

9.3. მოდელებში ირიბი მისამართის მეთოდით ობიექტების რიცხვის შემცირება

აქამდე განიხილებოდა მოდელებში **GPSS**-ის სხვადასხვა ობიექტებზე მიმართვა მათი სახელის მიხედვით. ასეთი პირდაპირი მისამართით მიმართვა ხელსაყრელია, მაშინ როდესაც ლაპარაკია

თითოეული ტიპის მცირე რაოდენობის ობიექტების შესახებ. თუ ობიექტების რიცხვი დიდია, მაშინ პროგრამის შესამცირებლად მოდელში იყენებენ ამ ობიექტებზე მიმართვას მათი ნომრების მიხედვით ირიბი დამისამართების მეთოდის გამოყენებით.

ირიბი დამისამართების იდეა მდგომარეობს იმაში, რომ თითოეული ტრანზაქტი რომელიმე თავის პარამეტრში შეიცავს თავის ან სხვა ობიექტის ნომერს, ხოლო ბლოკების ოპერანდებში ჩაიწერება მიმართვა ტრანზაქტის ამ პარამეტრზე. ირიბი დამისამართების საერთო სახე მოცემული იყო 1.4.2 პარაგრაფში: სტრა*პარამეტრი.

მაგალითად:

SEIZE P*X1

ერთარხიანი მოწყობილობის დაკავება, რომლის ნომერი შედის პარამეტრში, ხოლო ამ პარამეტრის ნომერი განისაზღვრება **X1** უჯრედის მნიშვნელობით.

SAVEVALUE 1, X*P2

1 ნომრით მითითებულ უჯრედში მოვათავსოთ მნიშვნელობა, რომლის ნომერი განისაზღვრება ტრანზაქტის მეორე პარამეტრის მნიშვნელობით.

X თავი

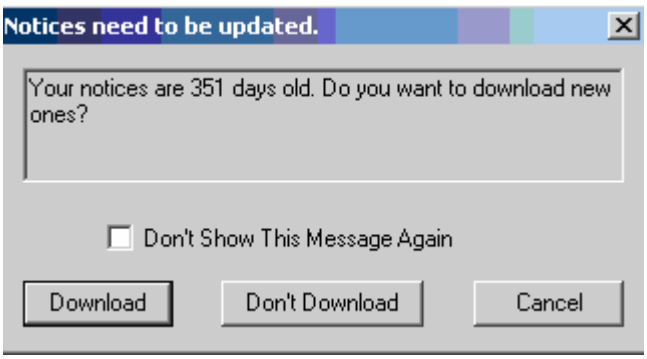
GPSS World-ში მოდელების დამუშავება და ექსპლუატაცია

10.1. ობიექტი „მოდელი“-ს შექმნა

10.1.1. GPSS World-ის გაშვება

GPSS World-ის გაშვებისათვის აუცილებელია თქვენს კომპიუტერში გადმოტვირთოთ **MinutemanSoftware.com** საიტიდან **GPSS World Student Version 5.2.2.0** ვერსია. მისი დაყენების შემდეგ სამუშაო მაგიდაზე გაჩნდება შესაბამისი იარლიყი და მასზე დაწკაპუნებისას გაიშვება **GPSS World** [17].

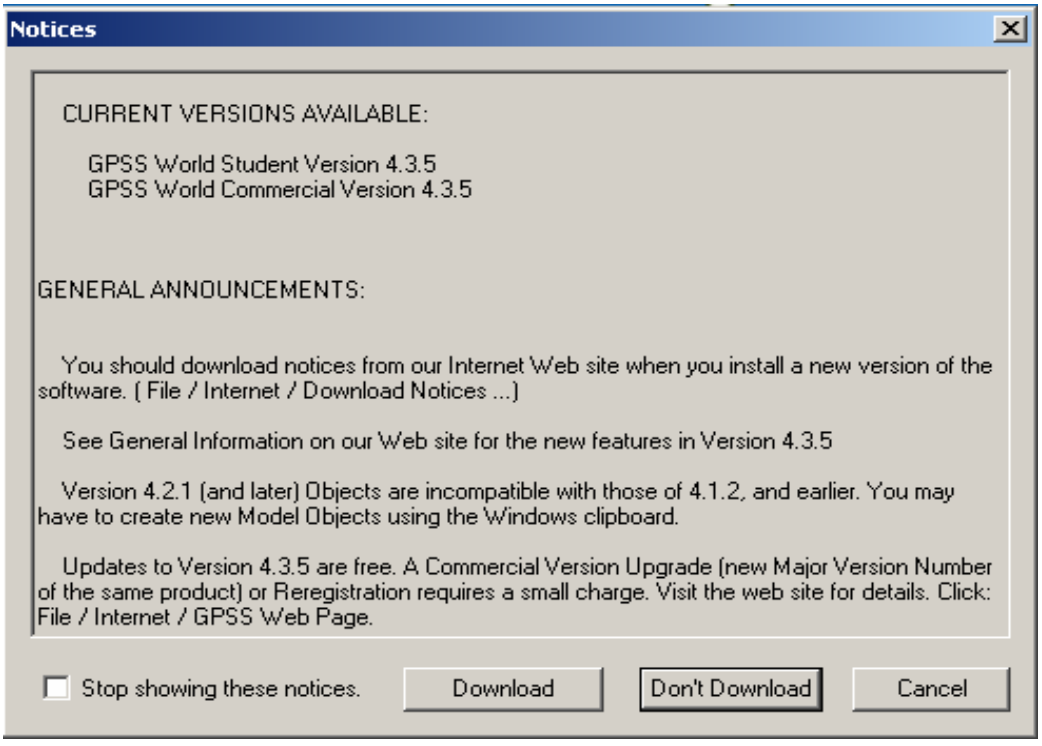
თუ გავიდა მნიშვნელოვანი დრო **MinutemanSoftware.com** საიტიდან „შენიშვნების“ ბოლო ჩატვირთვის მომენტიდან გამოჩნდება შემდეგი ფანჯარა (ნახ. 10.1), რომელიც შეგახსენებთ, რომ აუცილებელია აღნიშნული შენიშვნების განახლება.



ნახ. 10.1. „შენიშვნების“ განახლების აუცილებლობის შესახებ შეხსენების ფანჯარა

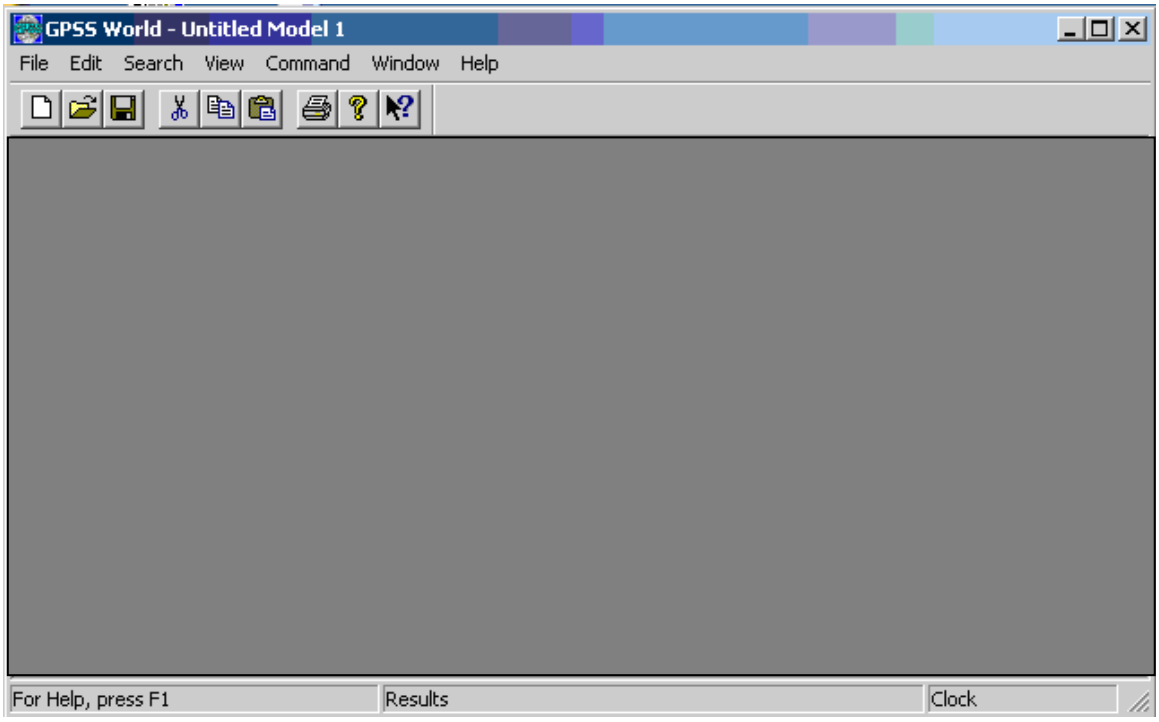
Download ღილაკზე დაწკაპუნებისას გაიხსნება ინტერნეტის გვერდი, რომლის დახმარებით შესაძლებელია მიმდინარე „შენიშვნების“ ჩატვირთვა. პრაქტიკულად ეს შენიშვნები შესაძლებელია ჩაიტვირთოს ნებისმიერ მომენტში მენიუს ბრძანების **File ►Internet ►GPSS Web Page ...** საშუალებით. ორივე შემთხვევაში საჭიროა მიუთითოთ საქაღალდე, სადაც გადმოიწერება ფაილები, მათ შორის მოთავსდება **GPSS World**-ის გამშვები ფაილი. ამასთან ერთად უნდა დარწმუნდეთ რომ **GPSSW.CNT** არსებული ფაილი ახლით შეიცვალა.

Notices (შენიშვნების) ფანჯარა გამოჩნდება დამოუკიდებლად (ნახ. 10.2). ის შეიცავს განახლების შესახებ არსებულ ინფორმაციას და სხვა სასარგებლო რჩევებს.



ნახ.10.2. **Notices** (შენიშვნების) ფანჯარა

Notices ფანჯრის დახურვის შემდეგ გამოჩნდება **GPSS World**-ის მთავარი ფანჯარა (ნახ. 10.3).



ნახ. 10.3. **GPSS World**-ის მთავარი ფანჯარა

მთავარი ფანჯარა შედგება მრავალი კომპონენტისაგან. ფანჯრის დასაწყისში მოთავსებულია სათაურის პანელი (ანუ სტრიქონი), ქვემოთ მენიუს პანელი და კიდევ ქვემოთ ინსტუმენტების პანელი. შემდეგ კი – მოდელის (ანუ სამუშაო) არე. მთავარი ფანჯრის სულ ბოლოს მოთავსებულია მდგომარეობის სტრიქონი, რომელიც გაყოფილია სამ ნაწილად. მდგომარეობის სტრიქონის მარცხენა ნაწილი გვიჩვენებს ინფორმაციას მენიუს პუნქტების გამოყენების შესახებ. მდგომარეობის სტრიქონის შუა ნაწილი გვაწვდის ინფორმაციას შეცდომების შესახებ და მდგომარეობის სტრიქონის მარჯვენა ნაწილი ასახავს სამოდელო დროს მოდელირების პროცესის შესრულებისას. „მოდელირების პროცესის“ თითოეული ობიექტისათვის სამოდელო დროის (წამმზომის) ჩართვა ან

გამორთვის ჩვენება მიმდინარეობს **View ►Simulation Clock** ალმის დაყენებით ან მოხსნით.

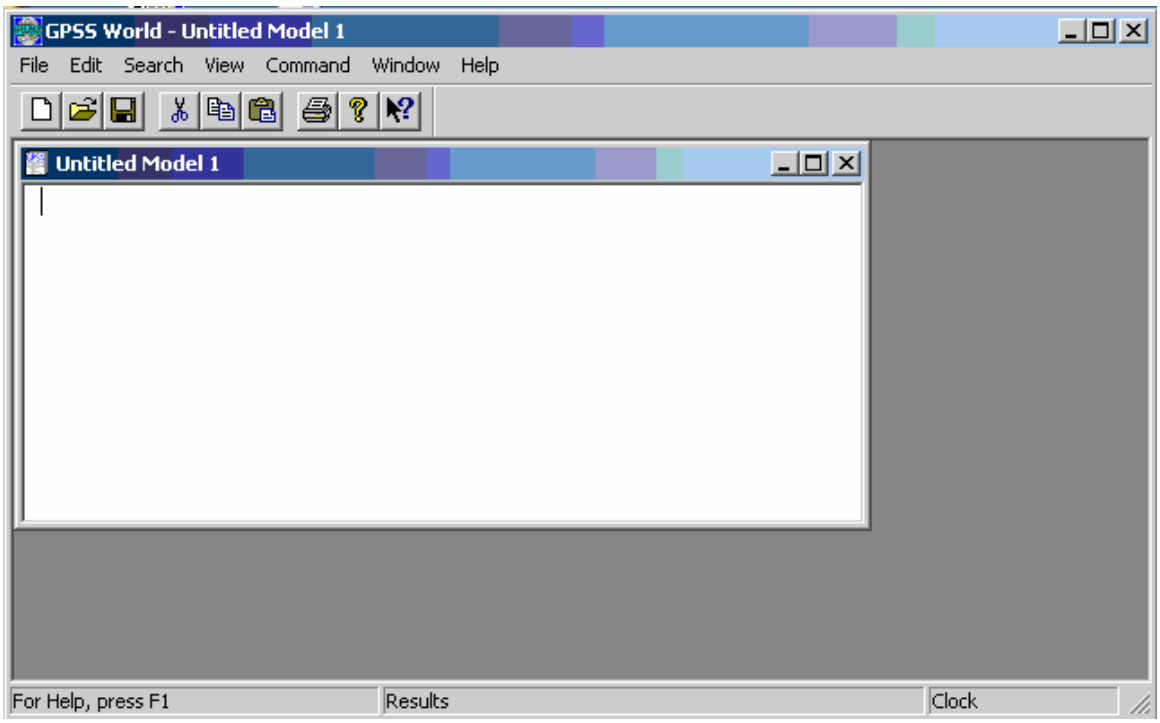
GPSS World-ის ყველა ობიექტს გააჩნია მენიუ და თავისი საკუთარი ფანჯარა [9]. მოქმედებების უმეტესობა იშვება მთავარი ფანჯრიდან ძირითადი მენიუს (**File, Edit, Search, View, Command, Window, Help)** დახმარებით. ძირითადი მენიუს თითოეულ პუნქტს გააჩნია ქვემენიუს ბრძანებები, მათგან ზოგიერთი ხელმისაწვდომია მხოლოდ განსაზღვრულ შემთხვევებში გამოყენებისათვის. ეს სრულდება სწორი მოქმედებების შესრულების შემდეგ დახმარების გასაწევად. მიუწვდომელ მდგომარეობაში (გამორთულ) მენიუს ბრძანებები გამოსახულია რუხი ფერით და მათი არჩევა შეუძლებელია.

10.1.2. მოდელში ოპერატორების შეტანა

File ►New მენიუს ბრძანების არჩევის შემდეგ გამოჩნდება მენიუ [9]:

ნახ. 10.4. „მოდელი“ ტიპის ობიექტის ან ტექსტური ობიექტის ასარჩევი მენიუ

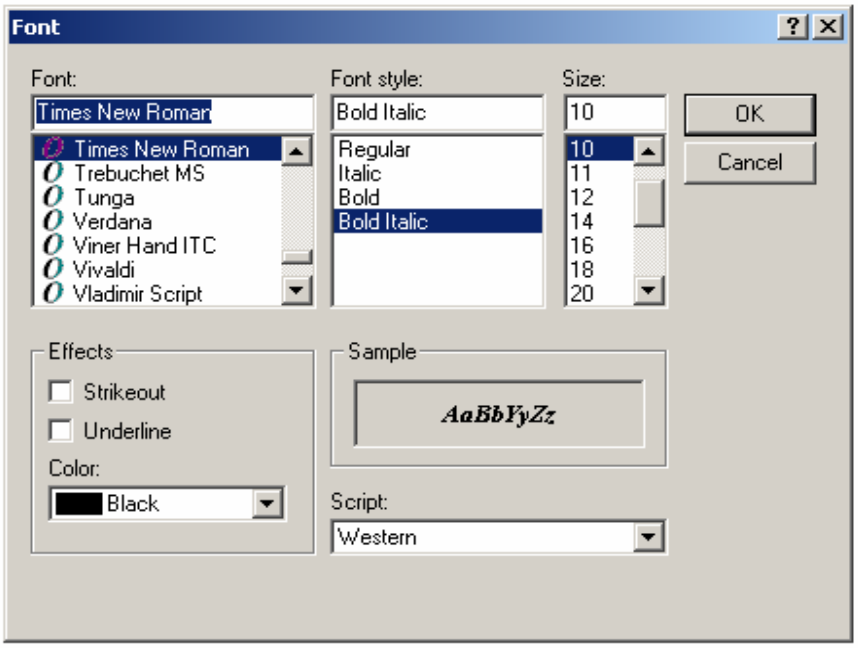
GPSS World-ს ტექსტის დასამუშავებლად გააჩნია ტექსტური რედაქტორი. თუ საჭიროა ობიექტი „მოდელი“-ს შექმნა, მაშინ აირჩიეთ **Model** და გაიხსნება ტექსტური რედაქტორის ფანჯარა (ნახ. 10.5):



ნახ. 10.5. ტექსტური რედაქტორის ფანჯარა

ტექსტურ რედაქტორში ფარულად გამოიყენება **Courier New** შრიფტი. ეს შრიფტი ამარტივებს ტექსტის სვეტების გათანაბრებას მოდელში და განსაკუთრებით სტანდარტული ანგარიშისას, რადგან ამ შრიფტის თითოეულ სიმბოლოს გააჩნია თანაბარი სიგანე. შესაძლებელია ამ შრიფტის შეცვლა. 5.1 მაგალითის სათაური გაფორმებულია მუქი შრიფტით. შრიფტის შეცვლა შესაძლებელია შემდეგნაირად: აირჩიეთ **File ► Font** მენიუს ბრძანება და გამოჩნდება ფანჯარა, რომელშიც შესაძლებელია

საჭირო შრიფტის დაყენება. დააყენეთ **AcadNusx, Bold Italic, 10** (ნახ. 10.6).

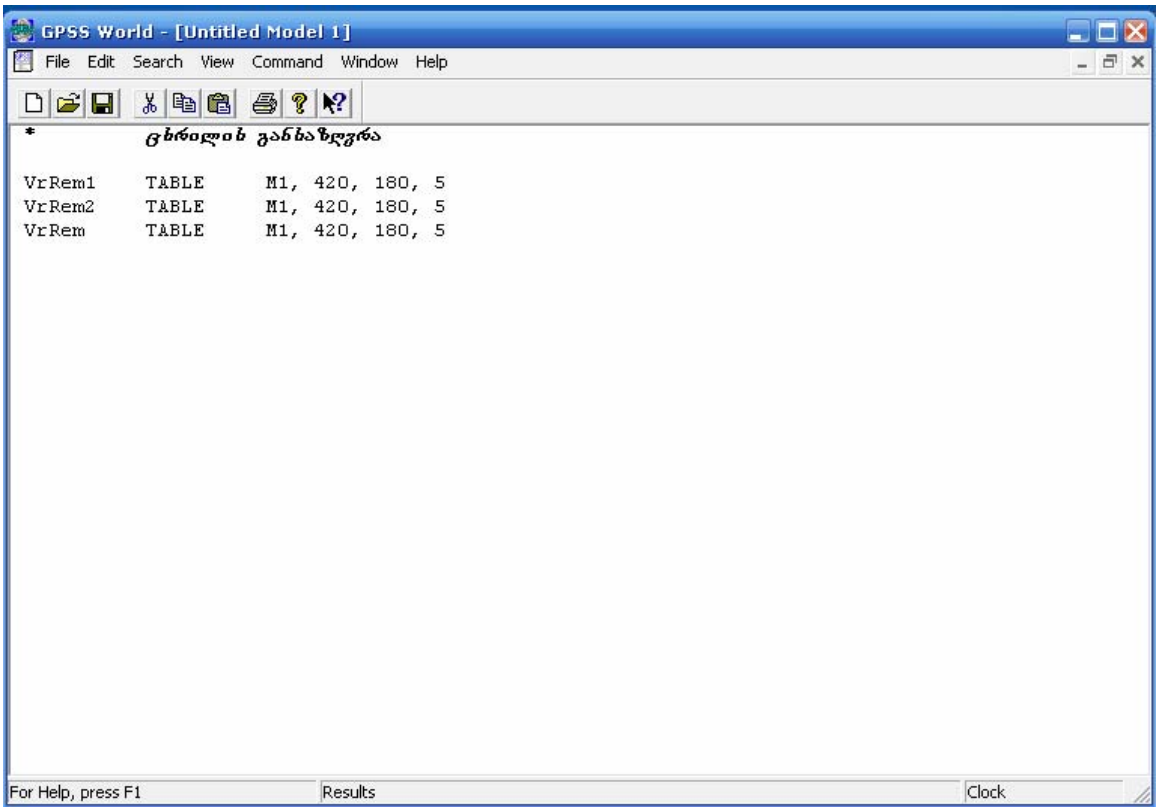


ნახ. 10.6. შრიფტის დაყენების ფანჯარა

ამის შემდეგ აირჩიეთ **Courier New** შრიფტი, შეიტანეთ ცხრილის სახელი **VRem,** შემდეგ **TABLE** და მისი ოპერანდები. გადასვლა განახორციელეთ **Tab** ტაბულაციის ღილაკის დახმარებით. მესამე სტრიქონის ანალოგიურია მეოთხე და მეხუთე სტრიქონები, განსხვავება მხოლოდ სახელის ბოლოს ნომერშია ამისათვის კი გამოიყენეთ **Edit ►Copy** და **Edit ►Paste** ბრძანებები.

ტექსტის გამოყოფილი ფარგმენტის ამოჭრის, ასლის შექმნისა (კოპირების) და ჩასმის ბრძანებებს გააჩნიათ იგივე დანიშნულება, როგორც სხვა ტექსტურ რედაქტორებში და მათ ნაცვლად შესაძლებელია **[Ctrl] + [X]**, **[Ctrl] + [C]**, **[Ctrl] + [V]** ღილაკების კომბინაციების გამოყენება.

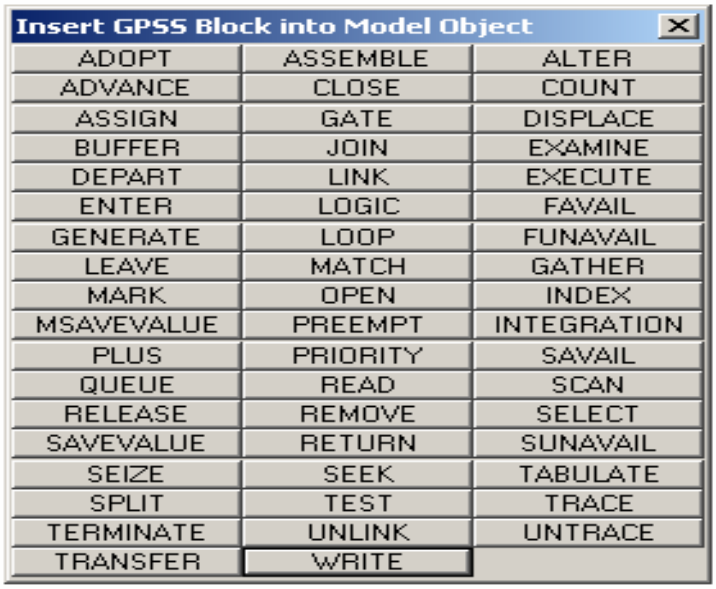
ცხრილის სახელის რედაქტირების შემდეგ მივიღებთ (ნახ. 10.7):



ნახ. 10.7. ტექსტური რედაქტორის ფანჯარა ცხრილის განსაზღვრის ბრძანებებით

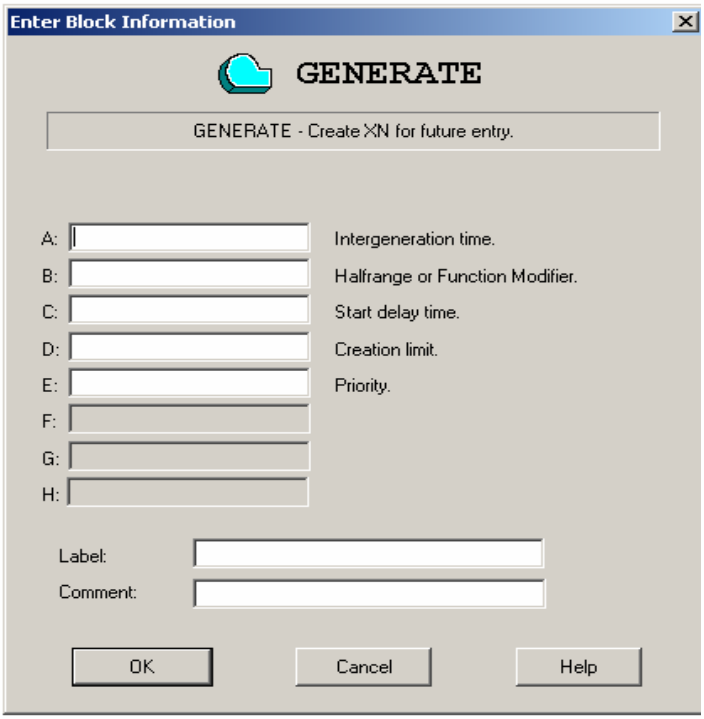
მოდელში მეშვიდე სტრიქონის შესატანად აირჩიეთ შრიფტი **AcadNusx, Bold, 10**.

მოდელში ოპერატორების შესატანად აირჩიეთ მენიუს ბრძანება **Edit ► Insert GPSS Bloks . . . ,** არჩევის შემდეგ გამოვა შემდეგი ფანჯარა (ნახ.10.8). გამოსულ ფანჯარაში აირჩიეთ **GENERATE** ბლოკი და დააწკაპუნეთ მაუსის მარცხენა ღილაკზე.

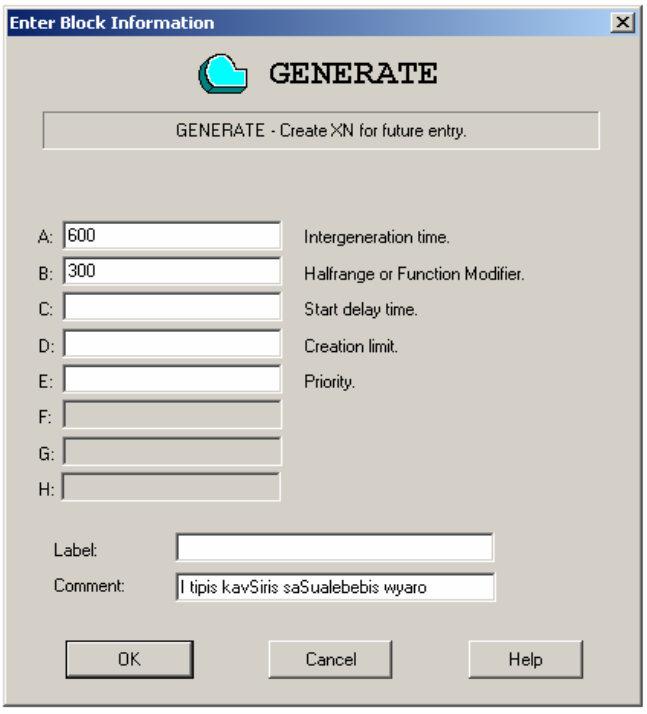


ნახ. 10.8. **GPSS**-ის ბლოკების მენიუ

გამოჩნდება **GENERATE** ბლოკის შექმნის ფანჯარა (ნახ. 10.9.), კურსორით **A** ოპერანდის ველში ჩაწერეთ 600, ხოლო **B** ოპერანდის ველში – 300.



ნახ. 10.9. **GENERATE** ბლოკის შექმნის ფანჯარა (ცარიელი)



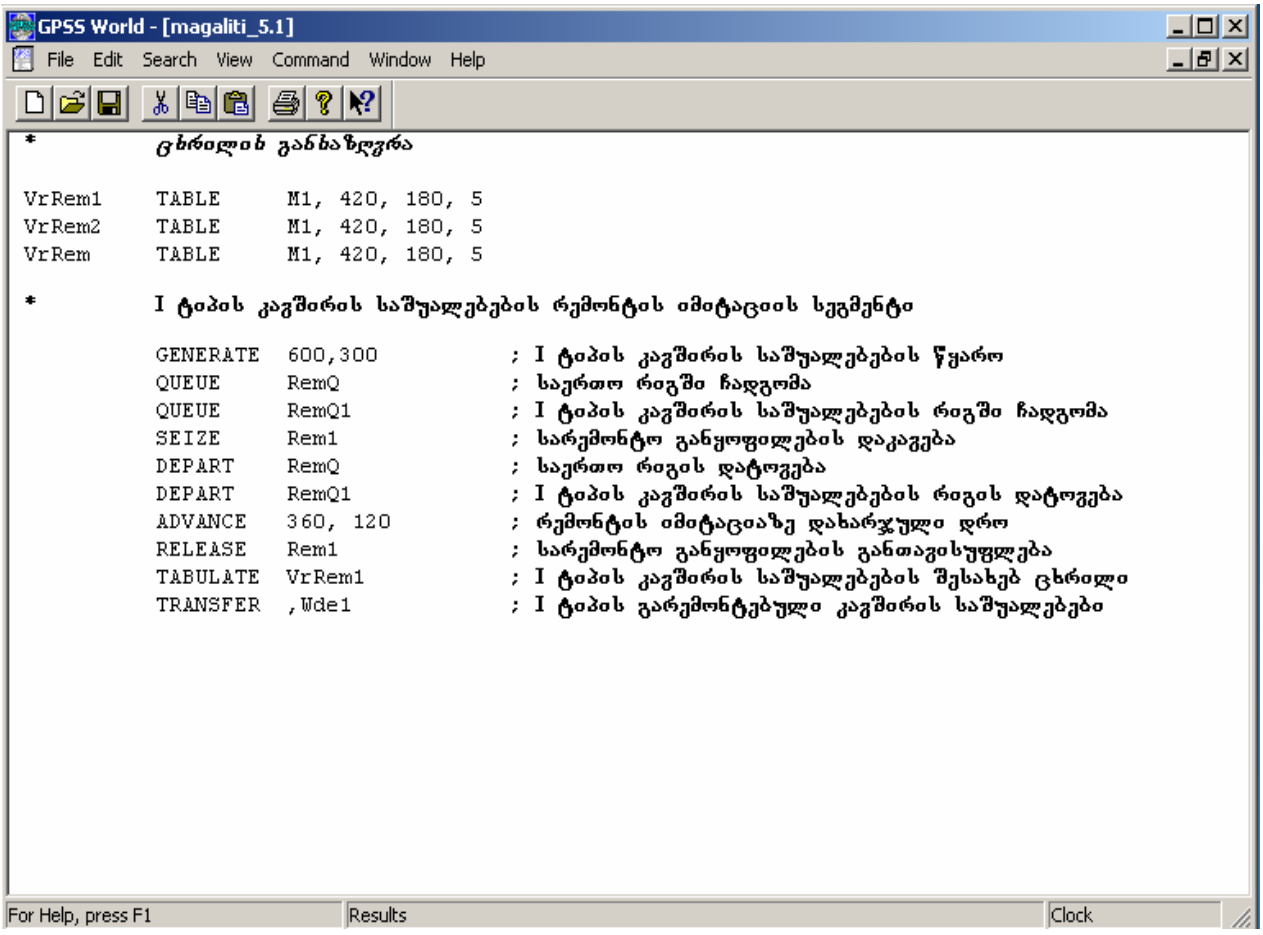
ნახ. 10.10. **GENERATE** ბლოკის შექმნის ფანჯარა (შევსებული)

ველებში გადასვლისათვის გამოიყენეთ **Tab** ღილაკი. შემდეგ **Tab** ღილაკით ან მაუსით გადადით **Comment** ველში და ჩაწერეთ შემდეგი კომენტარი: **I** ტიპის კავშირის საშუალებების წყარო (ნახ. 10.10).

ამის შემდეგ დააწკაპუნეთ **OK** ღილაკზე. მოდელში გაჩნდება ოპერატორი

GENERATE 600, 300 ; I ტიპის კავშირის საშუალებების წყარო

განაგრძეთ მოდელში ოპერატორების შეტანა და მიიღებთ შემდეგი სახის ფანჯარას (ნახ.10.11)

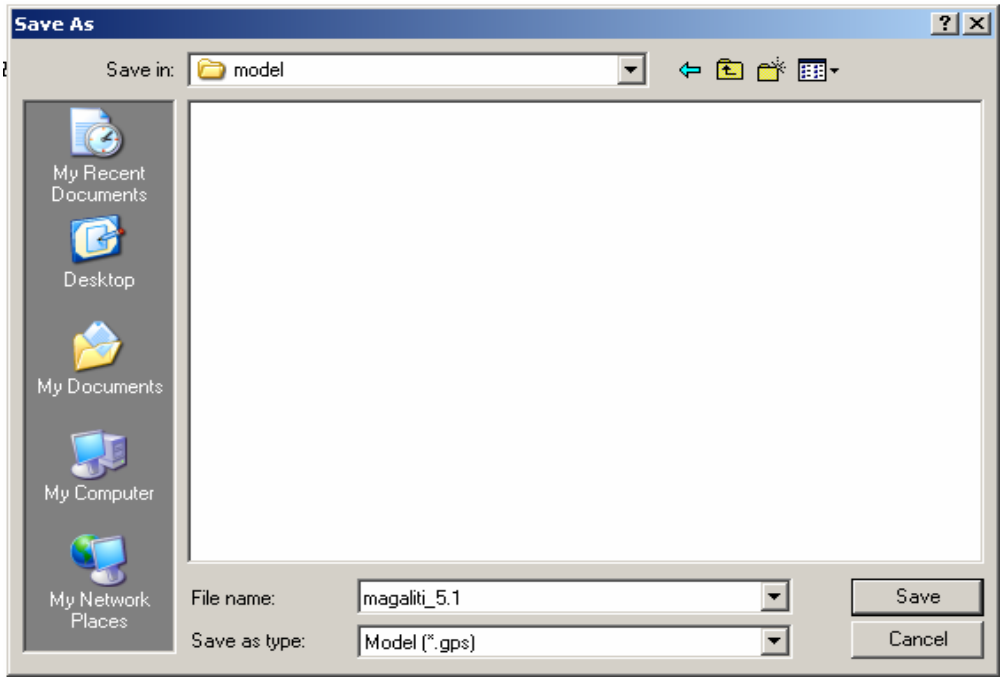


ნახ. 10.11. ტექსტური რედაქტორის ფანჯარაში

5.1 მაგალითის მოდელის ფრაგმენტი

ტექსტში სხვა ცვლილებების შესატანად გამოიყენება **Edit** მენიუს პუნქტის სხვა ბრძანებები, რომელთაც იგივე დანიშნულება აქვთ რაც სხვა ტექსტურ რედაქტორებში. არასწორად გაკეთებული ცვლილების გაუქმებისათვის ანუ ფაილის წინა მდგომარეობაში დასაბრუნებლად გამოიყენება მენიუს ბრძანება **Undo (Ctrl + Z).** ბრძანება **Delete Line (Ctrl + D)** შლის კურსორის მიერ მითითებულ სტრიქონს, ხოლო ბრძანება **Insert Line (Ctrl + I)** ამატებს კურსორის მიერ მითითებული სტრიქონის ბოლოს ახალ სტრიქონს.

„მოდელი“ ობიექტის შესანახად შექმენით საქაღალდე **Model.** შემდეგ აირჩიეთ **File Save As ...** . გამოსულ ფანჯარაში (ნახ. 10.12) **Untitled** სახელის ნაცვლად მიუთითეთ მაგალითი_5.1 და დააწკაპუნეთ **Save As ... (Ctrl + S)** ღილაკზე, ხოლო ფაილის გახსნისათვის აირჩიეთ ბრძანება **File ►Open (Ctrl + O)**.

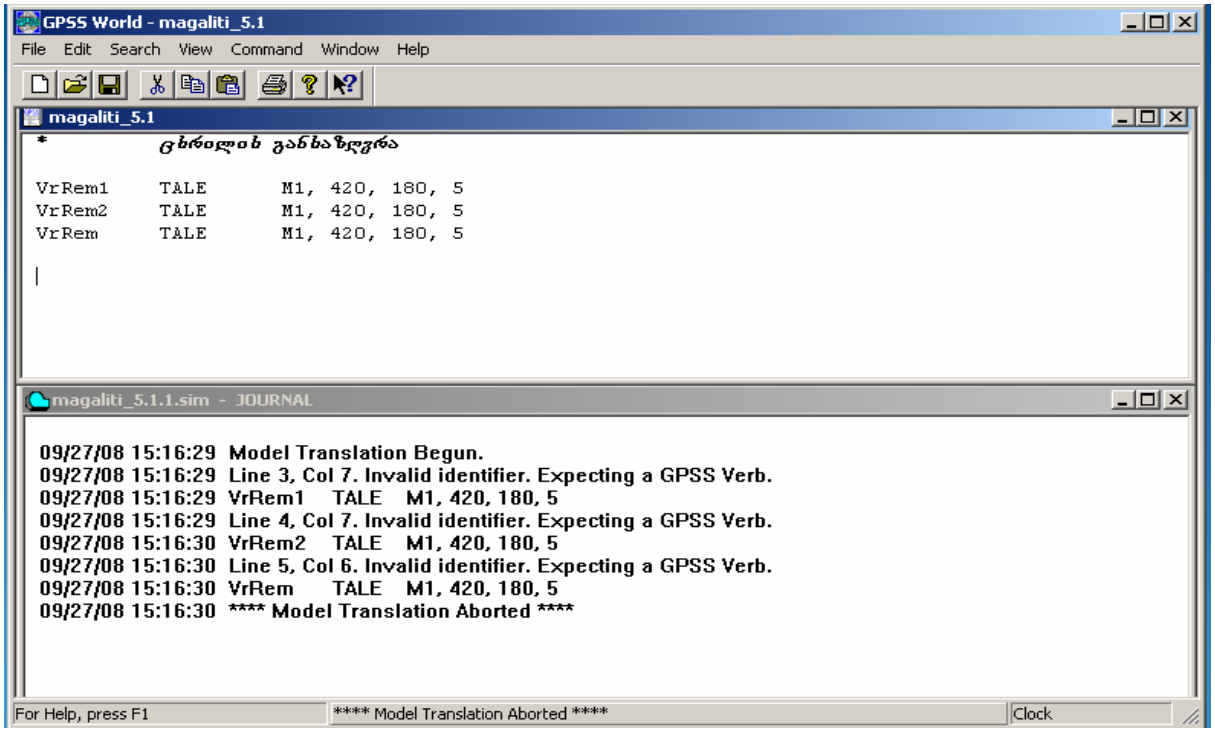


ნახ. 10.12. **Save As ...** დიალოგური ფანჯარა

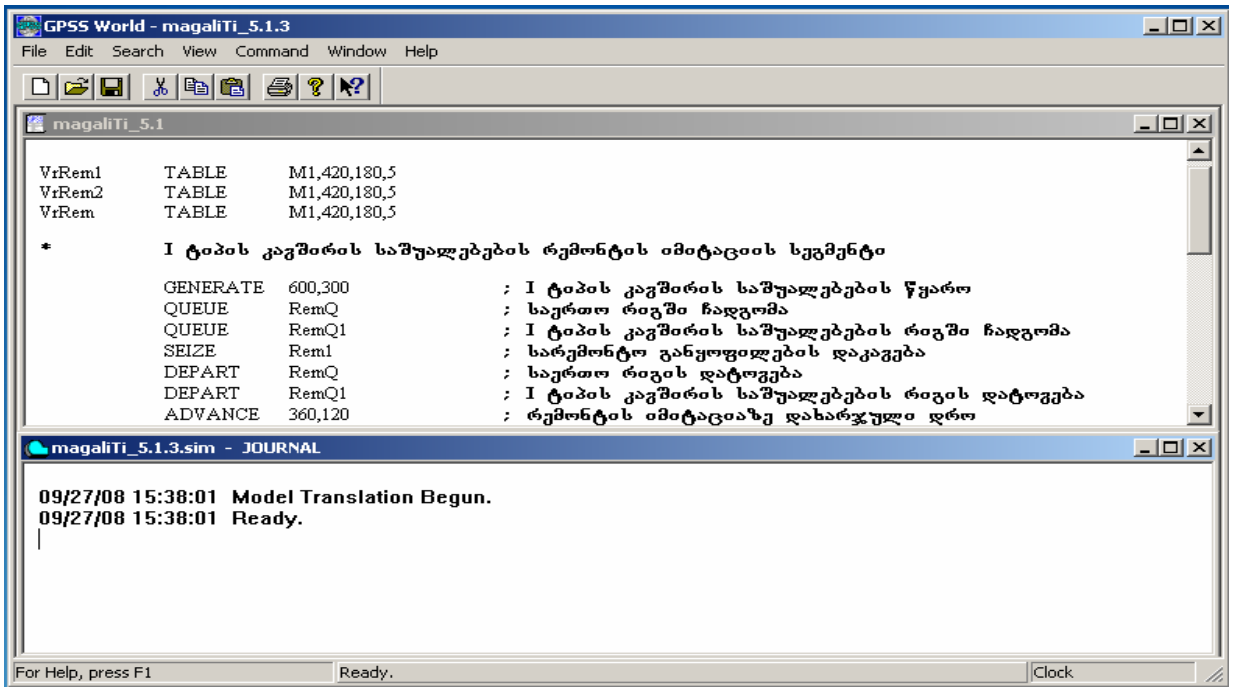
10.2. ობიექტი „მოდელირების პროცესის“ შექმნა

ობიექტი „მოდელირების პროცესი“ წარმოადგენს ობიექტი „მოდელი“-ს ტრანსლაციის შედეგად მიღებულ ობიექტს. ობიექტი „მოდელი“-ს ტრანსლაციისათვის აირჩიეთ ბრძანება **Command ►Greate Simulation(Ctrl + Alt + S).** ამ ბრძანების მიხედვით **GPSS** ტრანსლატორი ამოწმებს მოდელში სინტაქსური შეცდომების არსებობას. სინტაქსური შეცდომების არარსებობისას ტრანსლატორი ააქტიურებს ყველა ინტერაქტიულ ბრძანებებს და ფანჯრებს, რათა შესაძლებელი იყოს თვალყური ადევნოს მოდელირების პროცესის მდგომარეობას და მისით მართვას.

ხშირად ობიექტ „მოდელ“-ში ადგილი აქვს შეცდომებს. დავუშვათ, რომ 5.1 მაგალითში ადგილი აქვს შეცდომებს. მაგალითად, მესამე სტრიქონის შეტანისას **Table**-ს ნაცვლად იყო აკრეფილი **Tale** და შემდეგი ორი სტრიქონის გადმოწერისას იგივე სახის შეცდომები აქაც განმეორდა. ამიტომ ობიექტი „მოდელირების პროცესი“ არ შეიქნება. **JOURNAL** ფანჯარაში ტრანსლატორი გამოიტანს ტრანსლაციის შეცდომების შესახებ შეტყობინებების სიას (ნახ. 10. 13).



ნახ. 10.13. **JOURNAL** ფანჯარა ტრანსლაციის შეცდომების შესახებ შეტყობინებების სიით

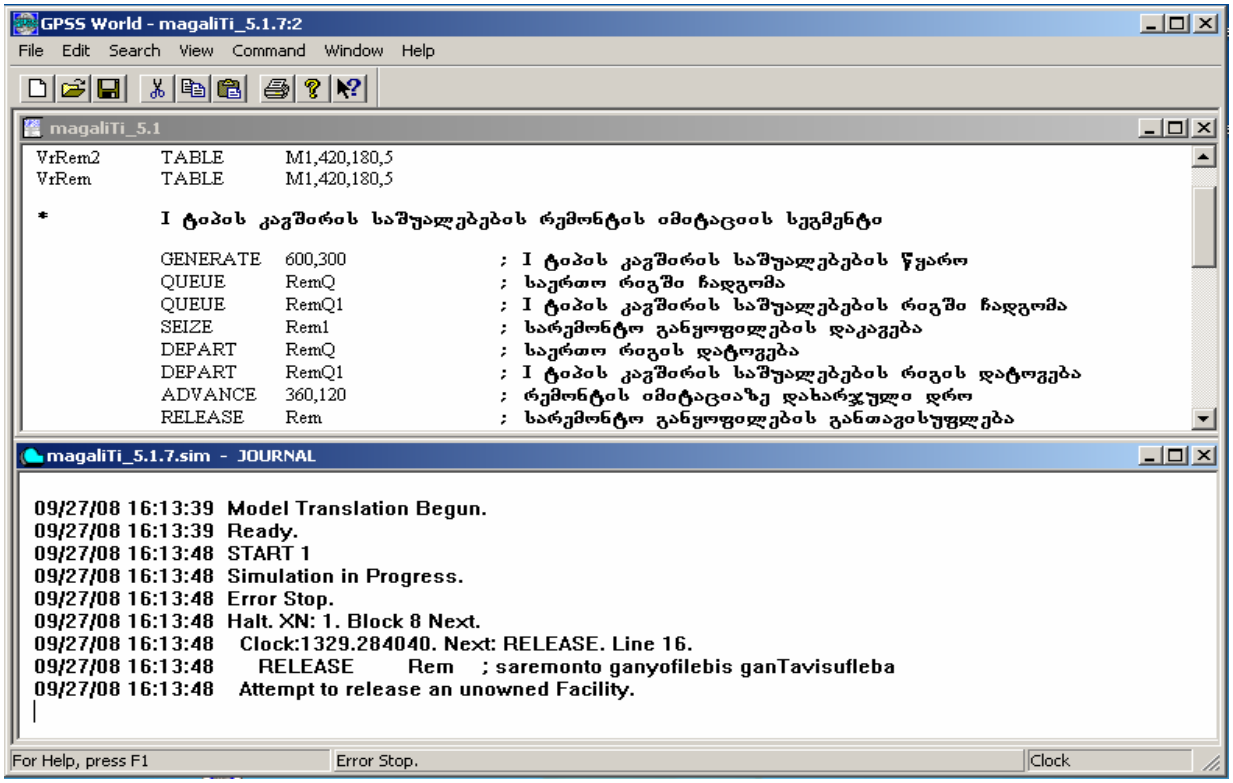


ნახ. 10.14. **JOURNAL** ფანჯარა შეცდომების არარსებობის შესახებ შეტყობინებით

შეცდომების გასწორების შემდეგ განმეორებით ტარდება ტრანსლაცია **Command ►Retranslate (Ctrl + Alt + R)** ბრძანების საშუალებით**.** ტრანსლატორის მიერ აღმოჩენილი შეცდომების არარსებობისას **JOURNAL** ფანჯარაში გამოიტანება შესაბამისი შეტყობინება (ნახ. 10.14).

ამის შემდეგ გაუშვით მოდელირების პროცესი. ამისათვის აირჩიეთ **Command ► Start** ბრძანება. გამოჩნდება დიალოგური ფანჯარა (ნახ. 10.15), რომელშიც შესაძლებელია 1-ის შეცვლა ნებისმიერი რიცხვით (მიუთითეთ მოდელის გაშვების რიცხვი).

ნახ. 10.15. **Start** ბრძანების დიალოგური ფანჯარა



ნახ. 10.16. **JOURNAL** ფანჯარა შესრულების შეცდომის შესახებ შეტყობინებით

JOURNAL ფანჯარაში გამოიტანება შემდეგი სახის შეტყობინება **error Stop (**ნახ. 10.16**)**. შეტყობინება მიუთითებს, რომ მოცემულ მოდელში ტრანზაქტი ცდილობს ერთარხიანი მოწყობილობის განთავისუფლებას, რომელიც არ იყო დაკავებული ანუ **RELEASE** ბლოკში **Rem** სახელის ნაცვლად უნდა იყოს **Rem1**.

ობიექტი „მოდელირების პროცესი“ შექმნილია. ვიდრე შეუდგებით მოდელის გამართვის პროცესს, ტესტირებასა და მის ექსპლუატაციას, განვიხილოთ **GPSS World**-ის საშუალებები, რომლებიც გამოიყენება ამ დროს.

10.3. GPSS World-ის ბრძანებები

10.3.1. ინტერაქტიული ოპერატორები

მას შემდეგ, რაც წარმატებით დასრულდა მოდელის პირველი ტრანსლაცია, შეიქმნა:

- ბლოკების თანამიმდევრობა;
- რეგისტრირებული პროცედურის სია;
- ბრძანებების სია.

ყოველივე ეს გადაეცემა ობიექტს „მოდელირების პროცესი“. ბლოკების თანამიმდევრობა და ბრძანებების სია ცალკე ინახება.

მოდელის პირველდაწყებითი ტრანსლაციის შემდეგ შექმნილ ობიექტს „მოდელირების პროცესი“ შესაძლოა გადავცეთ მოდელის ნებისმიერი ოპერატორები. მათ ინტერაქტიულ ოპერატორებს უწოდებენ [18, 19]. ესენი გახლავთ:

- ბრძანებები;
- ბლოკები.

პირველდაწყებითი ტრანსლაციის შექმნის დროს ბლოკების თანამიმდევრობა მომდევნო განმეორებით ტრანსლაციამდე არ იცვლება. ინტერაქტიული ბლოკები შედის აღნიშნულ თანამიმდევრობაში. მათი ტრანსლაცია მიმდინარეობს დამოუკიდებლად და გადაიცემა ობიექტის „მოდელირების პროცესი“ შესასრულებლად. ამისათვის მათი შესვლისას იქმნება დროებითი ბლოკი, რომელიც გამოიყენება ხელით მოდელირების რეჟიმში და შემდეგ ქრება.

ინტერაქტიულ ბლოკებს ეწოდებათ მოდელირების „ხელით შეტანის“ რეჟიმის ოპერატორები. პირველდაწყებითი ტრანსლაციის პროცესში ასევე იქმნება ბრძანებების სია, რომელიც წარმოშობს რიგს. აღნიშნული ბრძანებები იყოფა სწრაფ და ნელ ბრძანებებად. სწრაფად შესრულებად ბრძანებებს მიეკუთვნება მხოლოდ **HALT** და **SHOW** ბრძანებები, ხოლო ყველა დანარჩენი ბრძანება კი – ნელს. სწრაფად შესრულებადი ბრანებები სრულდება მაშინვე, მას შემდეგ რაც გადაეცა ობიექტს „მოდელირების პროცესი“, ხოლო ყველა დანარჩენი ბრძანებები დგება რიგში იმ ბრძანებების შემდეგ, რომლებიც ჯერ კიდევ არ შესრულდა.

GPSS World-ის ბრძანებები თავისი დანიშნულების მიხედვით იყოფა ორ ჯგუფად:

- **GPSS** ობიექტის განსაზღვრის ბრძანებები;
- მოდელირების პროცესის მართვის ბრძანებები.

10.3.2. GPSS-ის ობიექტების განსაზღვრის ბრძანებები

- **BVARIABLE –** განსაზღვრავს ბულის ცვლადს;
- **EQU –** ენიჭება მომხმარებლის ცვლადის მნიშვნელობა;
- **FUNCTION –** განსაზღვრავს ფუნქციას;
- **FVARIABLE –** განსაზღვრავს ცვლადს;
- **INITIAL –** ახდენს ინიციალიზაციას, ანუ ცვლის ლოგიკური გასაღებისა და მატრიცის უჯრედის მნიშვნელობას;
- **MATRIX –** განსაზღვრავს მატრიცის უჯრედის შენარჩუნებულ სიდიდეს;
- **QTABLE –** განსაზღვრავს რიგის შესახებ სტატისტიკურ მონაცემთა ცხრილს (**Q-**ცხრილები);
- **RMULT –** აყენებს პირველი შვიდი შემთხვევით რიცხვების გენერატორის პარამეტრებს;
- **START –** აყენებს დასასრულის მრიცხველის მნიშვნელობას და უშვებს მოდელს;
- **STORAGE –** განსაზღვრავს მეხსიერების ტევადობას (მრავალარხიან მოწყობილობას**) ;**
- **VARIABLE –** განსაზღვრავს ნამდვილ ცვლადს;
- **TABLE –** განსაზღვრავს ცხრილს.

10.3.3. მოდელირების პროცესის მართვის ბრძანებები

- **CLEAR –** ახდენს დაგროვილი სტატისტიკური მონაცემების ანულირებას და მოდელიდან შლის ყველა ტრანზაქტს;

- **CONDUCT** – უშვებს ექსპერიმენტს;
- **CONTINUE** – აგრძელებს მოდელირების პროცესს;
- **EXIT** – ასრულებს მუშაობას **GPSS World**-თან;
- **HALT** – აჩერებს მოდელირების პროცესს და ახდენს რიგიდან ყველა ბრძანების წაშლას;
- **INCLUDE** –კითხულობს და ახდენს მოდელის დამატებითი ფაილის ტრანსლაციას;
- **INTEGRATE** –ავტომატურად ინტეგრირებს მომხმარებლის ცვლადის დროის მიხედვით;
- **REPOR** – მიუთითებებს ანგარიშის ფაილის სახელს ან აწარმოებს დაუყოვნებლივი ანგარიშის დაკითხვას;
- **RESET** – ახდენს შემთხვევითი რიცხვების გენერატორის სტატისტიკური მონაცემების ნულში დაყენებას და აწარმოებს სამოდელო დროის წამმზომის ჩამოყრის გარეშე წარმართვას;
- **SHOW** – აწარმოებს გამოსახულების გამოთვლას და ჩვენებას;
- **STEP** – მიუთითებს ბლოკში ტრანზაქტების შესვლის შეზღუდულ რაოდენობას;
- **STOP** – მიუთითებს შეჩერების პირობას, რომელიც ეფუძნება ბლოკში ტრანზაქტების შესვლის რაოდენობას;

ქვემოთ განხილულია მოდელირების პროცესის მართვის ბრძანებები.

10.3.3.1. CLEAR ბრძანება

ექსპერიმენტის ჩასატარებლად აუცილებელია რამდენჯერმე თანამიმდევრობით იქნას გაშვებული მოდელი, რომელშიც შეტანილი იქნება მცირედი ცვლილებები. **GPSS World**-ში ამისათვის გამოიყენება **CLEAR** ბრძანება. მას გააჩნია შემდეგი ფორმატი:

CLEAR [A]

A ოპერანდი შესაძლებელია იყოს **ON** (ჩართული) ან **OFF** (გამორთული) მდგომარეობაში. ფარულად იგი იმყოფება ჩართულ მდგომარეობაში.

აღნიშნული ბრძანების საშუალებით მოდელირების პროცესი უბრუნდება საწყის მდგომარეობას. იგი უზრუნველყოფს ყველა დაგროვილი სტატისტიკური ინფორმაციის ჩამოყრას, მოდელირების პროცესიდან იშლება ყველა არასასურველი ტრანზაქტი და **GENERATE** ბლოკში შედის პირველი ტრანზაქტი. ამ დროს ერთარხიანი და მრავალარხიანი მოწყობილობები ხელმისაწვდომი ხდება, ყველა ბლოკის შემცველობა უტოლდება ნულს, შემთხვევითი რიცხვების გენერატორი არ ნულდება.

შენიშვნა: თუ **CLEAR** ბრძანებაში **A** ოპერანდი იმყოფება **OFF** (გამორთული) მდგომარეობაში, მაშინ უჯრედები, მატრიცის ელემენტები და ლოგიკური გასაღები შეუცვლელი რჩება.

10.3.3.2. CONDUCT ბრძანება

CONDUCT ბრძანება განკუთვნილია ექსპერიმენტის გასაშვებად. მას გააჩნია შემდეგი ფორმატი:

CONDUCT [A]

A ოპერანდი წარმოადგენს ექსპერიმენტის **PLUS**-პროცედურების გამოძახებას. ექსპერიმენტი შესაძლებელია წინასწარ იქნას დარეგისტრირებული ობიექტში „მოდელირების პროცესი“. თუ ზემოხსენებულ ობიექტში არგუმენტის გარეშე იმყოფება მხოლოდ ერთი ექსპერიმენტი, მაშინ ოპერანდი არ გამოიყენება. მაგალითად:

CONDUCT MyExperiment (Number of Tellrs, Starting Replicate Number)

Plus ექსპერიმენტი ისევე გაიშვება როგორც ნებისმიერი სხვა პროცედურა. მომხმარებლის ცვლადები **Number of Tellrs** და **Starting Replicate Number** გამოიყენება, რათა მიუთითოს თუ ექსპერიმენტის დროს საიდან დაიწყოს ან სად განაგრძოს მოდელის გაშვება. არგუმენტები გამოითვლება და გადაეცემა განხორციელებულ ექსპერიმენტს. ექსპერიმენტის დროს მისაწვდომია მხოლოდ **HALT** ბრძანება, ანუ მოდელირების პროცესში მომხმარებლების ურთიერთქმედება შეზღუდულია. ამასთან, მომხმარებელს გააჩნია შესაძლებლობა დააყენოს მხოლოდ აბსოლუტური სამოდელო

დრო **ViewSimulation Clock** ბრძანების საშუალებით, ხოლო ურთიერთქმედებისათვის საჭიროა ექსპერიმენტის შეჩერება.

10.3.3.3. CONTINUE ბრძანება

CONTINUE ბრძანება განკუთვნილია შეჩერებული მოდელირების პროცესის აღდგენისათვის. მას გააჩნია შემდეგი ფორმატი:

CONTINUE

მოდელირების პროცესი შეჩერდება დასრულების პირობის შესრულებისას, **HALT** ბრძანების შეტანისას შეცდომის წარმოშობის ან ნორმალური დასრულების შემთხვევაში. პირობები დგინდება **STOP** ან **STEP** ბრძანების საშუალებით.

თუ მოდელირების პროცესი შეჩერდება დასრულების პირობის მიხედვით **CONTINUE** ბრძანება გაუშვებს აღნიშნულ პირობას, მაგრამ არ წაშლის მას. თუ იგივე პირობა კვლავ წარმოიშვება მოდელირების პროცესი კვლავ შეჩერდება. დასრულების პირობა შესაძლოა გამოირიცხოს **STOP OFF** ბრძანების მოდელიდან ან ნათლად წაიშალოს **Blocks** ფანჯარაში.

CONTINUE ბრძანების გამოყენება შეიძლება მაშინ, როდესაც მოდელირების პროცესი წყდება **HALT** ბრძანებით. ამგვარად **HALT** ბრძანება შლის ყველა ბრძანებას ბრძანებათა რიგიდან და გრძლდება მოდელირების პროცესი.

CONTINUE ბრძანების დამუშავებისას პირველ რიგში მოიძიება დასრულების მრიცხველი. თუ მისი მნიშვნელობა არ უდრის ნულს, მაშასადამე **START** ბრძანების შესრულება არ იყო დასრულებული, მაშინ ტრანზაქტების დამუშავების გაგრძელების მიზნით თავიდან გამოიყენება **CONTINUE** ბრძანება. თუ დასრულების მრიცხველის მნიშვნელობა უდრის ნულს, მაშინ ფორმირდება სტანდარტული ანგარიშის სახით.

10.3.3.4 EXIT ბრძანება

GPSS World-ში **EXIT** ბრძანება განკუთვნილია მუშაობის სეანსის სწრაფად დასასრულებლად. მას გააჩნია შემდეგი ფორმატი:

EXIT [A]

A ოპერანდი წარმოადგენს **GPSS World**-დან გამოსვლის კოდს. **A** ოპერანდი გამოიყენება ობიექტების „მოდელი“ და „მოდელირების პროცესი“ ფაილში ჩაწერის წარმართვისათვის. დასაშვები მნიშვნელობებია:

- 1 – ყველა ობიექტი შენარჩუნდება;
- -1 – ყველა ობიექტი არ შენარჩუნდება;
- 0 ან ფარულად – თითოეული შეცვლილი ფაილი იწვევს ფანჯრის გამოჩენას შეკითხვით მისი შენახვის შესახებ.

დანართის ფანჯრის დახურვას აგრეთვე ემსახურება ბიბლიოთეკური პროცედურა **EXIT ().** მაგალითად **EXIT (-1)** ყველა ობიექტი არ შეინახოთ.

GPSS World-ში გამოიყენება მუშაობის ფონური პაკეტი, როდესაც მომხმარებელთან ურთიერთქმედების გარეშე საჭიროა მოდელირების გარკვეული პროცესის წარმართვა. **EXIT** ბრძანება გამოიყენება პაკეტურ რეჟიმში მუშაობის სეანსის დასრულებისათვის ფანჯრების გარეშე. ამისათვის მოდელში შედის **EXIT 1** ბრძანება ან **EXIT (1)** პროცედურის გამოძახება, რათა დახუროს მუშაობის სეანსი და შეინახოს ყველა ობიექტი.

10.3.3.5. HALT ბრძანება

HALT ბრძანება განკუთვნილია მოდელირების პროცესის შესაწყვეტად და რიგიდან ყველა ბრძანების წასაშლელად. მას გააჩნია შემდეგი ფორმატი:

HALT

HALT ბრძანება წარმოადგენს სწრაფ ბრძანებას, ამიტომ გამომდინარე აქედან ის არ თავსდება ბრძანების რიგში და დაუყოვნებლივ სრულდება. მისი მოქმედების შედეგად მოდელირების პროცესი წყდება და ბრძანებების რიგიდან იშლება ყველა დარჩენილი ბრძანებები. მოდელირების პროცესის განახლებისათვის გამოიყენება **CONTINUE** ბრძანება.

10.3.3.6. INCLUDE ბრძანება

GPSS World-ში **INCLUDE** ბრძანება განკუთვნილია მოდელის დამატებითი ფაილის ჩაწერისა და ტრანსლაციისათვის ოპერატორებით ან ბრძანების სიით. მას გააჩნია შემდეგი ფორმატი

INCLUDE [A]

A ოპერანდი წარმოადგენს ფაილის სპეციფიკაციას. (ფაილთან შეღწევის გზას). დასაშვები მნიშვნელობებია – **String,** მაგალითად:

INCLUDE “DanDon.txt“
NCLUDE “D:\Magaliti\DanDon.txt“

პირველ მაგალითში ფაილთან შეღწევა არ წარმოებს (არ არსებობს შეღწევის გზა), ასე რომ იგულისხმება, რომ ფაილი მითითებული სახელით იმყოფება მოდელის საქაღალდეში. მეორე შემთხვევაში ნაჩვენებია ფაილთან შეღწევის გზა. **INCLUDE** ბრძანება წარმოადგენს სწრაფ ბრძანებას. ტრანსლაციისას ის შეიცვლება ფაილით. ამიტომ **INCLUDE** ბრძანება მოდელში უნდა მოთავსდეს იმ ადგილას, სადაც უნდა იყოს ფაილში შემავალი ოპერატორები და ბრძანებები.

ყველა დამატებით შეტანილი ფაილი ტრანსლატორის მიერ ინომრება მთელი რიცხვებით, რომლებიც იწყება ნულით. ნული ნომერი მიენიჭება ობიექტ-მოდელს. ერთი ფაილის რამდენჯერმე

შეტანის შემთხვევაში მიენიჭებათ უნიკალური ნომერი, ანუ ფაილის თითოეული შეტანა იწვევს განსხვავებული ბლოკების შექმნას. **INCLUDE** ბრძანება უშვებს მოდელის ფაილის 5 დონეზე ჩაშენებას.

10.3.3.7. REPORT ბრძანება

REPORT ბრძანება იწვევს სტანდარტული ანგარიშის დამოუკიდებლად შექმნას. მას გააჩნია შემდეგი ფორმატი:

REPORT [A], [B]

შეთავსებისათვის დატოვება. მაგალითად:

REPORT
REPORT ,Now

ორივე აღნიშნული ჩანაწერი იდენტური გახლავთ და მათ მივყავართ სტანდარტული ანგარიშის დაუყოვნებლივ შექმნასთან. როდესაც ობიექტი მოდელირების პროცესი ხვდება **REPORT** ბრძანებას, დაუყოვნებლივ იწყებს **IN Windows** ფანჯარაში პარამეტრების შესაბამისი სტანდარტული ანგარიშის შექმნას, რომელიც იმყოფება ჟურნალში **REPORT** (ანგარიშის) გვერდზე.

თუ პარამეტრი **In Windows** დაყენებულია, მაშინ იქმნება ახალი ობიექტი-ანგარიში, რომელიც შეიცავს ახალ სტანდარტულ

ანგარიშს. შემდეგ ის შეიძლება იქნეს ან არ იქნეს შენახული ფაილში.

თუ პარამეტრი **In Windows** არ არის დაყენებული, მაშინ შეიქმნება სტანდარტული ანგარიში, რომელსაც მიენიჭება მომდევნო რიგითი ნომერი ან მოთავსდება ფაილში.

10.3.3.8. RESET ბრძანება

მოდელის საწყისი პირობები შეიძლება შესამჩნევად განსხვავდებოდეს იმ პირობებისაგან, რომელთაც ადგილი აქვთ სისტემის მდგრადი მდგომარეობის მიღწევისას. თუ სამოდელო სისტემა მუშაობს ციკლურ რეჟიმში ან წარმოადგენს შუალედურ რგოლს, ყველაზე მნიშვნელოვანი შეიძლება აღმოჩნდეს სისტემის ფუნქციონირების შესწავლა დატვირთვის გაზრდით ან შემცირებით. ასეთი ტიპის შეიძლება იყოს სამხედრო დანიშნულების სისტემათა უმეტესობა.

სისტემის მაგალითს, რომლის მუშაობა შეიძლება შეფასდეს დამყარებულ რეჟიმში, წარმოადგენს სისტემები რომლებიც მეტად ან/და ნაკლებად თანაბრად ფუნქციონირებს მთელი სამოდელო დროის განმავლობაში. შესაბამისი სტატისტიკური ინფორმაციის მისაღებად არსებობს სამი მიდგომა:

1. მოდელი ისე მუშავდება, რომ სისტემის ფუნქციონირების პირობები მის დაწყებიდანვე ტიპური გახლავთ;
2. სისტემა ისე ნელა მოდელირდება, რომ ნებისმიერი არატიპური სტატისტიკური მონაცემის ნაკრები

მოდელირების პროცესის დასაწყისშივე „ჩაიძირება“ ტიპურ მონაცემებს შორის, რომელთა შეკრებაც მოხდება მომდევნო პერიოდში;

3. შესაძლებელია შემდეგნაირი ქცევა: დაამოდელოთ სისტემის მუშაობა ტიპური პირობების წარმოქმნამდე; მოდელის მდგომარეობის შეუცვლელად მოახდინეთ ამ პერიოდამდე (დრომდე) შეკრებილი სტატისტიკური მონაცემების უგულებელყოფა; განაგრძეთ მოდელირება, შეაგროვეთ სტატისტიკური მონაცემები, რომელზეც უკვე აღარ ახდენს (ვერ მოახდენს) გავლენას არატიპური სიტუაციები.

პირველი მიდგომა მოდელის შემქმნელისაგან მოითხოვს მუშაობის ტიპური პირობების ცოდნას და მათ მოდელში შეტანას. რთულ სისტემათა მოდელებში მათი შესრულება სათუო გახლავთ.

მეორე მიდგომისას საჭიროა მოდელირების პროცესის გაგრძელება იმ მომენტის დადგომამდე, როდესაც გაქრება შეგროვილ, არასწორ მონაცემთა გავლენა. რთული სისტემებისათვის ამგვარი მოდელირების ფასი შეიძლება ძალიან დიდი აღმოჩნდეს, რაც აღნიშნულ მიდგომას არასასურველს ხდის.

მესამე მიდგომა აღმოჩნდება გაცილებით უფრო ხელსაყრელი. მოდელირების განსაზღვრულ სტადიაზე მოდელირების შემდეგი გაგრძელებისათვის უგულებელყავით სტატისტიკური მონაცემები მოდელის მოდიფიკაციის გარეშე. სტატისტიკის ნულში დაყენება მოდელირების პროცესიდან ტრანზაქტების წაუშლელად ხორციელდება **RESET** ბრძანების საფუძველზე. მას გააჩნია შემდეგი ფორმატი:

RESET

CLEAR ბრძანებისგან განსხვავებით **RESET** ბრძანება არ შლის ტრანზაქტს მოდელირების პროცესიდან. **RESET** ბრძანება ანულებს შემთხვევითი რიცხვებს, გენერატორს არ აყენებს ნულში, არ ახდენს გავლენას აბსოლუტურ სამოდელო დროსა და ტრანზაქტების ნუმერაციაზე. ფარდობითი სამოდელო დრო (**RESET** ბრძანების შემდეგ ბოლო დრო) დგება ნულში. ამგვარად, **RESET** ოპერატორი სისტემის ფუნქციონირების ტიპურ პირობებში წარმოადგენს სტატისტიკურ მონაცემთა შეკრების მოსახერხებელ საშუალებას. იგულისხმება, რომ თუ მოდელირების პროცესი გრძელდება დიდხანს, მაშინ სისტემა შედის სტაციონარულ მდგომარეობაში.

10.3.3.9. SHOW ბრძანება

SHOW ბრძანება განკუთვნილია ობიექტი „მოდელირების პროცესი“-ს გადასაცემად გამოსახულება „გამოთვლის“ და შედეგების ჩაწერის **MODEL** ფანჯრის მდგომარეობის სტრიქონში. მას გააჩნია შემდეგი ფორმატი:

SHOW X

X არის არითმეტიკული ან ლოგიკური გამოსახულება.

SHOW ბრძანება წარმოადგენს სწრაფს და შესაბამისად სწრაფად სრულდება. თუ გამოსახულებაში იმყოფება სტრა-ები, მათი გამოთვლა წარმოებს აქტიურ ტრანზაქტზე დამოკიდებულებით. აქტიური ტრანზაქტის არარსებობისას, ანუ როდესაც მოდელირების პროცესი ჯერ კიდევ არ არის გაშვებული, გამოვა შეტყობინება შეცდომის შესახებ. მომხმარებლის ცვლადები, თუ ისინი გამოიყენება გამოსახულებაში, მათ წინასწარ უნდა მიეცეთ მნიშვნელობები **EQU** ბრანების საშუალებით. მაგალითად:

SHOW XN1
SHOW N$Wde1/N$Wde2
SHOW LOG (AC1)

პირველ მაგალითში მდგომარეობის სტრიქონი აჩვენებს აქტიური ტრანზაქტის ნომერს. მეორე მაგალითში – **Wde1** ჭდით მითითებულ ბლოკში შესული ტრანზაქტების რიცხვის **Wde2** ჭდით მითითებულ ბლოკში შესული ტრანზაქტების რიცხვზე გაყოფის შედეგს. მესამე მაგალითში გამოითვლება აბსოლუტური სამოდელო დროის ნატურალური ლოგარითმი და შედეგი გამოვა მდგომარეობის სტრიქონში.

10.3.3.10. STEP ბრძანება

STEP ბრძანება განკუთვნილია მოდელირების პროცესის დროს ბლოკებში ტრანზაქტების მოცემული რაოდენობის შესვლათა

შესრულების ორგანიზებისათვის. ბრძანების ფორმატს გააჩნია შემდეგი სახე:

STEP [A]

A ოპერანდი არის ბლოკებში ტრანზაქტების შესვლათა რაოდენობა. ბრძანება:

STEP 1

გამოიყენება მოდელირების პროცესის ნაბიჯ-ნაბიჯ შესასრულებლად, ანუ მოდელირების პროცესი ჩერდება (წყდება) ბლოკში ტრანზაქტის ერთი შესვლის შედეგ. ამასთან ერთად, **JOURNAL** ფანჯარაში გამოჩნდება დროის, აქტიური ტრანზაქტის ნომრის, აქტიური ტრანზაქტის მქონე მიმდინარე ბლოკისა და აქტიური ტრანზაქტის შესასვლელად განკუთვნილი მომდევნო ბლოკის შემცველი შეტყობინება.

STEP ბრძანება მოხერხებულია მოდელირების პროცესის ტესტირების (გამართვის) დროს. მოდელირების პროცესის ნაბიჯ-ნაბიჯ შესრულება **JOURNAL** ფანჯრის გარდა დინამიურად აისახება **BLOCKS** (ბლოკები) ფანჯარაში, რაც საშუალებას იძლევა თვალყური მიედევნოს ტრანზაქტების მოძრაობის გზებს და რა თქმა უნდა, მოდელის მუშაობის ლოგიკას.

STEP ბრძანება შეიძლება შეწყვეტილ იქნას ღილაკების კომბინაციით [Ctrl]+[Alt]+[I] ან „მაუსზე“ დაწკაპუნებით **BLOCKS** (ბლოკები) ფანჯარაში.

10.3.3.11. STOP ბრძანება

STOP ბრძანება განკუთვნილია მოდელირების პროცესის შეჩერების (შეწყვეტის) პირობის დასადგენად ან მოსახსნელად (გასაუქმებლად). ბრძანების ფორმატს გააჩნია შემდეგი სახე:

STOP [A], [B], [C]

A ოპერანდი არის ტრანზაქტის ნომერი;

B ოპერანდი – ბლოკის ნომერი;

C ოპერანდი – ბრძანების მდგომარეობის ალამი, შესაძლოა იყოს **ON** ან **OFF**, ფარულად იმყოფება **ON** მდგომარეობაში.

მაგალითად:

STOP 34, 26

მოცემულ ბრძანებას შეაქვს პირობა, რომელიც გამოიწვევს მოდელირების პროცესის შეჩერებას (შეწყვეტას), როდესაც ნომერი 34 ტრანზაქტი შეეცდება შევიდეს ნომერ 26 ბლოკში. მდგომარეობის სტრიქონში და **JOURNAL**-ის ყველა გახსნილ ფანჯარაში გადაიცემა დრო, აქტიური ტრანზაქტის ნომერი, აქტიური ტრანზაქტის მქონე მიმდინარე ბლოკი და მის შესასვლელად განკუთვნილი მომდევნო ბლოკი.

მოდელირების პროცესის გაგრძელების მიზნით გამოიყენება ბრძანება **CONTINUE.** ეს ბრძანება არ ხსნის შეჩერების (შეწყვეტის) პირობას. თუკი აღნიშნული პირობა ხელახლა შესრულდება, მოდელირების პროცესი კვლავ შეჩერდება (შეწყდება). შეწყვეტის ყველა პირობათა მოსახსნელად აუცილებელია გამოვიყენოთ ბრძანება:

STOP , , O

მოვიყვანოთ ბრძანების მაგალითები:

STOP 83
STOP 83 , , OFF
STOP ,Wde2
STOP ,Wde2, OFF

პირველი ბრძანება წყვეტს მოდელირების პროცესს, როდესაც ნომერი 83 ტრანზაქტი ხდება აქტიური. მეორე ბრძანება ხსნის პირველი ბრძანებით დადგენილ (დაწესებულ) პირობას. მესამე ბრძანება იწვევს მოდელირების პროცესის შეწყვეტას, როდესაც ნებისმიერი ტრანზაქტი ცდილობს შევიდეს **Wde2** ჭდით მითითებულ ბლოკში. მეოთხე ბრძანება აუქმებს აღნიშნულ პირობას.

ტესტირების ეტაპზე **STOP** ბრძანების საშუალებით შეიძლება თვალყური ვადევნოთ მოდელირების პროცესის შესრულების მართებულობას.

10.4. GPSS World–ის ფანჯრები

მოდელირების პროცესის წარმართვაზე, მის ტესტირებასა და გამართვის ეტაპზე ბრძანებების მოქმედების თვალყურის სადევნებლად გამოიყენება ათი გრაფიკული ფანჯარა [16]. ობიექტების ტიპების მიხედვით ფანჯრები იყოფა შემდეგ კატეგორიებად:

- ბლოკები **(Blocks);**
- გამოსახულებები **(Expression);**
- მოწყობილობები **(Facilities);**
- ლოგიკური გასაღებები **(Logicswitches);**
- მატრიცა (**Matrix**);
- გრაფიკი (**Plot**);
- რიგები (**Queues**);
- უჯრედები (**Savevalues**);
- მეხსიერებები (**Storages**);
- ცხრილები (**Table**).

ფანჯარა **Blocks** გვიჩვენებს ბლოკებში ტრანზაქტების შესვლას. იგი „მაუსის“ ან კლავიატურის მეშვეობით საშუალებას იძლევა დადგენილ ან წაშლილ იქნან საკონტროლო წერტილები და ვიზუალურად მიადევნოს თვალყური ტრანზაქტების გადაადგილებას.

ფანჯარა **Expression** განკუთვნილია **Plus**-გამოსახულებათა ნებისმიერი რაოდე-ნობის ცვლილებებზე თვალყურის სადევნელად.

Plot ფანჯრის მეშვეობით ერთდროულად შესაძლებელია მრავალფერიანი გრაფიკების ნებისმიერი რაოდენობის თვალყურის დევნება. აღნიშნული გრაფიკები შეიძლება შევინახოთ ან/და დავბეჭდოთ. თითოეულ ფანჯარას ინტეგრირებადი ცვლადების ჩათვლით შეუძლია გამოიტანოს ცვლადების ერთიდან რვამდე გრაფიკი. გრაფიკებს შეუძლიათ მასშტაბირება ნებისმიერი მიმართულებით.

ფანჯარა **Table** წარმოადგენს დინამიკურ ჰისტოგრამას, რომელიც სასარგებლოა როგორც მონაცემთა შეგროვების, ასევე გამონატყორცნების ძიებაზე, აგრეთვე წარმომშობი ალბათური განაწილების მსგავსების შესაფასებლად.

ფანჯრები **Blocks, Facilities, Logicswitches, Queues, Savevalues**, **Storages** გააჩნიათ დაწვრილებითი მიმოხილვა და ზოგადი მიმოხილვა. ისინი ყოველთვის იხსნება დაწვრილებითი მიმოხილვით. ზოგად მიმოხილვაზე გადასვლა წარმოებს **View ► Entity Details** (სახეობა ► დაწვრილებითი მიმოხილვა) მეშვეობით.

დინამიკური ფანჯრების გარდა **GPSS World**-ში ხელმისაწვდომია შემდეგი მდგომარეობის კადრები:

- მიმდინარე მოვლენათა სიისათვის (**CEC Snapshot**);
- სამომავლო მოვლენათა სიისათვის (**FEC Snapshot**);
- ცალკეული ტრანზაქტებისათვის (**Transaction Snapshot**);
- რიცხვითი ჯგუფებისათვის (**Numeric Groups Snapshot**);
- ტრანზაქტების ჯგუფებისათვის (**Transaction Groups Snapshot);**
- მომხმარებლის სიებისათვის (**Userchains Snapshot**).

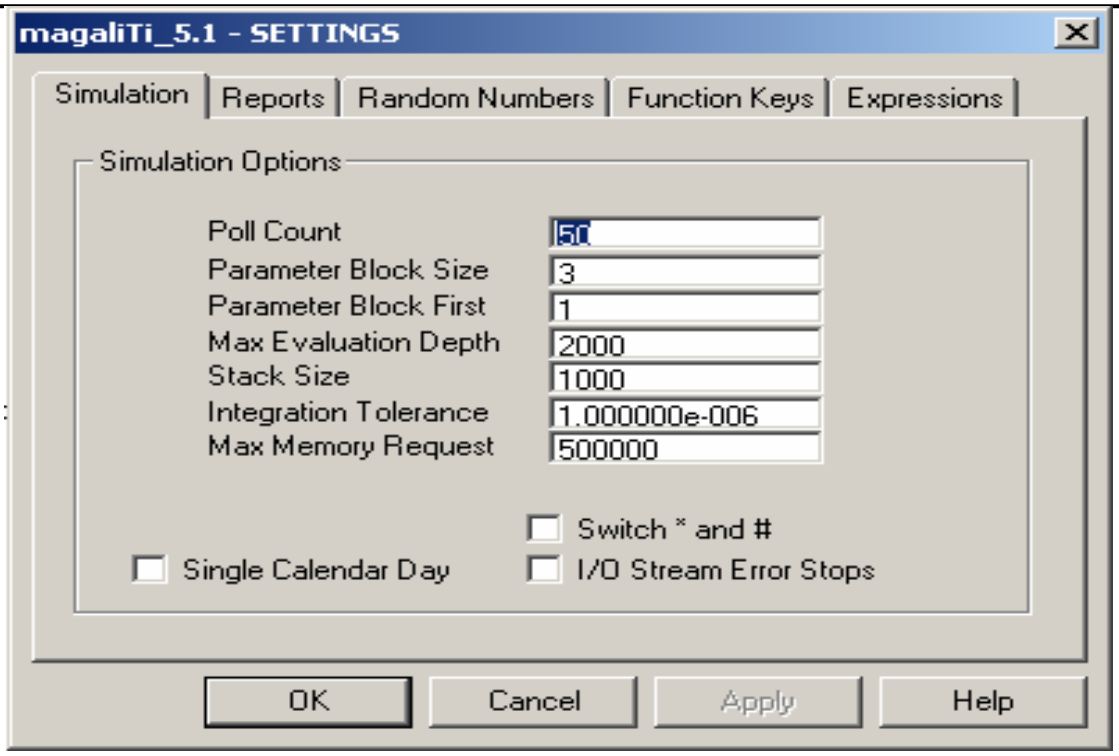
შენიშვნა: დინამიკური ფანჯრები ავტომატურად განახლდება მოდელირების პროცესის ცვლილების შესაბამისად. დინამიკური ფანჯრებისაგან განსხვავებით მდგომარეობის კადრები წარმოადგენს ცვალებადი ობიექტების სტატიკურ ასახვას, რომლებიც დაფიქსირებულია დროის რაღაც მომენტში შესრულებადი მოდელირების პროცესში, ანუ მდგომარეობის კადრები დინამიკურად არ აღდგება.

10.5. მოდელების გამართვა

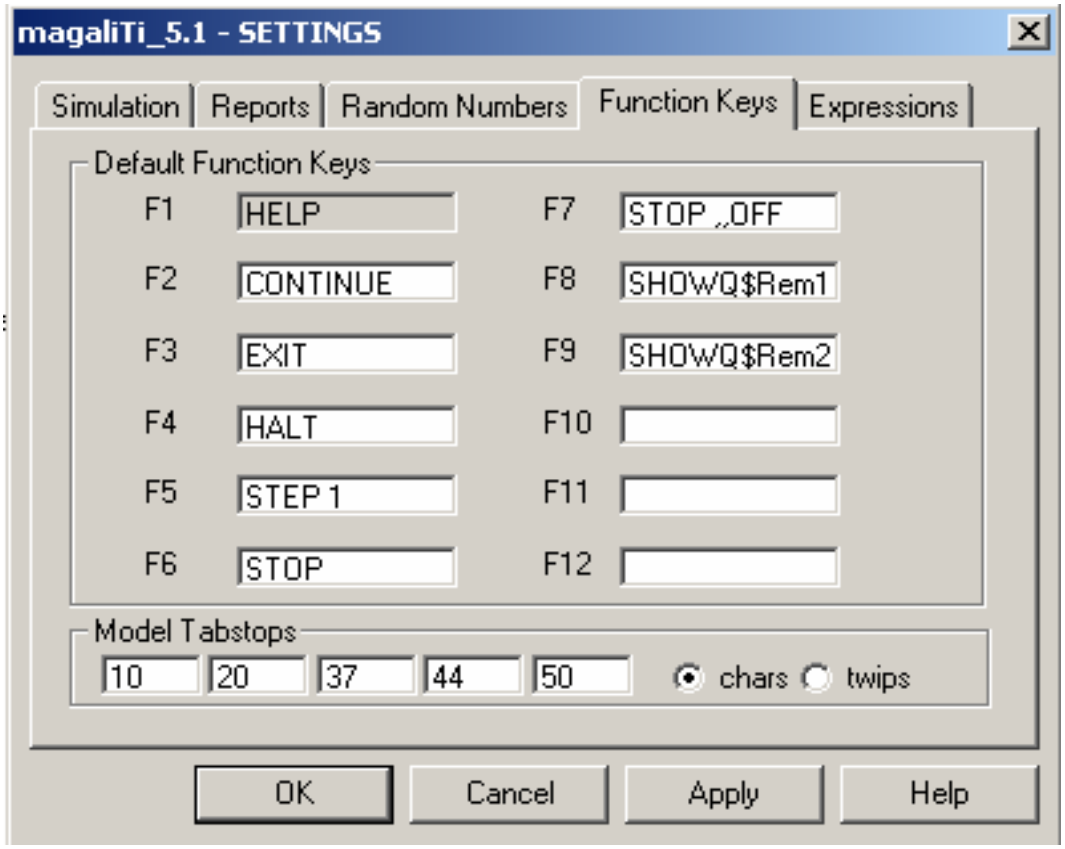
გახსენით ადრე შექმნილი მოდელი, მაგალითი_5.1. ამისათვის აირჩიეთ ბრძანება **File ►Open**, შემდეგ მაგალითი_**5.1.gps** და დააჭირეთ **Open** ღილაკს. აირჩიეთ ბრძანება **Edit ► Settings** (ნახ. 10.17) და გადადით გვერდზე (ჩანართზე) **Function Keys** (ფუნქციონალური ღილაკები) (ნახ. 10.18). [**F8**] და[**F9**] ღილაკების გვერდით აკრიფეთ შემდეგი ორი ბრძანება:

SHOW Q$Rem1
SHOW Q$Rem2

დააწკაპუნეთ **OK** ღილაკზე.



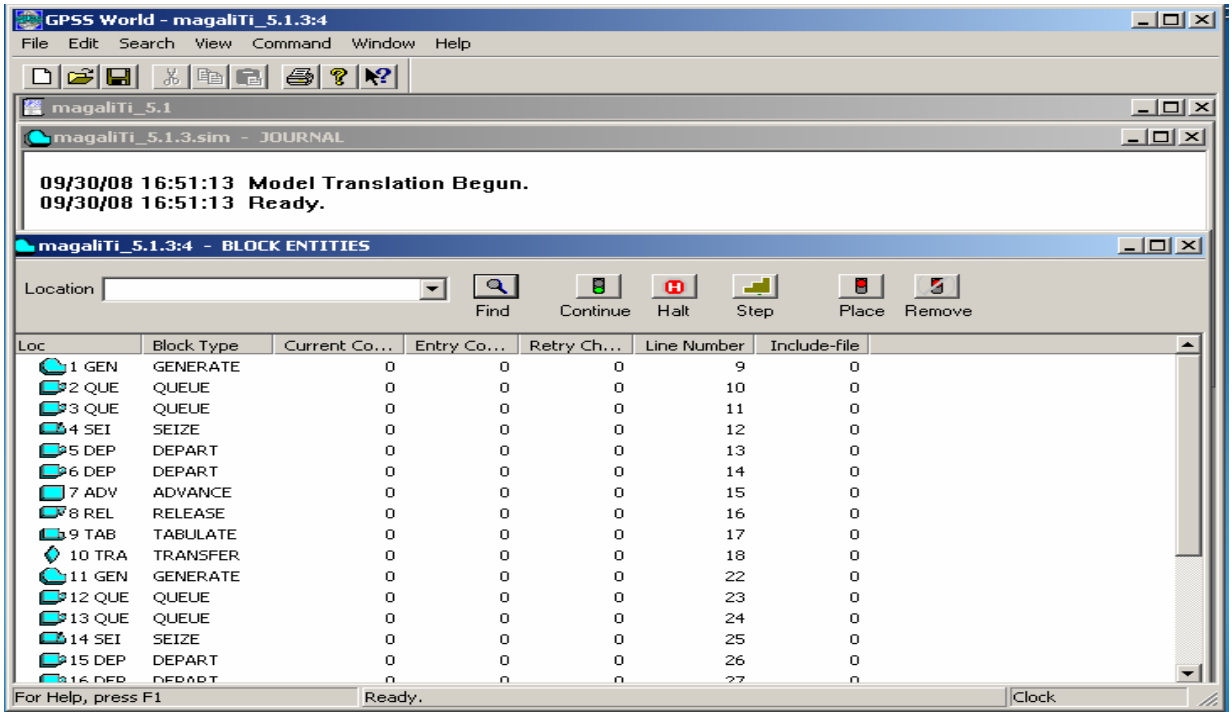
ნახ.10.17. „**Simulation**“ გვერდი მოდელის დაყენების ჟურნალის ფანჯარაში



ნახ.10.18. „**Function Keys**“ გვერდი მოდელის დაყენების ჟურნალის ფანჯარაში

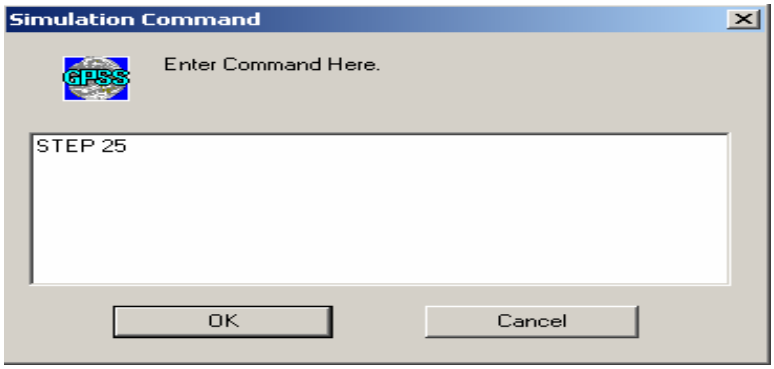
თუ დაუშვით შეცდომა, მაგალითად შეიტანეთ მხოლოდ **Q$Rem1 OK** ღილაკზე დაჭერის შემდეგ გამოჩნდება შემდეგი შეტყობინება: **„Syntax error in operator. Invalid SNA. Expecting a label or GPSS Verb“** – „ოპერატორში დაშვებულია სინტაქსური სახის შეცდომა. მიუთითებელია სტრა. ნავარაუდევია **GPSS**-ს ჭდე ან ზმნა“. თუ შეცდომა არ არის დაშვებული, მაშინ აღნიშნული ბრძანებები მიმაგრებულია ფუნქციონალურ ღილაკებთან. მოდელირების პროცესის შესრულებისას შესაბამის ღილაკებზე დაჭერისას შეიძლება ინტერაქტიულად გამოიკვლიოთ **RemQ1** და **RemQ2** რიგების მიმდინარე სიგრძეები.

გაუშვით მოდელირების პროცესი. აირჩიეთ **Command ►Greate Simulation**. აგრეთვე გახსენით **Blocks** ფანჯარა **Window ►Simulation Window ►Blocks Window** ბრძანების საშუალებით (ნახ. 10.19).



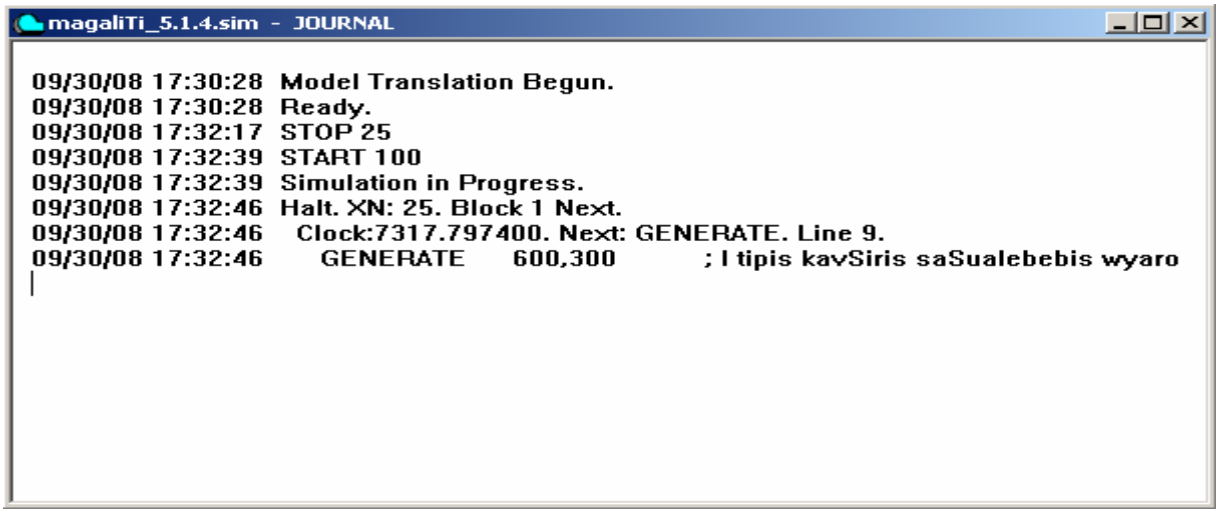
ნახ. 10.19. **Blocks** (ბლოკები) და **Journal** (ჟურნალი) ფანჯრები მაგალით_5.1-ის მოდელში

იმისათვის რომ, ნომერ 25 აქტიურ ტრანზაქტზე მუთითოთ შეჩერების პირობა აირჩიეთ ბრძანება **Command ► Custom …** და დიალოგურ ფანჯარაში აკრიფეთ **STOP 25** შემდეგ დააწკაპუნეთ **Ok** ღილაკზე (ნახ. 10.20).



ნახ. 10.20. ფანჯარა **Command ► Custom …**

მთავარ მენიუში აირჩიეთ ბრძანება **Command ► START** და დიალოგურ ფანჯარაში 1-ის ნაცვლად აკრიფეთ 100, შემდეგ **OK.** გამოვა შეტყობინება იმის შესახებ, რომ მოდელირების პროცესში ნომერი 25 ტრანზაქტი შეჩერდა ბლოკში შესვლის დროს პირველივე მოსინჯვისას (ნახ. 10.21). მოცემულ შემთხვევაში ეს იყო პირველი **GENERATE** ბლოკი.

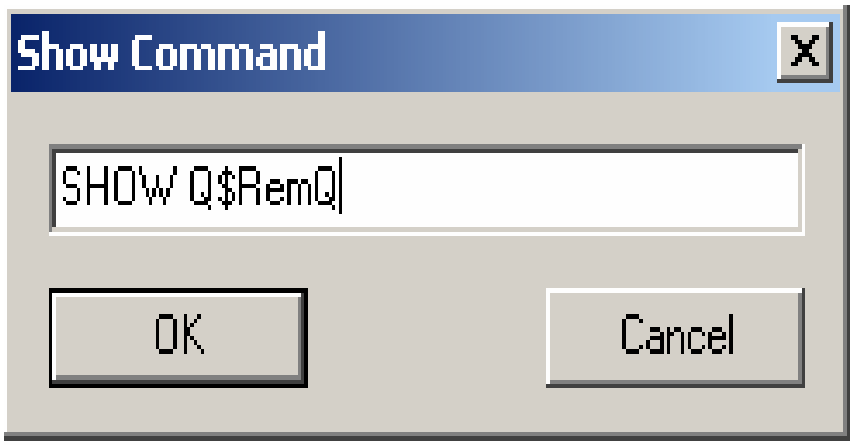


ნახ. 10.21. შეტყობინება **Journal** ფანჯარაში ტრანზაქტის შეჩერების შესახებ

ახლა დაიწყეთ მოდელირების პროცესის შესრულება ბიჯურ რეჟიმში, ამისათვის გამოიყენეთ [**F5**] ფუნქციონალური ღილაკი, რომელზეც მიბმულია მოდელის დაყენების ჟურნალში ბრძანება **STEP 1** (იხ. ნახ. 10.20). მოხსენით შეჩერების პირობა. ამისათვის აირჩიეთ ბრძანება **Window ► Simulation Snapshot ►User Stops** (ფანჯარა ►მოდელირების პროცესის კადრი ►შეჩერების პირობა) მონიშნეთ 25 და დააჭირეთ **Remove** ღილაკს, შემდეგ **OK**.

[**F5**] ფუნქციონალური ღილაკის საშუალებით გამოიძახეთ ბრძანება **STEP** 1, იგივეს გაკეთება შესაძლებელია [**Ctrl**] + [**Alt**] + [**1**] ღილაკების კომბინაციით ან აირჩიეთ მთავარი მენიუს ბრძანება **Command ►STEP 1.** 25-ე ნომერი ტრანზაქტი შევა **GENERATE** ბლოკში. ამის დანახვა შესაძლებელია **Journal** ფანჯრის ტრასულ შეტყობინებაში.

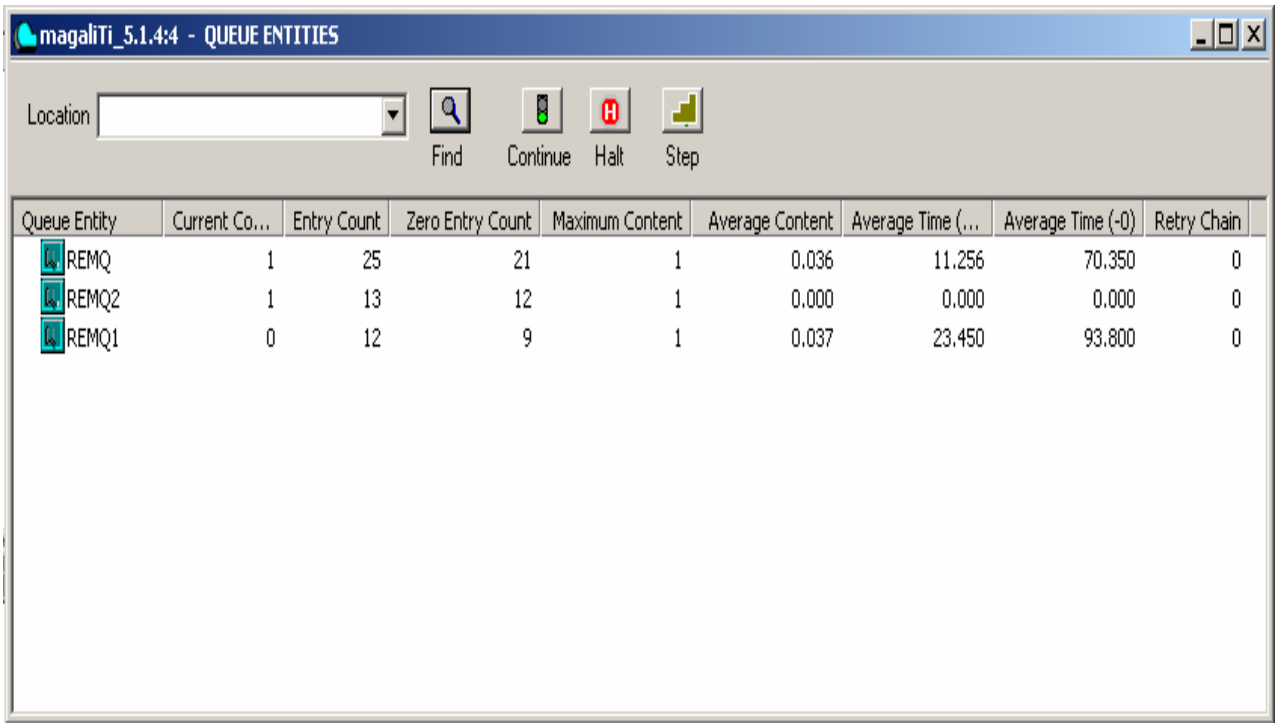
ახლა გამოვიკვლიოთ მოდელირების პროცესის ზოგიერთი მაჩვენებლები. აირჩიეთ ბრძანება **Command ►SHOW** ... და დიალოგურ ფანჯარაში აკრიფეთ **Q$RemQ,** შემდეგ **OK** (ნახ. 10.22).



ნახ. 10.22. **Command ►SHOW** ... დიალოგური ფანჯარა

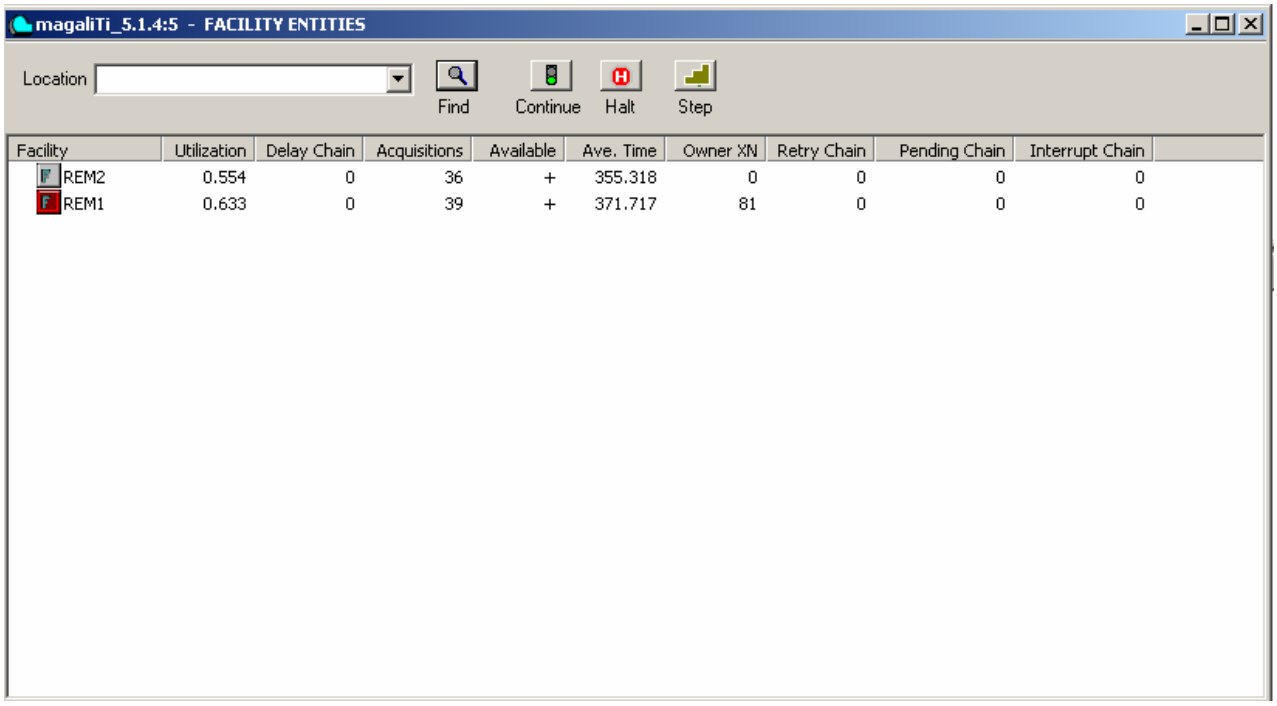
გაუმართავი კავშირის საშუალებების **RemQ** საერთო რიგის სიგრძე აისახება მთავარი ფანჯრის **Journal** ფანჯარაში მდგომარეობის სტრიქონში. შეიტანეთ ბრძანებები, რომლებიც მიბმულია **[F8]** და **[F9]** ფუნქციონალურ ღილაკებთან. **[F8]** ღილაკზე დაჭერისას მთავარი ფანჯრის მდგომარეობის სტრიქონში და **Journal** ფანჯარაში აისახება I ტიპის გაუმართავი კავშირის საშუალებების რიგის სიგრძე, ხოლო **[F9]** – II ტიპის გაუმართავი კავშირის საშუალებების რიგის სიგრძე.

აირჩიეთ **Window ► Simulation Window ►Queues Window** ბრძანება. გაიხსნება ფანჯარა რიგების მდგომარეობის დაწვრილებითი მიმოხილვით (ნახ. 10.23). რიგების პარამეტრების ცვლილების დაკვირვებისათვის დააჭირეთ **[F5]** ღილაკს რამდენჯერმე და დახურეთ ფანჯარა.



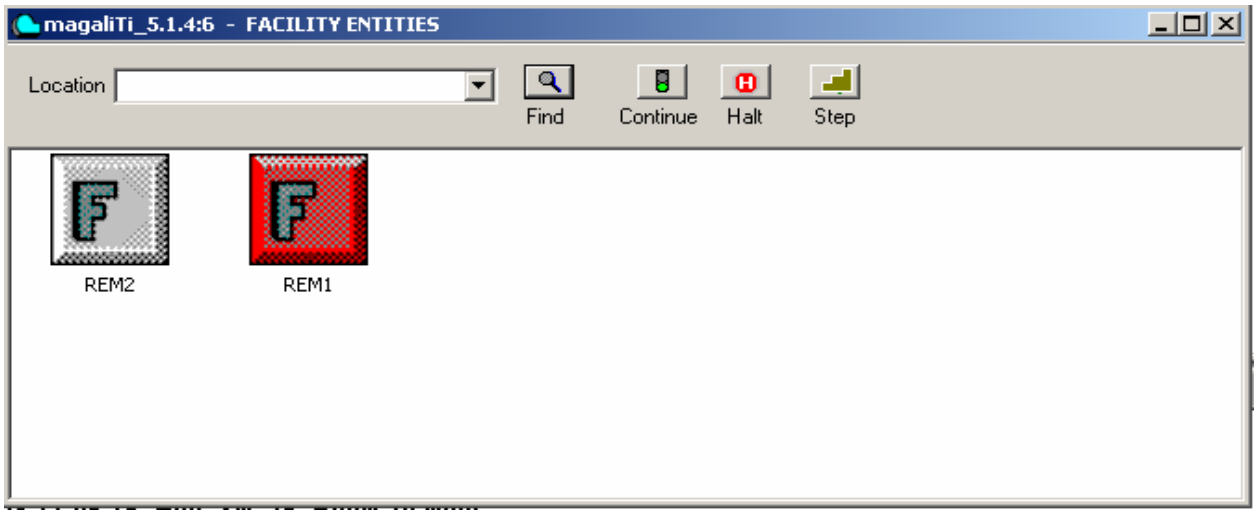
ნახ.10.23. **Queues** (რიგის) ფანჯარა. დაწვრილებითი მიმოხილვა

აირჩიეთ **Window ► Simulation Window ►Facilities Window** ბრძანება (ფანჯარა ► მოდელირების პროცესის ფანჯარა ► „მოწყობილობის“ ფანჯარა). გაიხსნება ფანჯარა მოწყობილობის შესაბამისი მიმოხილვით (ნახ. 10.24). ერთარხიანი მოწყობილობის მდგომარეობის ცვლილების დაკვირვებისათვის დააჭირეთ **[F2]** ღილაკს (იგივეს გაკეთება შესაძლებელია **[Ctrl]** + **[Alt]** + **[C]** ღილაკების კომბინაციით ან აირჩიეთ მთავარი მენიუს ბრძანება **Command ► CONTINUE**), ხოლო შემდეგ **[F4]** ღილაკს (იგივეს გაკეთება შესაძლებელია **[Ctrl]** + **[Alt]** + **[H]** ღილაკების კომბინაციით ან აირჩიეთ მთავარი მენიუს ბრძანება **Command ► HALT**).



ნახ.10.24. **Facilities** (მოწყობილობის) ფანჯარა. დაწვრილებითი მიმოხილვა

Facilities ფანჯარაში საერთო მიმოხილვისათვის აირჩიეთ **View ►Entity Details** (დათვალიერება ►დაწვრილებით) ბრძანება. გახსნება ფანჯარა (ნახ. 10.25).

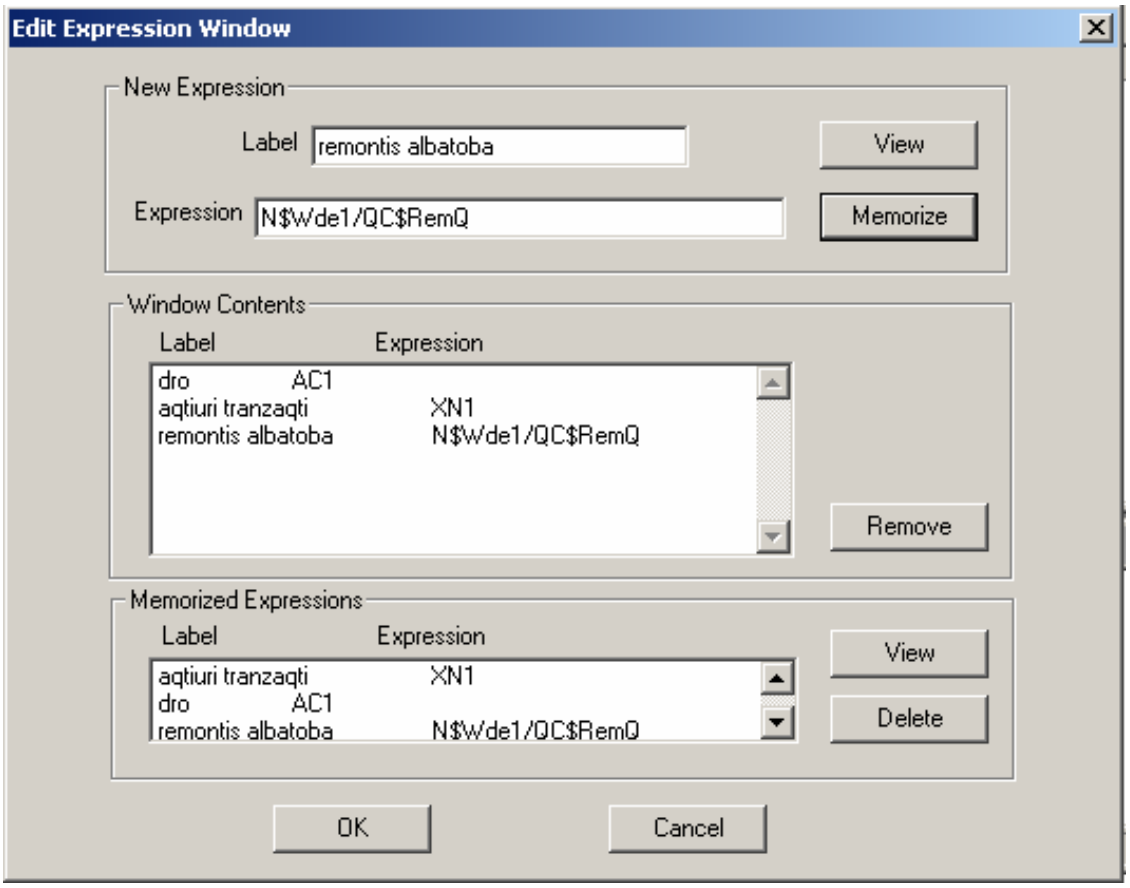


ნახ.10.25. **Facilities** (მოწყობილობის) ფანჯარა. საერთო მიმოხილვა

დააჭირეთ **[F2]** ფუნქციონალურ ღილაკს, შემდეგ კი **[F4]** ფუნქციონალურ ღილაკს. ყურადღება მიაქციეთ **Rem1** და **Rem2** მოწყობილობების მდგომარეობის ცვლილებას. მოწყობილობის მიმთითებელი ნიშნის ფერი იცვლება ნაცრისფერიდან (როდესაც ის თავისუფალია) წითლით (როდესაც ის დაკავებულია). ნიშნის მარჯვენა მხარეს პატარა თეთრი ფერის ზოლი მიგვანიშნებს პატარა რიგის შესახებ. დიდი თეთრი ფერის ზოლი მიგვანიშნებს იმაზე, რომ რიგის სიგრძე აღემატება ათ ტრანზაქტს. დახურეთ **Facilities** ფანჯარა.

Blocks ფანჯრის ორივე ფორმა მისაღებია. ზოგჯერ საჭიროა იმის ცოდნა სად გროვდება ტრანზაქტები, ზოგჯერ კი საჭიროა ტრანზაქტების რაოდენობის ზუსტი ცოდნა სხვადასხვა ბლოკებში.

გახსენით ფანჯარა **Expression** (გამოსახულება). აირჩიეთ **Window ►Simulation Window ► Expression Window** (ფანჯარა ► მოდელირების პროცესის ფანჯარა ► „გამოსახულების“ ფანჯარა) ბრძანება. გაიხსნება ფანჯარა **Edit Expression** (გამოსახულების რედაქტორი) (ნახ. 10.26).



ნახ. 10.26. **Edit Expression** (გამოსახულების რედაქტორის) ფანჯარა შეტანილი გამოსახულებებითა და მათი ჭდეებით

გამოსახულების რედაქტორი დაკითხავს ჭდესა და გამოსახულებას, რომლის გამოთვლასაც თქვენ აპირებთ. ამისათვის დიალოგურ ფანჯარაში **Label** (ჭდე) ველში შეიტანეთ:

დრო

Expression ველში შეიტანეთ:

AC1

დააჭირეთ **View** ღილაკზე და შემდეგ კი **Memorize** (დამახსოვრება) ღილაკზე. ამის გაკეთება საჭიროა იმისათვის, რათა მიმდინარე მოდელირების პროცესში გვაჩვენოს ამ გამოსახულების მნიშვნელობები და დავათვალიეროთ აღნიშნული გამოსახულება მოდელირების პროცესის შენახვის შემდეგ და მისი განმეორებით გაშვებისათვის.

დიალოგურ ფანჯარაში **Label** (ჭდე) ველში ნაცვლად **დრო** შეიტანეთ:

აქტიური ტრანზაქტი

AC1–ის ნაცვლად შეიტანეთ:

XN1

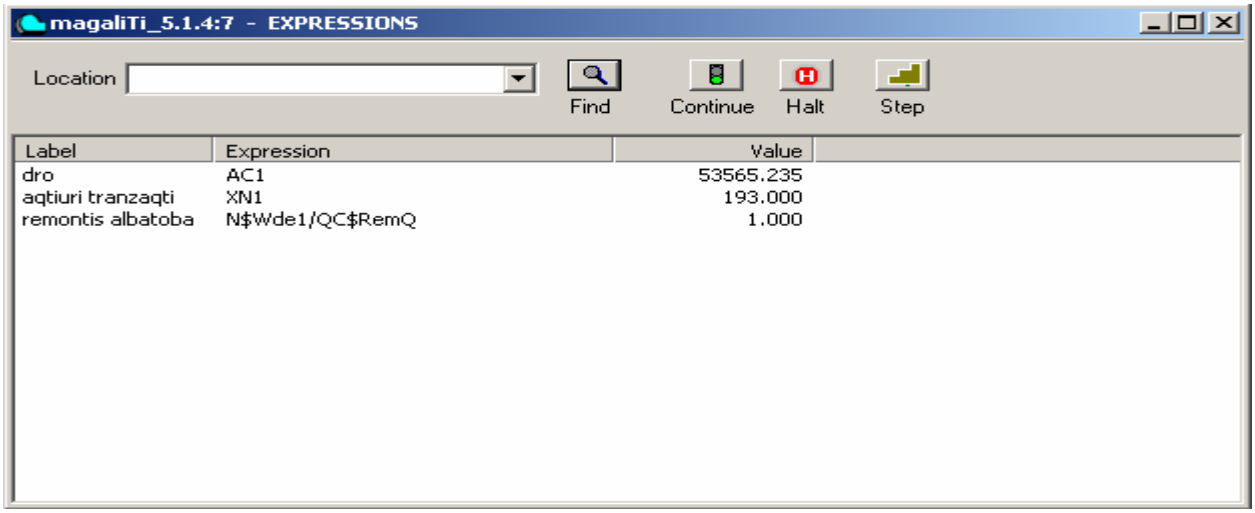
კვლავ დააჭირეთ **View** და **Memorize** ღილაკებს**.** ამ ღილაკებზე დაჭერა აუცილებელია ყოველი ჭდის ან გამოსახულების შეტანის შემდეგ.

შეიტანეთ ჭდე და მესამე გამოსახულება:

რემონტის ალბათობა

(N$Wde1/QC$RemQ)

კვლავ დააჭირეთ **View** და **Memorize** ღილაკებს. და ბოლოს **OK.** თქვენ დაინახავთ აბსოლუტურ სამოდელო დროს, აქტიური ტრანზაქტის ნომერს და კავშირის საშუალებების რემონტის ალბათობას (ნახ. 10.27). დააკვირდით გამოსახულების მნიშვნელობების ცვლილებას. დააჭირეთ **[F2]** ღილაკს და შემდეგ კი **[F4]** ღილაკს.



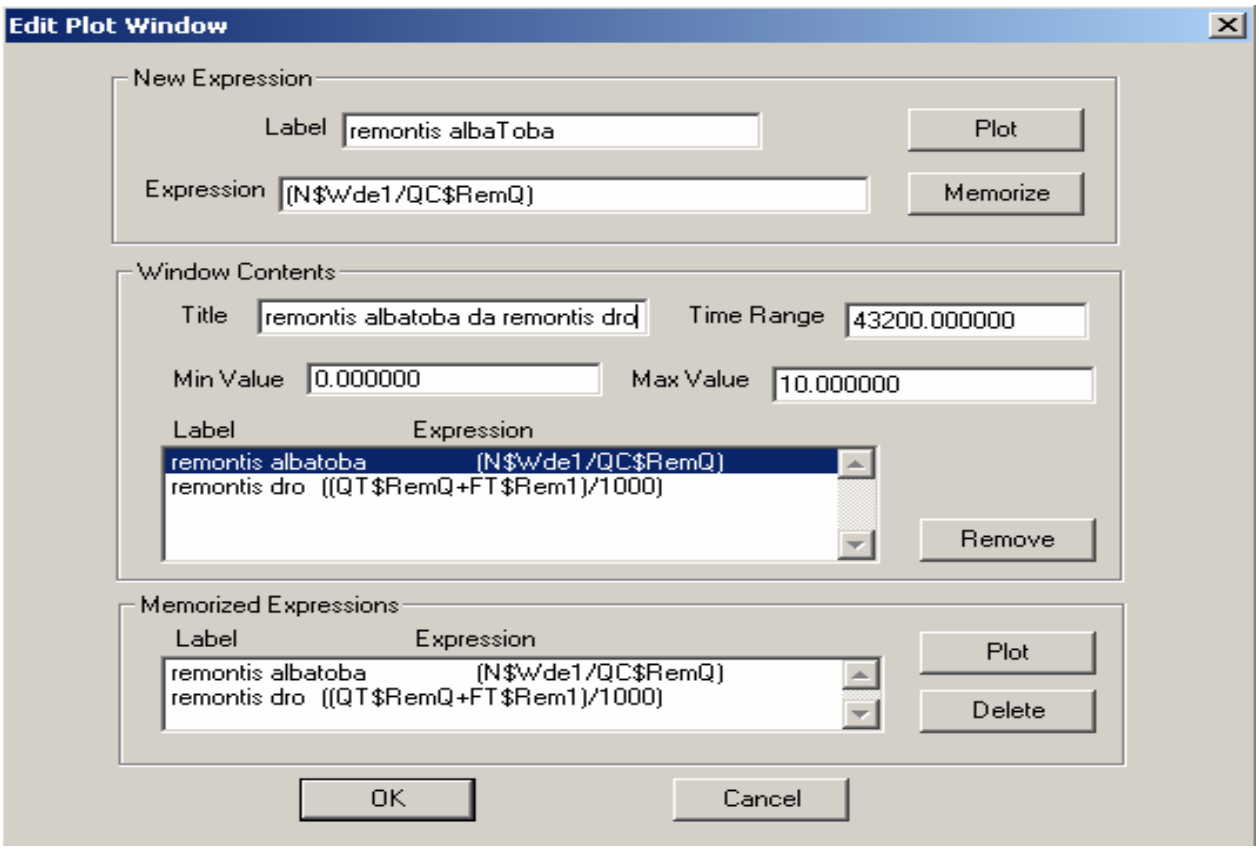
ნახ. 10.27. **Expression** ფანჯარა

ახლა დახურეთ **Expression** ფანჯარა შემდეგ ისევ გახსენით. გახსნილ **Edit Expression** ფანჯარაში (იხ. ნახ. 10.25) გამოსახულებები და მათი ჭდეები შენახულია მხოლოდ **Memorized Expressions** (შენახული გამოსახულებები) სიაში. აირჩიეთ ის გამოსახულება, რომელიც გსურთ რომ აისახოს. ამისათვის საჭიროა აირჩიოთ გამოსახულება **Memorized Expressions** სიაში და დააჭირეთ **View** ღილაკს. აირჩიეთ დრო და დააჭირეთ **View** ღილაკს, შემდეგ კი – რემონტის ალბათობას და კვლავ დააჭირეთ **View** ღილაკს. ორივე გამოსახულება გამოჩნდება ველში **Window**

Contents. დააჭირეთ **OK.** ამის შემდეგ თქვენ დაინახავთ **Expression** ფანჯარას, რომელიც აჩვენებს ორივე ტიპის გაუმართავი კავშირის საშუალებების აბსოლუტურ სამოდელო დროსა და რემონტის ალბათობას. დახურეთ ფანჯარა.

დაასრულეთ მოდელირების პროცესი, დახურეთ ყველა ფანჯარა ობიექტი „მოდელი“-ს გარდა. თავიდან აირჩიეთ **Command ► Greate Simulation** (ბრძანება ► მოდელირების პროცესის შექმნა) ბრძანება.

აირჩიეთ **Window ►Simulation Window ► Plot Window** (ფანჯარა ► მოდელირების პროცესის ფანჯარა ► „გრაფიკის“ ფანჯარა) ბრძანება. გამოჩნდება **Edit Plot** ფანჯარა (გრაფიკის რედაქტორი) (ნახ. 10.28).



ნახ. 10.28. დიალოგური ფანჯარა **Edit Plot** (გრაფიკის რედაქტორი) შეტანილი მონაცემებით

თქვენ ხედავთ, რომ ეს ფანჯარა ჰგავს **Edit Expression** ფანჯარას (გამოსახულებების რედაქტორი). **Edit Plot** ფანჯარა განკუთვნილია ღერძების, გამოსახულებებისა და სხვა საერთო მონაცემების, მათ შორის გრაფიკების მოსაცემად.

გრაფიკის ღერძების გაგების თვალსაზრისით **Label** ველში შეიძლება შეიტანოთ:

რემონტის ალბათობა

Expression ველში შეიტანეთ პირველი ცვლადი გრაფიკის ასაგებად:

(N$Wde1/QC$RemQ)

მოსალოდნელი რემონტის ალბათობა განისაზღვრება, როგორც გამართული კავშირის საშუალებების დამოკიდებულება რემონტში შესულ კავშირის საშუალებებთან.

ველში **Title** (სათაური) შეიტანეთ გრაფიკის სათაური, რომელიც აღწერს ორივე ცვლადს – რემონტის შესრულების ალბათობას და რემონტის დროს. აკრიფეთ:

რემონტის ალბათობა და რემონტის დრო

თითოეული გრაფიკისათვის უნდა მიუთითოთ დროის ინტერვალი. საჭიროა ეს ინტერვალი დიდი გააკეთოთ, რათა გრაფიკი არ იყოს პატარა და იგი თვალსაჩინოდ გამოიყურებოდეს. შეიტანეთ **X** ღერძის დროის ინტერვალი. ველში **Time Range**

(დროის ინტერვალი) აკრიფეთ 43 200 (მოდელირების დრო გამრავლებული გაშვების რაოდენობაზე). **Y** ღერძში ფარულად მითითებულია 0-დან 100-მდე მნიშვნელობები. დააყენეთ 10.

დააჭირეთ **Plot** (აგება) ღილაკს და შემდეგ კი – **Memorize** (დამახსოვრება) ღილაკს.

შეიტანეთ მეორე ცვლადი. ამისათვის **Label** ველში შეიძლება აკრიფოთ:

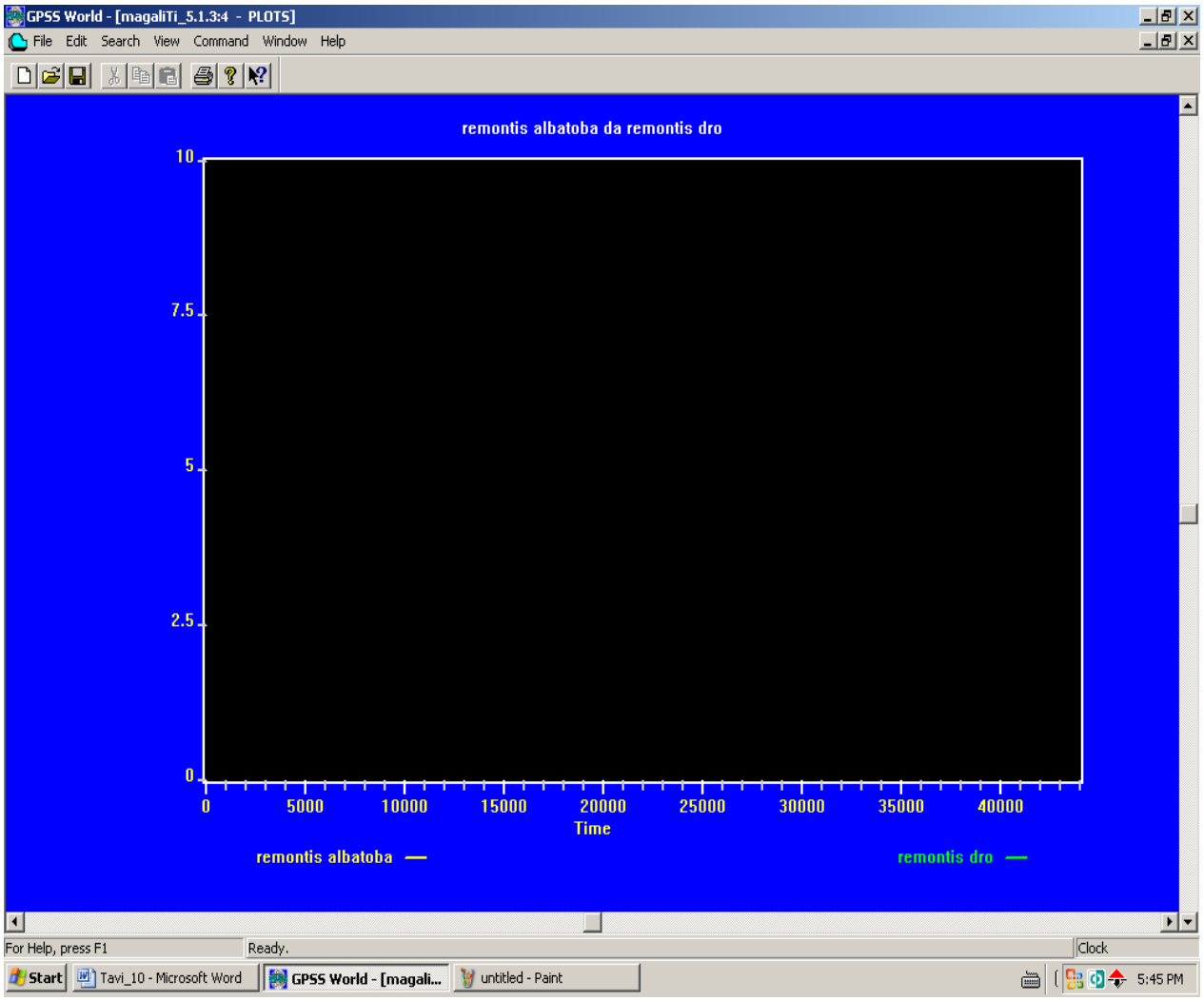
რემონტის დრო

Expression (გამოსახულება) ველში შეიტანეთ:

((QT$RemQ + FT$Rem1)/1000)

რემონტის საშუალო დრო განისაზღვრება, როგორც რიგში ყოფნის საშუალო დროისა და უშუალოდ რემონტში ყოფნის საშუალო დროის ჯამი. ასე, რომ ალბათობა არ შეიძლება აღემატებოდეს 1-ს, ხოლო რემონტის დრო უნდა აღემატებოდეს 1000, ამიტომ რემონტის დროისათვის შემოტანილია მასშტაბის 0, 001. ამ შემთხვევაში შესაძლებელია ერთ ფანჯარაში გადაფურცვლის გარეშე დააკვირდეთ ორივე გრაფიკს.

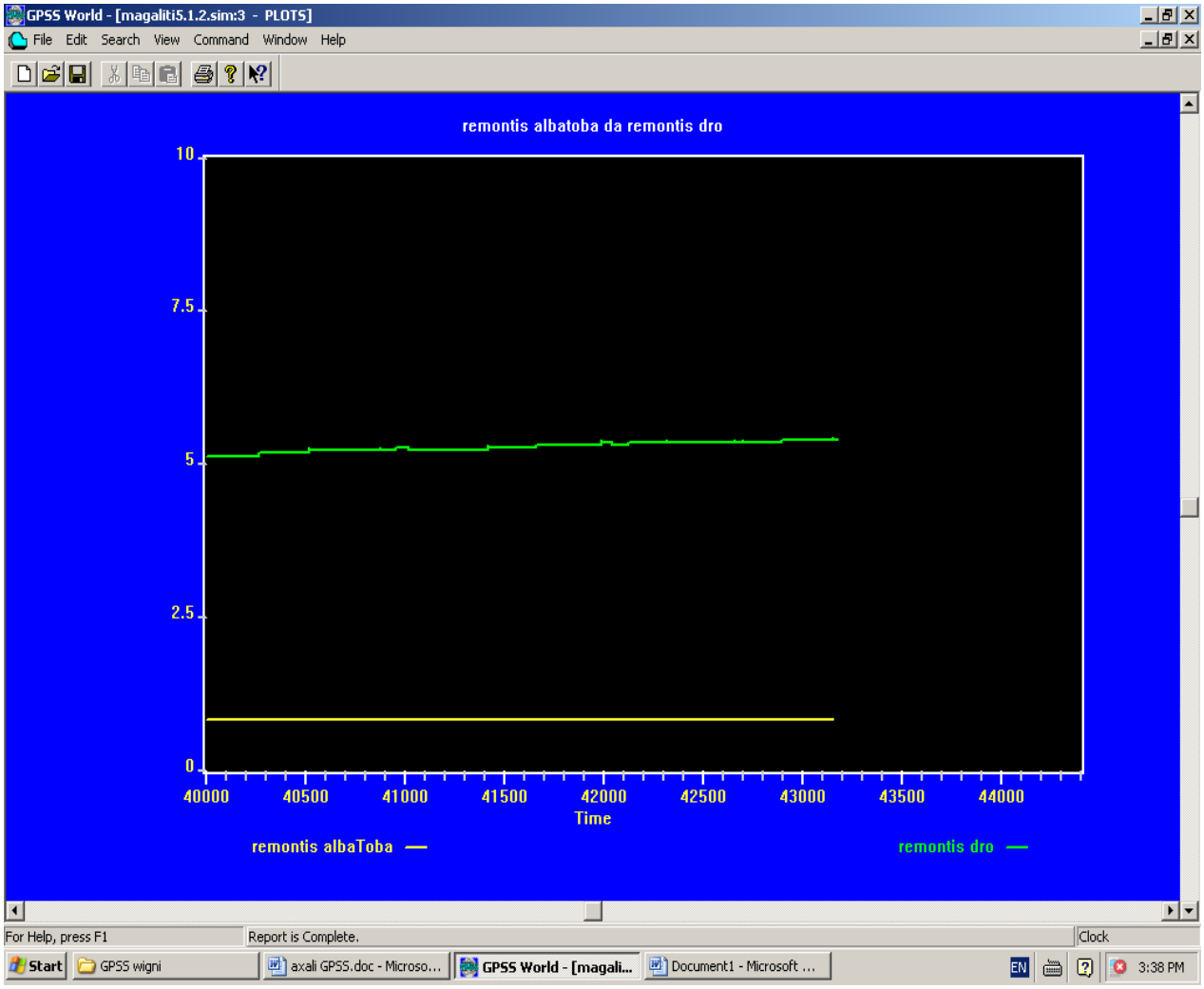
დააჭირეთ **Plot** (აგება) ღილაკს და შემდეგ კი – **Memorize** (დაიმახსოვრეთ) ღილაკს. და ბოლოს **OK. Plots** (გრაფიკის) ფანჯარა შეიძლება გამოიყურებოდეს 10.29 ნახაზზე წარმოდგენილი სახით.



ნახ. 10.29. **Plot** (გრაფიკის) ფანჯარა

თუ თქვენ ფანჯარას გახსნით არა მოდელირების პროცესის გაშვებამდე, არამედ მისი მიმდინარეობისას, მაშინ 10.29 ნახაზზე წარმოდგენილი სურათის ნაცვლად მიიღებთ სულ სხვა სურათს.

ახლა გაუშვით მოდელირების პროცესი და გაშვების რიცხვი მიუთითეთ 10. გაშვების მითითებული რიცხვის შემდეგ **Plots** (გრაფიკების) ფანჯარა შეიძლება გამოიყურებოდეს 10.30 ნახაზზე წარმოდგენილი სახით.



ნახ.10.30. **Plots** (გრაფიკების) ფანჯარა ორი აგებული გრაფიკით

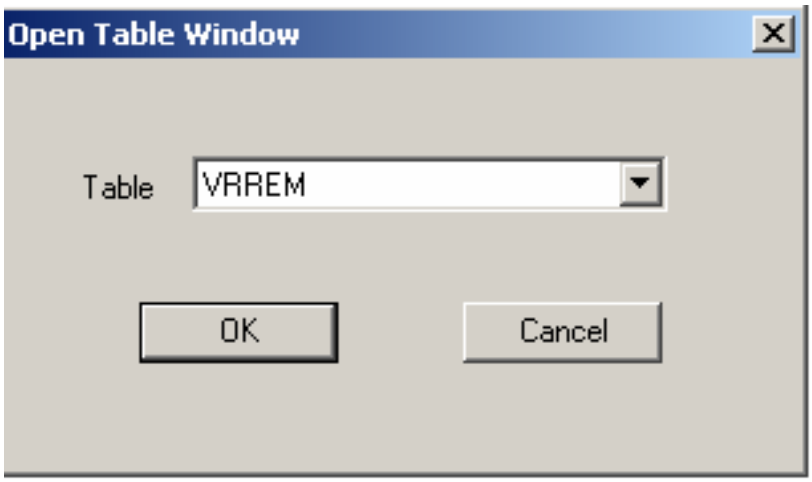
ერთ ფანჯარაში შეიძლება ააგოთ რვამდე გრაფიკი. ამისათვის საჭიროა შეიტანოთ შესაბამისი რაოდენობის ჭდეები და გამოსახულებები.

დახურეთ **Plots** (გრაფიკების) ფანჯარა. აირჩიეთ **File ► Save** ბრძანება. ფაილი შეინახეთ **magaliTi_5.1** სახელით. დახურეთ ყველა გახსნილი ფანჯარა. განაგრძეთ მოდელირების პროცესის შენახვა.

აირჩიეთ **File ► Open** ბრძანება. ფანჯრის ქვედა ნაწილში გამოსულ სიაში აირჩიეთ ფაილის ტიპი **Simulation** (მოდელირების პროცესი). შემდეგ დიალოგური ფანჯრის ზედა ნაწილში აირჩიეთ **magaliTi_5.1** და დააჭირეთ **Open** ღილაკს. გაიხსნება შენახული მოდელირების პროცესი. აირჩიეთ **Window ► Simulation Window ►Plot Window** (ფანჯარა ►მოდელირების პროცესის ფანჯარა ► გრაფიკის ფანჯარა) ბრძანება. გამოჩნდება **Edit ►Plots** ფანჯარა. **Memorized** სიაში (შენახული გამოსახულებები) გამოყავით პირველი ჭდე და დააჭირეთ **Plot** (აგება) ღილაკს, შემდეგ მეორე ჭდეს და აგრეთვე **Plot** (აგება) ღილაკს. შეიტანეთ **Title** ველში გრაფიკის სათაური „რემონტის ალბათობა და რემონტის დრო“, ველში **Time Range** (დროის ინტერვალი) – 43 200, ველში **Max Value** (მაქსიმალური მნიშვნელობა) – 10. ახლა კი ყველაფერი მზად არის გრაფიკების ასაგებად (ნახ.10.30).

მოდელირების პროცესი შენახული იყო **START** 10 ბრძანებაში მითითებული გაშვების რიცხვის დასრულებისას, ამიტომ თავიდან აირჩიეთ **START** ბრძანება და მიუთითეთ გაშვების საჭირო რიცხვი. გრაფიკის შეჩერება შესაძლებელია [**F4**] ღილაკის საშუალებით, ხოლო გაგრძელებისათვის დააჭირეთ [**F2**] ღილაკს. დახურეთ **Plots** ფანჯარა.

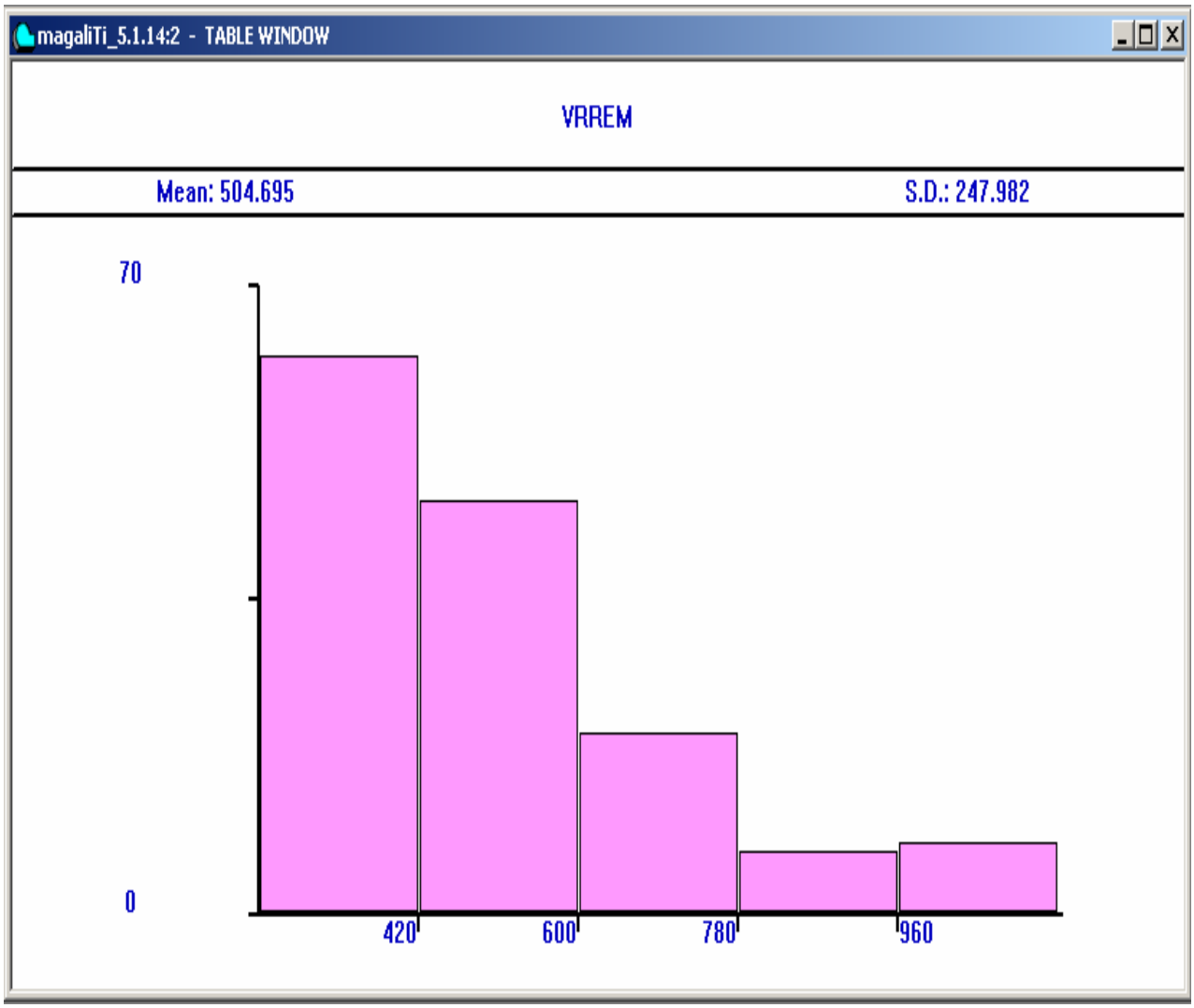
აირჩიეთ **Window ► Simulation Window ►Table Window** (ფანჯარა ► მოდელირების პროცესის ფანჯარა ► ცხრილის ფანჯარა) ბრძანება. გამოსულ **Open Table** ფანჯარაში აირჩიეთ **VrRem** და დააწკაპუნეთ **OK** (ნახ.10.31).



ნახ. 10.31. **Open Table** დიალოგური ფანჯარა

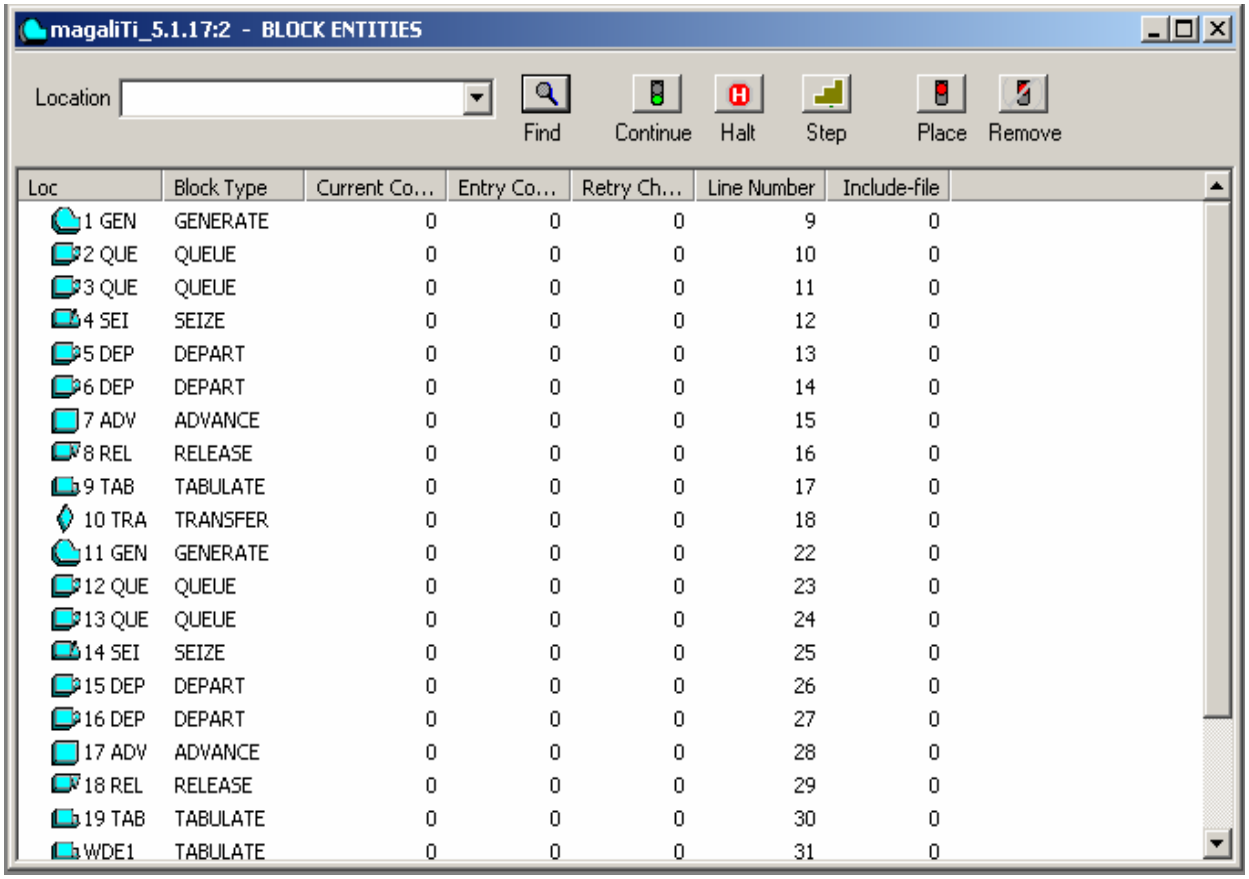
თქვენ დაინახავთ რომ**, VrRem** ცხრილის საფუძველზე აგებულ გრაფიკზე **X** და **Y** ღერძების გარდა არაფერი არ არის ნაჩვენები**.** გაუშვით მოდელირების პროცესი და მიუთითეთ **START** ბრძანებაში 10. დააკვირდით დიაგრამის ცვლილებას. ათი გაშვების შემდეგ გამოჩენილ ჰისტოგრამაზე ნაჩვენებია, რომ ორივე ტიპის გაუმართავი კავშირის საშუალებების საშუალო დრო შეადგენს 504, 695, საშუალო სტანდარტული გადახრა კი – 247, 982 (ნახ. 10.32).

დახურეთ ყველა ფანჯარა და კვლავ გაუშვით მოდელირების პროცესი. აირჩიეთ **Window ► Simulation Window ►Blocks Window** (ფანჯარა ► მოდელირების პროცესის ფანჯარა ► ბლოკების ფანჯარა) ბრძანება. **Blocks** ფანჯარაში აირჩიეთ **TRANSFER** ბლოკი (მე-10 ბლოკი). ამისათვის „მაუსის“ მაჩვენებელი მიიტანეთ მეათე ბლოკთან და დააწკაპუნეთ. შემდეგ დააჭირეთ **Place** ღილაკს **Blocks** ფანჯარაში ინსტრუმენტების პანელზე.



ნახ. 10.32. **Table** (ცხრილის) ფანჯარა

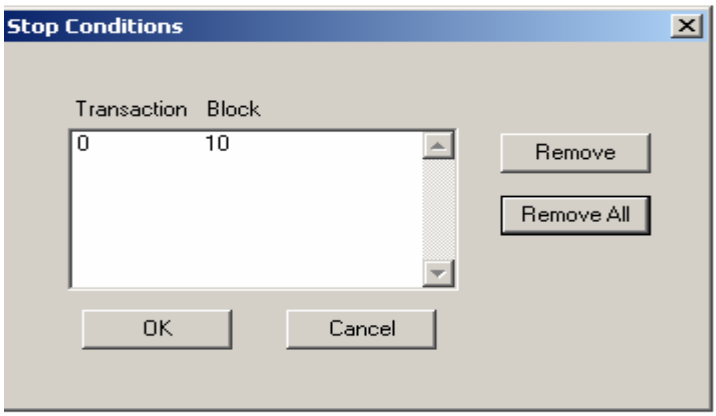
მეათე ბლოკში მოთავსდება გაჩერების პირობა. ამის გაკეთება აგრეთვე შესაძლებელია მთავარი ფანჯრის ინტერაქტიული ბრძანებების დახმარებით. გაუშვით მოდელირების პროცესი. აირჩიეთ **Command ► START** ბრძანება და გამოსულ დიალოგურ ფანჯარაში აკრიფეთ 10, დააწკაპუნეთ **OK**. მოდელირების პროცესი შეჩერდება როდესაც მომდევნო ტრანზაქტი ეცდება **TRANSFER** ბლოკში შესვლას (ნახ. 10. 33).



Loc	Block Type	Current Co...	Entry Co...	Retry Ch...	Line Number	Include-file
1 GEN	GENERATE	0	0	0	9	0
2 QUE	QUEUE	0	0	0	10	0
3 QUE	QUEUE	0	0	0	11	0
4 SEI	SEIZE	0	0	0	12	0
5 DEP	DEPART	0	0	0	13	0
6 DEP	DEPART	0	0	0	14	0
7 ADV	ADVANCE	0	0	0	15	0
8 REL	RELEASE	0	0	0	16	0
9 TAB	TABULATE	0	0	0	17	0
10 TRA	TRANSFER	0	0	0	18	0
11 GEN	GENERATE	0	0	0	22	0
12 QUE	QUEUE	0	0	0	23	0
13 QUE	QUEUE	0	0	0	24	0
14 SEI	SEIZE	0	0	0	25	0
15 DEP	DEPART	0	0	0	26	0
16 DEP	DEPART	0	0	0	27	0
17 ADV	ADVANCE	0	0	0	28	0
18 REL	RELEASE	0	0	0	29	0
19 TAB	TABULATE	0	0	0	30	0
WDE1	TABULATE	0	0	0	31	0

ნახ. 10.33. **TRANSFER** ბლოკში შესული ტრანზაქტის შეჩერება

წაშალეთ შეჩერების პირობა **TRANSFER** ბლოკში. ამისათვის აირჩიეთ **Window ► Simulation Snapshot ►User Stops** (ფანჯარა ► მოდელირების პროცესის კადრი ► შეჩერების პირობა) ბრძანება, დააჭირეთ **Remove All** (წაშალეთ ყველა) ღილაკს და **Ok** (ნახ. 10.34).



ნახ. 10.34. ფანჯარა **Stop Conditions** (შეჩერების პირობა)

მოდელის გამართვისა და ტესტირებისას ასევე გამოიყენება ანგარიშები. გადავიდეთ მათ განხილვაზე.

10.6. ანგარიშები

მოდელის მუშაობის დამთავრებისთანავე სისტემა ავტომატურად ქმნის ანგარიშს. სტატისტიკური ინფორმაცია გამოიტანება მხოლოდ იმ ობიექტების შესახებ, რომლებიც იმყოფება მოდელში. ზოგადად, სტანდარტული ანგარიში მოდელის საერთო მახასიათებლების გარდა შეიცავს შემდეგ მონაცემებს:

- მოდელის ობიექტების სახელების;
- მოდელის ბლოკების;
- მოწყობილობების;
- მრავალახიანი მოწყობილობების;
- რიგების;
- მომხმარებლის სიების;
- სტატისტიკური ცხრილების;
- მატრიცების;
- ტრანზაქტების ჯგუფების;
- რიცხვითი ჯგუფების;
- ლოგიკური გადამრთველების;
- შენარჩუნებული სიდიდეების;
- მიმდინარე მოვლენათა სიების;
- სამომავლო მოვლენათა სიების შესახებ.

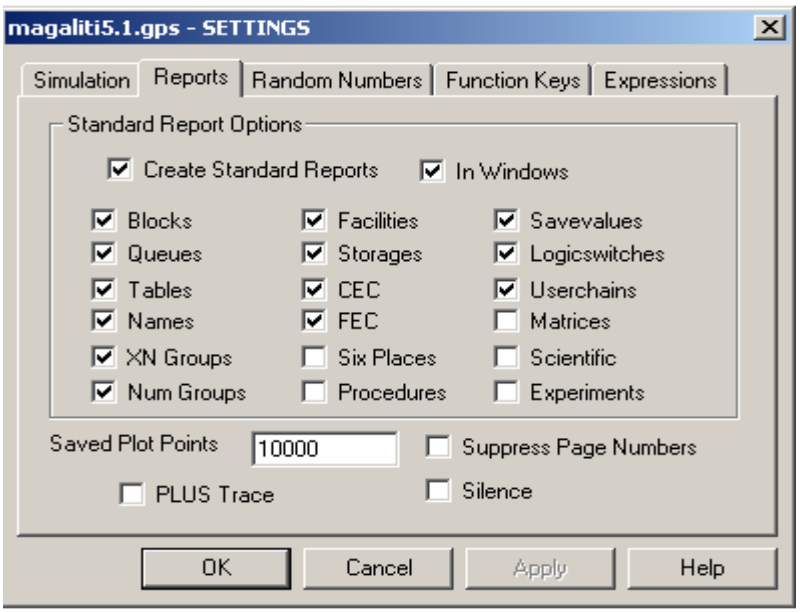
მიმდინარე და სამომავლო მოვლენათა სიების გამოტანის აუცილებლობას განსაზღვრავს **START** ბრძანების **D** ოპერანდი. თუ **D** ოპერანდი არ უდრის ნულს, მაშინ სიები გამოიტანება. ანგარიშის გამოტანა შესაძლებელია დაბლოკილ იქნას. ამოსათვის **START** ბრძანების **B** ოპერანდის ველში აუცილებელია ჩაიწეროს **NP**. სტანდარტული ანგარიშების შექმნა ასევე შესაძლებელია მოდელის დაყენების ჟურნალის **Reports** (ანგარიშები) გვერდზე **Create Standart Reports** (სტანდარტული ანგარიშების შექმნა) ალმის მოხნით.

თითოეულ ანგარიშს ენიჭება სახელი. ეს სახელი ფარულად იქმნება მოდელის ფაილის სახელიდან, ამ ფაილიდან შექმნილი მოდელირების პროცესის რიგითი ნომრიდან და მოცემული მოდელირების პროცესის ანგარიშის ნომრიდან. ანგარიშის ნუმერაცია იწყება 1-დან. ანგარიშის მქონე ფაილებს გააჩნიათ გაფართოება .qpr. მაგალითად, magaliTi_5.1.1.1.qpr, magaliTi_5.1.1.2.qpr და სხვ. რომელიმე ანგარიშების წაშლისას მათი რიგითი ნომრები მუშაობის სეანსში იქნება არათანამიმდევრობით.

სისტემის მიერ ფორმირებული ანგარიშის სახელის შეცვლა მომხმარებელს არ შეუძლია. ანგარიშები წარმოადგენს სპეციალურად დაფორმატებულ ფაილებს. თუ იქმნება იმის აუცილებლობა, რომ რაიმე ანგარიში შენახულ იქნას მომხმარებლის სახელით უნდა მოხდეს ასლის გადაღება გაცვლის ბუფერში, ხოლო მისგან – შესაბამის ფაილში. **Report**

(ანგარიში) ფანჯრიდან შესაძლებელია ნებისმიერი ანგარიშის დაბეჭდვა **File ► Print** მთავარი მენიუს ბრძანების საშუალებით.

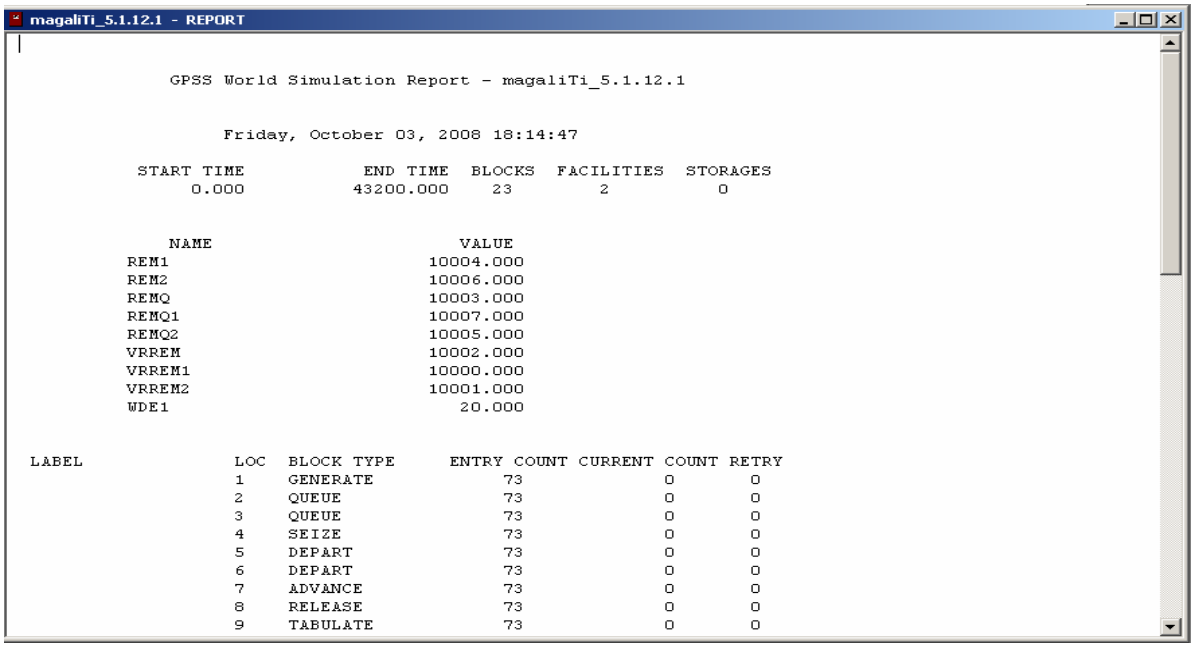
ანგარიში შესაძლებელია დავაყენოთ შემდეგნაირად. ამისათვის გახსენით მოდელი **magaliTi_5.1**. აირჩიეთ **Edit ► Settings** (რედაქტირება ► დაყენება), გადადით **Reports** (ანგარიშები) გვერდზე (ნახ. 10.35). თქვენ დაინახავთ **GPSS**-ის ობიექტების ჩამონათვალს, რომელთა შესახებ მონაცემები ფარულად ჩაირთვება ანგარიშებში. მომხმერებლის სურვილისამებრ ობიექტების ჩამონათვალი შეიძლება შეიცვალოს შესაბამისი ალმების მოხსნითა და დაყენებით.



ნახ. 10.35. მოდელის დაყენების ჟურნალის **Reports** (ანგარიშების) გვერდი

გაუშვით მოდელირების პროცესი. მისი დასრულებისთანავე ავტომატურად გამოჩნდება **Report** (ანგარიშის) ფანჯარა (ნახ. 10.36). ეს ფანჯარა ტექსტის რედაქტირების,შრიფტების შეცვლის,

ტექსტის ფერისა და დაბეჭდვის საშუალებას იძლევა. ანგარიშის ფაილის შენახვის შესახებ გამოსულ შეკითხვაზე უპასუხეთ **No**.



ნახ. 10.36. **Report** (ანგარიშის) ფანჯარა

გამოიყენეთ **Report** ბრძანება ანგარიშის მისაღებად დროის იმ მომენტში, როდესაც ტრანზაქტი შეეცდება შევიდეს თქვენი ინტერესის სფეროში მოქცეულ ბლოკში. ამისათვის დააწკაპუნეთ **Journal** (ჟურნალი) ფანჯრის ნებისმიერ ადგილზე. აირჩიეთ **Command** ►**CLEAR** (ბრძანება ►გასუფთავება) ბრძანება, შემდეგ **Window** ►**Simulation Window** ► **Blocks Window** (ფანჯარა ► მოდელირების პროცესის ფანჯარა ► „ბლოკების“ ფანჯარა). **Blocks** ფანჯარაში აირჩიეთ **TERMINATE** (21-ე ბლოკი) ბლოკი, შემდეგ დააჭირეთ **Blocks** ფანჯარის ინსტრუმენტების პანელზე **Place** (მოთავსება) ღილაკზე. 21-ე ბლოკში მოთავსებული იყო შეჩერბის პირობა. აირჩიეთ **Command** ►**START** ბრძანება, 1-ის ნაცვლად აკრიფეთ 100 და დააწკაპუნეთ **OK** ღილაკზე. ამის შემდეგ

მოდელირების პროცესი შეჩერდება და ტრანზაქტი შეჩერდება მე-20 ბლოკში, რომელიც მითითებულია **Wde1** ჭდით. აირჩიეთ **Command ►Custom** (ბრძანება ► არჩევა) ბრძანება, შეიტანეთ **REPORT** და **OK.** მაშინვე გამოჩნდება ანგარიშების ფანჯარა დროის იმ მომენტისათვის, როდესაც შეჩერდა მოდელირების პროცესი.

XI თავი
ტექსტური ობიექტებისა და მონაცემთა ნაკადების გამოყენება

მოდელი წარმოადგენს ოპერატორების ნაკრებს, რომლებიც შედის ერთ ობიექტში „მოდელი“ და არააუცილებელი ტექსტური ობიექტების ნებისმიერ რაოდენობაში.

ტექსტური ობიექტები მოდელის ოპერატორების ნაკრებებით ობიექტს „მოდელი“ უერთდება **INCLUDE** ბრძანების საფუძველზე (იხ. 10.3.3.6). **INCLUDE** ოპერატორები ასევე შეიძლება მიმაგრებულ იქნას ფუნქციონალურ ღილაკებზე. აღნიშნული, ღილაკის ერთი დაწკაპუნებით ობიექტს „მოდელირების პროცესი“ საშუალებას აძლევს გადასცეს ტექსტური ფაილის შემცველობაში მყოფი **Plus**–ოპერატორების ბრძანებათა ნაკრებიც. ტექსტური ობიექტები ასევე გამოიყენება მონაცემთა ნაკადებთან ერთად. მონაცემთა ნაკადები მოდელირების პროცესს საშუალებას აძლევს არსებული მონაცემები წაკითხულ იქნას ფაილებიდან და მხოლოდ შემდეგ მოხდეს მათი ჩაწერა ტექსტურ ფაილებში, ასევე შემდგომი გამოყენებისათვის შეიქმნას მოდელირების შედეგების ფაილები.

მონაცემთა ნაკადი არის მოდელირების პროცესის მიერ გამოყენებული ტექსტური სტრიქონების თანამიმდევრობა. არსებობს მონაცემთა ნაკადის ორი ტიპი:

- შეტანა-გამოტანის ნაკადები (**I/O** ან „ფაილი“-ს ნაკადები), რომლებიც საჭიროა ფაილებში შეღწევისათვის;
- ნაკადები მეხსიერებაში, რომლებიც საჭიროა ტესტირებისათვის ან შიდა მონაცემებში პირდაპირ შესაღწევად.

მონაცემთა ნაკადის ძირითადი ელემენტია – ტექსტური სტრიქონი, რომელიც ინტერვალების ჩათვლით სიმბოლოების სტრიქონს წარმოადგენს. მონაცემთა ნაკადების დასამუშავებლად არსებობს **GPSS World**-ის ხუთი ბლოკი: **OPEN, CLOSE, READ, WRITE, SEEK.** სამი მათგანი – **READ, WRITE, SEEK** ასრულებს ოპერაციებს ტექსტის მხოლოდ ერთ ცალკეულ სტრიქონთან. გადავიდეთ მონაცემთა ნაკადების დამუშავების ბლოკების განხილვაზე.

11.1. OPEN ბლოკი

OPEN ბლოკი განკუთვნილია მონაცემთა ნაკადის ინიციალიზაციისათვის.

ბლოკის ჩაწერის ფორმატს აქვს შემდეგი სახე:

OPEN A, [B], [C]

A ოპერანდი არის მონაცემთა ნაკადის დესკრიპტორი. ოპერანდი განსაზღვრავს მონაცემთა ნაკადის ტიპს. იგი მას ამუშავებს როგორც სტრიქონი. თუ იგი ნულოვანი სტრიქონია, ნაკადი იქმნება მეხსიერებაში. თუკი იგი არხის სახელია, ისეთი როგორც „**pipe\mypipe**“, იქმნება არხის ნაკადი. წინააღმდეგ შემთხვევაში იქმნება შეტანა-გამოტანის ნაკადი და ნავარაუდევია, რომ **A** ოპერანდი წარმოადგენს ფაილის სპეციფიკაციას.

B ოპერანდი – მონაცემთა ნაკადის ნომერი, მომხმარებლის მიერ მითითებული წარმოსახვითი დადებითი რიცხვი. ნაკადების ნუმერაცია შემოტანილია ერთდროულად მოდელირების ერთ პროცესში მონაცემთა რამდენიმე ნაკადის გამოყენების მიზნით. მონაცემთა ნაკადის ნომერი ფარულად უდრის 1.

C ოპერანდი – ბლოკის ჭდე, რომელშიც შედის ტრანზაქტი მონაცემთა ნაკადის ინიციალიზაციის შეცდომის შემთხვევაში. შეცდომათა კოდები:

- 0 – არ არის შეცდომა;
- 10 – ფაილის გრძელი სახელი;
- 11 – გარე ფაილის წაკითხვის შეცდომა;
- ფაილის გახსნის მცდელობის დროს აკრძალული იქნა მეხსიერებაში შეღწევა.

მაგალითად:

OPEN (“Gegma. txt”), 3, Kon1

OPEN (“C \ Model \ NorPogr. txt”)

OPEN (“ ”)

პირველ მაგალითში იხსნება შეტანა-გამოტანის ნაკადი, მას ენიჭება ნომერი 3 და გახსნისას შეცდომის შემთხვევაში აქტიური ტრანზაქტი შედის **Kon1** ჭდის მქონე ბლოკში. **A** ოპერანდით მითითებულია ფაილში შესვლის არასრული გზა, ამიტომ იგულისხმება, რომ გამოიყენება ობიექტი „მოდელირების პროცესი“-ს საქაღალდე. თუ მინიშნებული სახელის მქონე ფაილი არ იქნა აღმოჩენილი, ნავარაუდევია, რომ ფაილი იქმნება და შეცდომა არ წარმოიშვება.

მეორე მაგალითში **A** ოპერანდით მითითებულია ფაილში შეღწევის სრული გზა, შეტანა-გამოტანის გასახსნელ ნაკადს ფარულად ენიჭება ნომერი ერთი. გახსნისას შეცდომის შემთხვევაში აქტიური ტრანზაქტი მიემართება რიგის მიხედვით მომდევნო ბლოკში.

მესამე მაგალითში ნაკადი იხსნება მეხსიერებაში.

შენიშვნა: თუ **A** ოპერანდის მიერ მითითებული ფაილი არ იქნა აღმოჩენილი, **OPEN** ბლოკის დამუშავების დროს იგი სრულად იტვირთება ვირტუალურ მეხსიერებაში. ფაილში არსებული ყველა მონაცემი ინახება როგორც ობიექტი - „მოდელირები პროცესი“-ს ნაწილი იქამდე, ვიდრე მონაცემთა ნაკადი არ იქნება დახურული. მონაცემთა ყველა შესაძლო ცლილება ფაილის სისტემაში აისახება მხოლოდ მას შემდეგ, რაც მონაცემთა ნაკადი დაიხურება **CLOSE** ბლოკის ან **CLOSE ( )** ბიბლიოთეკური **Plus**-პროცედურის მიერ.

11.2. CLOSE ბლოკი

CLOSE ბლოკი განკუთვნილია მონაცემთა ნაკადის დახურვისათვის. ბლოკის ჩაწერის ფორმატს აქვს შემდეგი სახე:

CLOSE A, [B], [C]

A ოპერანდი არის ტრანზაქტის პარამეტრის ნომერი ან სახელი, რომელშიც იწერება მონაცემთა ნაკადის დახურვის შეცდომის კოდი. თუკი ამგვარი პარამეტრი არ არსებობს, იგი იქმნება.

B ოპერანდი – მონაცემთა დასახური ნაკადის ნომერი, ფარულად უდრის 1, ანუ იმ შემთხვევაში თუ **B** ოპერანდი არ გამოიყენება, იხურება ნაკადი ნომერი ერთი.

C ოპერანდი – ბლოკის ჭდე, რომელშიც მონაცემთა ნაკადის დახურვის შემთხვევაში შედის ტრანზაქტი. შეცდომათა კოდები:

- 0 – არ არის შეცდომა;
- 41 – შეტანა-გამოტანის შეცდომის გამო დისკოზე ფაილის ჩაწერა არ არის განხორციელებული;
- 42 – ფაილი არ იქნა გახსნილი.

მაგალითად:

CLOSE Parm_Error, (P1+1)

განხილულ მაგალითში **CLOSE** ბლოკი დახურავს მონაცემთა ნაკადს, რომლის ნომერიც დაწესებულია ფრჩხილებში ჩასმული

გამოსახულებით. აღნიშნული გამოსახულება გამოითვლება, დამრგვალდება და გამოიყენება მონაცემთა ნაკადის ნომრის სახით. შედეგი უნდა იყოს დადებითი მთელი რიცხვი. **CLOSE** ბლოკი შეტანა-გამოტანის ნაკადებისათვის ვირტუალური მეხსიერებიდან მონაცემებს იწერს დისკოს ფაილში.

11.3. READ ბლოკი

READ ბლოკი განკუთვნილია მონაცემთა ნაკადიდან ტექსტური სტრიქონის წასაკითხად. ბლოკის ჩაწერის ფორმატს აქვს შემდეგი სახე:

READ A, [B], [C]

A ოპერანდი არის ტრანზაქტის პარამეტრის ნომერი ან სახელი, რომელშიც იწერება მიმდინარე სტრიქონის პოზიციაზე მყოფი შეტანის ნაკადიდან ან მეხსიერების ნაკადიდან წაკითხული სტრიქონი.

წაკითხვის შემდეგ მიმდინარე სტრიქონის პოზიცია იზრდება ერთი ერთეულით. თუკი აქტიური ტრანზაქტის ასეთი პარამეტრი არ არსებობს, იგი იქმნება.

B ოპერანდი – მონაცემთა ნაკადის ნომერი, რომლისგანაც მიმდინარეობს წაკითხვა. იგი ფარულად ტოლია 1, ანუ თუ **B** ოპერანდი არ გამოიყენება, მაშინ წაკითხვა წარმოებს ნომერი ერთი ნაკადიდან.

C ოპერანდი – ბლოკის ჭდე, რომელშიც წაკითხვის შეცდომის შემთხვევაში შედის ტრანზაქტი. თუ **C** ოპერანდი არ გამოიყენება, შეცდომის კოდი მაინც ინახება. მის მისაღებად უნდა გამოვიყენოთ **CLOSE** ბლოკი. შეცდომათა კოდები:

- 21 – წაკითხვის განხორციელების მცდელობის დროს აკრძალული იქნა მეხსიერებაში შეღწევა;
- 22 – ფაილი არ იქნა გახსნილი.

მაგალითი:

READ Striqoni_text, 4, Wde5

მოყვანილ მაგალითში **READ** ბლოკი მონაცემთა ნომერი 4 ნაკადიდან კითხულობს ტექსტურ სტრიქონს და წერს მას **Striqoni_text** სახელის მქონე ტრანზაქტის პარამეტრში. თუკი ტექსტური სტრიქონი იკითხება შეუცდომლად, ხოლო აქტიური ტრანზაქტის პარამეტრი არ არსებობს, იგი აუცილებლად იქმნება. შეცდომის შემთხვევაში სტრიქონი არ ექვემდებარება წაკითხვას და აქტიური ტრანზაქტი შედის **Wde5** ჭდის მქონე ბლოკში.

11.4. WRITE ბლოკი

WRITE ბლოკი განკუთვნილია მონაცემთა ნაკადისთვის ტექსტური სტრიქონის გადასაცემად. ბლოკის ფორმატი:

WRITE A, [B], [C], [D]

A ოპერანდი არის ტექსტური სტრიქონი, რომელიც გადაცემულ უნდა იქნას მონაცემთა ნაკადისათვის.

B ოპერანდი – მონაცემთა ნაკადის ნომერი, ფარულად ტოლია 1-ის.

C ოპერანდი – ბლოკის ჭდე, რომელშიც ჩაწერის შეცდომის შემთხვევაში შედის ტრანზაქტი. შეცდომის კოდები:

- 0 – არ არის შეცდომა;
- 31 – ჩაწერის განხორციელების მცდელობის დროს აკრძალული იქნა მეხსიერებაში შეღწევა;
- 32 – ფაილი არ იქნა გახსნილი.

D ოპერანდი – აწესებს **WRITE** ბლოკის მუშაობის რეჟიმს. თუ **D** ოპერანდი არ გამოიყენება (ფარულად) ან უდრის **ON**-ს, **WRITE** ბლოკი მუშაობს ჩასმის რეჟიმში. თუ **D** ოპერანდი უდრის **OFF**-ს – ამ შემთხვევაში იგი მუშაობს შეცვლის რეჟიმში.

ჩასმის რეჟიმი:

- ყველა ტექსტური სტრიქონი, რომლებიც იმყოფება მიმდინარე სტრიქონის პოზიციაზე (ან პოზიციის წინ) გადაინაცვლებს ერთი პოზიციით;
- თუ მიმდინარე სტრიქონის პოზიცია ბოლო ტექსტური სტრიქონის ახლოს იმყოფება, იგი დგება მონაცემთა ნაკადის ბოლო სტრიქონის შემდეგ;
- ახალი ტექსტური სტრიქონის ასლი თავსდება მიმდინარე სტრიქონის პოზიციაზე;
- მიმდინარე სტრიქონის პოზიცია იზრდება ერთი ერთეულით.

შეცვლის რეჟიმი:

- თუ მიმდინარე სტრიქონის პოზიცია შორს იმყოფება ბოლო ტექსტური სტრიქონისაგან, ყველა შუალედური პოზიცია ივსება ნულოვანი ტექსტური სტრიქონებით;
- მიმდინარე სტრიქონის პოზიციაზე მყოფი ტექსტური სტრიქონი იშლება;
- ახალი ტექსტური სტრიქონის ასლი თავსდება მიმდინარე სტრიქონის პოზიციაზე;
- მიმდინარე სტრიქონის პოზიცია იზრდება ერთი ერთეულით.

მაგალითი:

WRITE “INITIAL MX$TDon (1, 1), 420”, 5, Wde3

მოყვანილ მაგალითში **WRITE** ბლოკი ნომერ 5 მონაცემთა ნაკადს გადასცემს ტექსტურ სტრიქონს. თუკი ადგილი აქვს შეცდომას, აქტიური ტრანზაქტი გადადის **Wde** ჭდის მქონე ბლოკში. წინააღმდეგ შემთხვევაში იგი შედის რიგის მიხედვით მომდევნო ბლოკში. თუკი მოცემულ შემთხვევაში მონაცემთა ნაკადი წარმოადგენს გამოტანის ნაკადს ან ნაკადს მეხსიერებაში, ჩაწერა წარმოებს ჩასმის რეჟიმში, ვინაიდან **D** ოპერანდი არ გამოიყენება.

შენიშვნა 1: თუ **OPEN, CLOSE, READ** და **WRITE** ბლოკების **C** ოპერანდები არ გამოიყენება, შეცდომა იგნორირებულია. ჩვეულებისამებრ, აღნიშნული ოპერანდები გამოიყენება იმისათვის,

რომ აქტიური ტრანზაქტი მიმართონ **CLOSE** ბლოკისაკენ, რომელიც შეცდომის კოდს ათავსებს ტრანზაქტის პარამეტრში. პარამეტრიდან შენარჩუნებულ უჯრედში შეიძლება ჩაწერილ იქნას შეცდომის კოდი და დასრულდეს მოდელირება. უჯრედის მნიშვნელობა გამოტანილი იქნება სტანდარტული ანგარიშის სახით.

შენიშვნა 2: ფაილიდან მონაცემთა ქვესტრიქონული ჩაწერის დროს შეიძლება აღმოჩნდეს, რომ მიმდინარე სტრიქონის პოზიციაზე არ არის მონაცემთა სტრიქონი. ეს ხდება მაშინ, როდესაც ფაილიდან ჩაწერილია ყველა ტექსტური სტრიქონი. მოცემულ შემთხვევაში აქტიური ტრანზაქტი შედის **READ** ბლოკში და მიემართება ბლოკში, რომლის ჭდეც მითითებულია **C** ოპერანდის მიერ. ამასთან, შეცდომის კოდი არ ნარჩუნდება.

11.5. SEEK ბლოკი

SEEK ბლოკი აწესებს მონაცემთა ნაკადის მიმდინარე სტრიქონის პოზიციას. ბლოკის ჩაწერის ფორმატს აქვს შემდეგი ფორმატი:

SEEK A, [B]

A ოპერანდი არის მიმდინარე სტრიქონის ახალი პოზიცია;

B ოპერანდი – მონაცემთა ნაკადის ნომერი, ფარულად ტოლია 1-ის.

მიმდინარე სტრიქონის პოზიცია წარმოადგენს ერთკავშირიან ინდექსს (მაჩვენებელს), რომელიც მიუთითებს მომდევნო სტრიქონის პოზიციას, რომელიც აუცილებლად უნდა იქნას დათვლილი ან ჩაწერილი. იგი არ შეიძლება იყოს ერთეულზე ნაკლები. მისი ერთეულზე ნაკლებად დადგენის მცდელობისას მიმდინარე სტრიქონის პოზიცია დგინდება ერთეულის ტოლად.

- 51 არის შეცდომის კოდი მიმდინარე პოზიციის შეცვლის დროს.

მაგალითი:

SEEK (Striqoni + P1), (Nakadi + 3)

მოცემულ მაგალითში როდესაც ტრანზაქტი შედის **SEEK** ბლოკში, **A** ოპერანდი გამოითვლება, დამრგვალდება და გამოიყენება როგორც მიმდინარე სტრიქონის პოზიციის ახალი ნომერი. ასევე გამოითვლება **B** ოპერანდი**,** დამრგვალდება და გამოიყენება როგორც მონაცემთა ნაკადის ნომერი. **A** და **B** ოპერანდები უნდა იყოს მთელი დადებითი რიცხვები.

ამოცანები მზა მოდელებით

ა მ ო ც ა ნ ა 1

ტურნიკეტის იმიტაციური მოდელირება

ფეხბურთის სტადიონზე მაყურებლები შემოდის თანაბარალბათურად ყოველ 7±5 წმ-ის ინტერვალში და დგება რიგში, რათა გაიაროს სტადიონი. თითოეული მაყურებლის ადგილამდე მისვლის დრო განაწილებულია თანაბარალბათურად 5±3 წმ-ის ინტერვალში.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს 300 მაყურებლის მიერ ტურნიკეტის გავლის დროს.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                * * * * * * * * * * * * * * * * * *
*                *  CPSS WORLD SIMULATION  *
*                * * * * * * * * * * * * * * * * * *

*         TURNSTILE PROGRAM  IN  GPSS World

*              MODEL SEGMENT 1

      GENERATE      7,5       ; მაყურებლის მოსვლის დროებს
                                შორის შუალედი
      QUEUE         TURN      ; რიგში ჩადგომა
      SEIZE         TURN      ; ტურნიკეტის დაკავება
      DEPART        TURN      ; რიგის დატოვება
```

```
        ADVANCE     5,3       ; ტურნიკეტის მომსახურების დრო
        RELEASE     TURN      ; ტურნიკეტის განთავისუფლება
        TERMINATE             ; ტურნიკეტის დატოვება

*       MODEL SEGMENT 2

        GENERATE    480       ; მოდელირების დრო
        TERMINATE   1         ; დასასრული
```

ა მ ო ც ა ნ ა 2

სატელეფონო სისტემის იმიტაციური მოდელირება

სატელეფონო ზარების განსახორციელებლად გაგვაჩნია კავშირის ორი ხაზი. დავუშვათ, რომ ზარები (ტრანზაქტები) შემოდის თანაბარალბათურად ყოველ 100±60 წმ-ის ინტერვალში. როდესაც ხაზი დაკავებულია აბონენტი რეკავს განმეორებით, რაზედაც ეხარჯება 5±1 წმ დრო. საუბარი ხორციელდება თანაბარალბათურად 3±1 წმ ინტერვალში. საჭიროა ისე გავანაწილოთ სამოდელო დრო, რომ ყველა აბონენტმა შეძლოს საუბარი.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს თუ რა დრო დასჭირდება 200 საუბრის განხორციელებას.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*               * * * * * * * * * * * * * * * * * *
*               *  CPSS WORLD SIMULATION  *
*               * * * * * * * * * * * * * * * * * *

*          TELEPHONE SYSTEM MODEL

*        STORAGE CAPACITY DEFINITIONS

SETS     STORAGE      2

*           MODEL SEGMENT 1

TRANSIT  TABLE     M1,100,100,20  ; სატრანზიტო დრო
         GENERATE  100,60         ; ზარების შემოსვლა
AGAIN    GATE SNF  SETS,OCCUPIED  ; ხელახალი მოსინჯვა
         ENTER     SETS           ; დაკავშირება (ხაზები
                                    დაკავებულია)
```

```
          ADVANCE   180,60          ; საუბრის დრო 3±1
                                       წმ-ის ინტერვალი
          LEAVE     SETS            ; ხაზები თავისუფალია
          TABULATE  TRANSIT         ; სატრანზიტო დროის
                                       ტაბულირება
          TERMINATE                 ; ტრანზაქტების წაშლა
OCCUPIED  ADVANCE   300,60          ; დალოდება 5±1 წმ-ის
                                       იტერვალში
          TRANSFER  ,AGAIN          ; ხელახალი მოსინჯვა

*              MODEL SEGMENT 2

          GENERATE    480           ; მოდელირების დრო
          TERMINATE   1             ; დასასრული
```

ა მ ო ც ა ნ ა 3
მარაგების დონის პერიოდული შემოწმება

გარკვეული რაოდენობის პროდუქციის მარაგი პერიოდულად კონტროლდება ყოველდღიური შემოწმების საფუძველზე. საწყისი მარაგია 1000 ერთეული. ყოველდღიური მოთხოვნა 63±40 ერთეულის ფარგლებში ვარირებს თანაბარალბათურად. მიზნობრივი მარაგი 1000 ერთეულის ტოლია, ანუ შეკვეთა ხორციელდება მიმდინარე მარაგისა და 1000 ერთეულის სხვაობით. თუ მიმდინარე შეკვეთა 800 და მეტი ერთეულია, ამ შემთხვევაში შეკვეთა ამ კვირაში არ სრულდება.

შეადგინეთ **GPSS** მოდელი 5 სამუშაო დღის განმავლობაში, რომელიც განსაზღვრავს მარაგების რაოდენობას 200 დღის განმავლობაში. ამასთან, განვსაზღვროთ ადგილი ექნება თუ არა მარაგის სრულად ამოწურვას.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                * * * * * * * * * * * * * * * * * *
*                *  CPSS WORLD SIMULATION  *
*                * * * * * * * * * * * * * * * * * *

*           PERIODIC REVIEW INVENTORY MODEL

            RMULT      39941          ; გენერატორის საწყისი
                                        მნიშვნელობის განსაზღვრა
STOCK       STORAGE    10000          ; მარაგის მიმდინარე
                                        დონე 10000
STOCK       TABLE      S$STOCK,100,100,20 ; მარაგის ყოველ-
                                        დღიური მოხმარების ცხრილი
```

```
ORDERQTY VARIABLE  TARGET-S$STOCK      ; შეკვეთის
                                       რაოდენობა =  მარაგის
                                       მიზნობრივ დონეს (1000) –
                                       არსებული მარაგი
DEMAND   VARIABLE  RN1@24 + 40 ; გამოთვლილი
                                         არითმეტიკული ცვლადი
TARGET   EQU       1000          ; საწყისი მარაგი
REORDER  EQU       800           ; შეკვეთის წერტილი


*            MODEL SEGMENT 1

*         The reorder process

         GENERATE  5,,,,1            ; ტრანზაქტი-მოთხოვნა
                                     შედის  ყოველ მეხუთე დღეს
         TEST L    S$STOCK,REORDER,SKIP  ; მარაგის მიმდ.
                                   დონე<შეკვეთის დონეზე
         ASSIGN    2,V$ORDERQTY    ; პარამეტრი 2(P2)=
                                  მარაგის შეკვეთის რაოდენობას
         ADVANCE   5               ; შეკვეთის შესრულებას
                                     სჭირდება 5 დღე
         ENTER     STOCK,P2        ; P2 პარამეტრით ხდება
                                     მარაგის შევსება
SKIP     TERMINATE                 ; განახლება არ წარმოებს

*            MODEL SEGMENT 2

*  The daily demand decrements quantity on hand

         GENERATE  1 ; ტრანზაქტების ყოველდღიური მოთხოვნა
         ASSIGN    1,V$DEMAND ; პარამეტრი 1(P1) = ყოველ-
                                  დღიური მოთხოვნა
         TABULATE  STOCK      ; ყოველდღიური მარაგის
                                  ტაბულირება
         TEST GE   S$STOCK,P1,STOCKOUT  ; მარაგის დონე
                             მეტია ან ტოლია ყოველდღიურ
                             მოთხოვნაზე
```

```
          LEAVE     STOCK,P1  ; მარაგის დონე მცირდება
                                ყოველდღიური მოთხოვნით
          TERMINATE           ; ყოველდღიური მოთხოვნა
STOCKOUT  TERMINATE           ; შესრულდება მხოლოდ
                              მაშინ, როდესაც ყოველდღიური მოთხოვნა
                              არ შეიძლება შესრულდეს

*           MODEL SEGMENT 3

*       Initialize the inventory

          GENERATE  ,,,1,10   ; მოთხოვნის მიმდინარე დონეს
                              გააჩნია მაღალი პრიორიტეტი,
                              ვიდრე სხვა ტრანზაქტს
          ENTER     STOCK,TARGET ; მოთხოვნის მიმდინარე
                                   დონე = საწყისი მარაგი
          TERMINATE   1           ; დასასრული
```

ა მ ო ც ა ნ ა 4

ტელევიზორების სახელოსნოს იმიტაციური მოდელირება

ტელევიზორების სახელოსნოში მუშაობს ერთი ოსტატი, რომელიც აწარმოებს საწარმოს მიერ განვადებით გაცემული, აგრეთვე კლიენტების მიერ პროფილაქტიკური მომსახურებისა და სწრაფი რემონტის ჩატარების მიზნით მოტანილი ტელევიზორების რემონტს. ფირმის კუთვნილი ტელევიზორები რემონტზე შემოდის 40±8 სთ-ის ინტერვალით და იკავებს 10±1 სთ სარემონტო დროს. ჩავთვალოთ, რომ ეს არის სრული რემონტი. სწრაფი რემონტი, როგორიც გახლავთ დამცველების შეცვლა, მომართვა და რეგულირება სრულდება იმავე წუთს. ტელევიზორები, რომლებიც საჭიროებს სწრაფ რემონტს შემოდის ყოველი 90±10 წთ-ის ინტერვალში და მათ მომსახურებაზე იხარჯება 15±5 წთ. ტელევიზორები, რომლებიც საჭიროებს პროფილაქტიკურ მომსახურებას შემოდის 5±1 სთ-ის განმავლობაში და დათვალიერებაზე მოითხოვს 120±30 წთ-ს. სატელევიზიო მიმღებების პროფილაქტიკურ მომსახურებას გააჩნია გაცილებით უფრო მაღალი პრიორიტეტი, ვიდრე სრულ რემონტს.

შეადგინეთ **GPSS** მოდელი, რომელიც გაანალიზებს სახელოსნოს მუშაობას 50 სამუშაო დღის განმავლობაში. განსაზღვრეთ ოსტატის დაკავებულობისა და მომსახურების დაყოვნების ხარისხი.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*               * * * * * * * * * * * * * * * * * *
*               *  CPSS WORLD SIMULATION  *
*               * * * * * * * * * * * * * * * * * *

*             TELEVISION MAINTENANCE MAN

*********************************************************
*   REPAIR OF RENTED SETS ONE EACH WEEK   *
*********************************************************
*                 MODEL SEGMENT 1

        GENERATE    2400,480,,,1     ; მოტანის დროებს
                                       შორის ინტერვალი (ფირმის
                                       კუთვნილი ტელევიზორების რემონტი)
        QUEUE       OVERHAUL         ; რიგი სერვისის ადგილზე
        QUEUE       ALLJOBS          ; რიგი სრული რემონტისა
                                     და რემონტის სხვა ადგილებში
        SEIZE       MAINTENANCE      ; ხელოსანი დაკავებულია
        DEPART      OVERHAUL         ; სერვისის ადგილის
                                       დატოვება
        DEPART      ALLJOBS ; სრული რემონტისა და რემონტის
                              სხვა ადგილების დატოვება
        ADVANCE     600,60  ; რემონტის მომსახურების დრო
        RELEASE     MAINTENANCE      ; ხელოსანი თავისუფალია
        TERMINATE                ; სარემონტო ადგილის დატოვება

*                 MODEL SEGMENT 2

*           On the spot repairs

        GENERATE    90,10,,,3          ; მოტანის დროებს
                                     შორის ინტერვალი (სწრაფი
                                     რემონტის შემთხვევაში)
        QUEUE       SPOT             ; რიგი სწრაფი რემონტის
                                       ადგილზე
        QUEUE       ALLJOBS          ; რიგი სრული რემონტისა და
                                       რემონტის სხვა ადგილებში
```

```
          PREEMPT     MAINTENANCE      ; ხელოსანი დაკავებულია
          DEPART      SPOT             ; სწრაფი რემონტის
                                         ადგილის დატოვება
          DEPART      ALLJOBS          ; სრული რემონტისა და
                                   რემონტის სხვა ადგილების დატოვება
          ADVANCE     15,5             ; სწრაფი რემონტის
                                         მომსახურების დრო
          RETURN      MAINTENANCE      ; ხელოსანი
                                           თავისუფალია
          TERMINATE                 ; სარემონტო ადგილის დატოვება

*                   MODEL SEGMENT 3

*          Normal repairs on customer owned sets

          GENERATE    300,60,,,2   ; მოტანის დროებს შორის
                                     ინტერვალი (პროფილაქტიკური
                                     რემონტის შემთხვევაში)
          QUEUE       SERVICE      ; რიგი პროფილაქტიკური
                                     რემონტის ადგილზე
          QUEUE       ALLJOBS      ; რიგი სრული რემონტისა და
                                     რემონტის სხვა ადგილებში
          PREEMPT     MAINTENANCE,PR  ; ხელოსანი დაკავებულია
          DEPART      SERVICE          ; პროფილაქტიკური
                                     რემონტის ადგილის დატოვება
          DEPART      ALLJOBS    ; სრული რემონტისა და რემონტის
                                     სხვა ადგილების დატოვება
          ADVANCE     120,30           ; პროფილაქტიკური
                                     რემონტის მომსახურების დრო
          RETURN      MAINTENANCE      ; ხელოსანი თავისუფალია
          TERMINATE                 ;სარემონტო ადგილის დატოვება

*                MODEL SEGMENT 4

          GENERATE    2400             ;მოდელირების დრო
          TERMINATE   1                ; დასასრული
```

ა მ ო ც ა ნ ა 5

ხარისხის კონტროლის სისტემის იმიტაციური მოდელირება

დეტალების დასამზადებლად თანამიმდევრულად სრულდება სამი ოპერაცია, სადაც თითოეული მათგანის შესრულების შემდგომ წარმოებს ორწუთიანი კონტროლი. პირველი ოპერაციის შემდეგ კონტროლს არ ექვემდებარება დეტალების 20%. მეორე და მესამე ოპერაციების შემდეგ კონტროლს შესაბამისად არ გადის დეტალების 15 და 5%. კონტროლს არდაქვემდებარებული დეტალების 60% მიდის წუნდებაში, ხოლო დარჩენილი 40% საჭიროებს ოპერაციის ხელახალ განხორციელებას. დეტალების შემოსვლა ხორციელდება ექსპონენციალურად განაწილებული ყოველი 30 წუთის ინტერვალში. პირველი ოპერაციის შესრულების დრო მოცემულია შემდეგი ცხრილით.

პირველი ოპერაციის შესრულების დრო:

სიხშირე	.05	.13	.16	.22	.19	.15
ოპერაციის შესრულების დრო (წუთი)	10	14	21	32	38	45

მეორე ოპერაცია სრულდება 15±6 წთ-ის ინტერვალით, ხოლო მესამე ოპერაცია სრულდება ნორმალური განაწილებით საშუალოდ 24 წთ-ის განმავლობაში და სტანდარტული გადახრით 4 წთ.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს:

ა) დასამუშავებელი დეტალის ოპერაციის შესრულების დროს;

ბ) განსაზღვრეთ დროითი დანახარჯები და დაწუნებული დეტალების რიცხვი.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                * * * * * * * * * * * * * * * * * *
*                *  CPSS WORLD SIMULATION  *
*                * * * * * * * * * * * * * * * * * *

*                   QUALITY CONTROL PROGRAM

  RMULT  93211 ; გენერატორის საწყისი მნიშვნელობის განსაზღვრა

*         TABLE DEFINITIONS
 TRANSIT TABLE  M1,100,100,20    ; სატრანზიტო დროის
                                   განსაზღვრის ცხრილი

*        FUNCTION DEFINITIONS

 XPDIS FUNCTION RN1,C24; ექსპონენციალური განაწილების ფუნქცია
0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915
.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52
.94,2.81/.95,2.99/.96,3.2/.97,3.5/.98,3.9/.99,4.6/.995,5.3
.998,6.2/.999,7/.9998,8

 SNORM FUNCTION RN1,C25    ; ნორმალური განაწილების ფუნქცია
0,-5/0.00003,-4./.00135,-3/.00621,-2.5/.02275,-2./.06681,-1.5
.11507,-1.2/.15866,-1./.21186,-.8/.27425,-.6/.34458,-.4/.42074,-.2
.5,0/.57926,.2/.65542,.4/.72575,.6/.78814,.8/.84134,1/.88493,1.2
.93319,1.5/.97725,2/.99379,2.5/.99865,3/.99997,4/1,5

  PROCESS    FUNCTION  RN1,D7  ; დისკრეტული ფუნქცია
0,0/.05,10/.18,14/.34,21/.56,32/.85,38/1.0,45

*           VARIABLE DEFINITIONS
  THIRD    FVARIABLE 24+4#FN$SNORM    ; არითმეტიკული
                                        ცვლადის განსაზღვრა
```

```
*            MODEL SEGMENT 1

             GENERATE 30,FN$XPDIS    ; დეტალის შემოსვლებს
                                       შორის შუალედი
             ASSIGN   1,FN$PROCESS   ; P1 = პროცესის დრო
 STAGE1      SEIZE    MACHINE1       ; I კონტროლერის დაკავება
             ADVANCE  P1             ; I პროცესის დრო
             RELEASE  MACHINE1       ; I კონტროლერის განთავისუფლება
             ADVANCE  2              ; I ინსპექტირების დრო
             TRANSFER .200,,REWORK1  ; ნამუშევარის 20%-ის
                                       ხელახალი კონტროლი
 STAGE2      SEIZE    MACHINE2       ; II კონტროლერის დაკავება
             ADVANCE  15,6           ; II პროცესის დრო
             RELEASE  MACHINE2       ; IIკონტროლერის განთავისუფლება
             ADVANCE  2              ; II ინსპექტირების დრო
             TRANSFER .150,,REWORK2  ; ნამუშევარის 15%-ის
                                       ხელახალი კონტროლი
 STAGE3      SEIZE    MACHINE3       ; III კონტროლერის დაკავება
             ADVANCE  V$THIRD        ; III პროცესის დრო
             RELEASE  MACHINE3       ; IIIკონტროლერის განთავისუფლება
             ADVANCE  2              ; III ინსპექტირების დრო
             TRANSFER .05,,REWORK3   ; ნამუშევარის 5%-ის
                                       ხელახალი კონტროლი
TABULATE     TRANSIT                 ; სატრანზიტო დროის ტაბულირება
             TERMINATE    1          ; I პროცესის დასასრული
 REWORK1     TRANSFER  .400,,STAGE1  ; გადასვლა STAGE1-ზე
             TERMINATE               ; დასასრული
 REWORK2     TRANSFER  .400,,STAGE2  ; გადასვლა STAGE2-ზე
             TERMINATE               ; დასასრული
 REWORK3     TRANSFER  .400,,STAGE3  ; გადასვლა STAGE3-ზე
             TERMINATE               ; დასასრული

*            MODEL SEGMENT 2

             GENERATE     480        ; მოდელირების დრო
             TERMINATE    1          ; დასასრული
```

ა მ ო ც ა ნ ა 6

შეკვეთის წერტილის მიხედვით მარაგების სისტემის იმიტაციური მოდელირება

მარაგების სისტემა იმართება ე. წ. შეკვეთების წერტილით, რომლის დროსაც სრულდება 600 ერთეულით განსაზღვრული 500 ერთეულის ტოლი ეკონომიკური სიდიდის მქონე შეკვეთა. შეკვეთის საწყისი სიდიდე უდრის 700 ერთეულს. დღიური მოთხოვნა მერყეობს თანაბარალბათურად განაწილებულ 63±40 ინტერვალში. შეკვეთის შესრულებიდან საქონლის ჩატანამდე ლოდინის დრო ერთი კვირის (5 სამუშაო დღე) ტოლია.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს მარაგების განაწილებასა და ფაქტიურ დღიურ გაყიდვებს 100 სამუშაო დღის განმავლობაში.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*              * * * * * * * * * * * * * * * * * *
*              *  CPSS WORLD SIMULATION  *
*              * * * * * * * * * * * * * * * * * *

*              ORDER POINT INVENTORY SYSTEM

*          INITIALIZE

          INITIAL    X$EOQ,500    ; ეკონომიკური სიდიდის შეკვეთა
          INITIAL    X$POINT,600  ; შეკვეთის წერტილი
          INITIAL    X$STOCK,700  ; შეკვეთის საწყისი სიდიდე=700

*           TABLE DEFINITIONS
INVENTORY TABLE     X$STOCK,0,50,20    ; შეკვეთის ოდენობა
SALES     TABLE     P$DEMAND,38,2,20   ; გაყიდვების ოდენობა
```

```
*          VARIABLE DEFINITIONS
VAR2       VARIABLE    RN1@24+40      ; არითმეტიკული ცვლადის
                                        განსაზღვრა

*             MODEL SEGMENT 1

           GENERATE    ,,,1                ; შეკვეთების შემოსვლა
AGAIN      TEST L      X$STOCK,X$POINT     ; შეკვეთის წარმატებით
                                             შესრულება
           ADVANCE     5                   ; შეკვეთის ლოდინის
                                             დრო =1 კვირა
           SAVEVALUE STOCK+,X$EOQ  ; ეკონომიკური შეკვეთის სიდიდე
           TRANSFER    ,AGAIN      ; კიდევ მოსინჯეთ

*             MODEL SEGMENT 2

             GENERATE  ,,,1        ; ყოველდღიური მოთხოვნის
                                     ცვლილება
             ASSIGN    DEMAND,V$VAR2     ; ყოველდღიური მოთ-
                                           ხოვნის მინიჭება
             TABULATE  INVENTORY ;შეკვეთის ჩანაწერის ტაბულირება
             TEST GE   X$STOCK,P$DEMAND ;შეკვეთის საწყისი სი-
                                         დიდე <= დღიურ მოთხოვნაზე
            SAVEVALUE  STOCK,P$DEMAND   ;მოთხოვნის საწყისი სი-
                                         დიდეები ინარჩუნებს გაყიდული
                                         სიდიდის ტოლ მნიშვნელობას
             SAVEVALUE SOLD,P$DEMAND ;გაყიდვები = დღიურ
                                           მოთხოვნას
             TABULATE  SALES     ; გაყიდვების სიდიდის ტაბულირება
             TERMINATE 1         ; გაყიდული საქონელი
             PLOT      X$SOLD,60,0,100  ;ყოველდღიური მოთხოვნა
             PLOT      X$STOCK,800,0,100; შეკვეთის სიდიდე

   *          MODEL SEGMENT 3

             GENERATE   480       ; მოდელირების დრო
             TERMINATE  1         ; დასასრული
```

ა მ ო ც ა ნ ა 7

ელექტრონული საათების საწარმოო საამქროს იმიტაციური მოდელირება

ელექტრონული ფირმის საწარმოო საამქრო ამზადებს ელექტრონულ საათებს. შემფუთავ საამქროში ლამაზ პაკეტებში საათები შეფუთვას ექვემდებარება ავტომატური შემფუთავი მანქანის მეშვეობით იმ რაოდენობით, რომლებიც შეკვეთილია საცალო ვაჭრობის ქსელის მიერ. შეკვეთის ოდენობა მოცემულია შემთხვევითი ფუნქციით.

შეკვეთის ოდენობა:

სიხშირე	10	25	30	15	12	05	03
შეკვეთის ოდენობა	6	12	18	24	30	36	48

შეკვეთების მიღებებს შორის საშუალო დრო შეადგენს 15 წუთს და განაწილებულია ექსპონენციალურად. თითოეული საათის შეკვეთის შეფუთვის დრო განაწილებულია თანაბარალბათურად 120±10 წმ-ის ინტერვალში. საწარმოო საამქრო ელექტრონულ საათებს უშვებს 45±5 წთ-ის ინტერვალით, პარტიებით 60 ცალი.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს:

ა) შემფუთავ საამქროში რიგის მომლოდინე შეკვეთების საშუალო რაოდენობას;

ბ) ყოველდღიურად გაგზავნილი საათების რაოდენობას;

გ) შეკვეთების შესრულების დროის განაწილებას.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                * * * * * * * * * * * * * * * * * *
*                *  CPSS WORLD SIMULATION  *
*                * * * * * * * * * * * * * * * * * *


*                 MANUFAKTURING COMPANY

*    TIME UNIT IS ONE SECOND

*               FUNCTION DEFINITIONS

XPDIS FUNCTION RN1,C24 ;ექსპონენციალური განაწილების ფუნქცია

 0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915
.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52
.94,2.81/.95,2.99/.96,3.2/.97,3.5/.98,3.9/.99,4.6
.995,5/.3.998,6.2/.999,7/.9998,8

    SIZEORDER FUNCTION  RN1,D7       ; დისკრეტული ფუნქცია
.1,6/.35,12/.65,18/.8,24/.92,30/.97,36/1,48

TRANSIT TABLE         M1,100,100,20    ; სატრანზიტო დრო
NUMBER  TABLE         X1,100,100,20    ; ყოველდღიური
                                         შეკვეთის განსაზღვრა
PTIME   VARIABLE      10#P1+120 ; შეფუთვის დრო
AMOUNT  EQU           1000      ; საწყისი შეკვეთის განსაზღვრა
STOCK   STORAGE       4000      ; შეკვეთის სიდიდე

*             MODEL SEGMENT 1

        GENERATE      900,FN$XPDIS     ;შეკვეთების შემოსვლა
        ASSIGN        1,1,SIZEORDER    ;P1 = შეკვეთის სიდიდე
        TEST GE       S$STOCK,P1,STOCKOUT  ;საკმარისია
                                             შეკვეთა?
        LEAVE         STOCK,P1  ; შეკვეთის სიდიდეზე გადასვლა
        QUEUE         PACKING   ;შეფუთვის რიგში ჩადგომა
        SEIZE         MACHINE   ; შემფუთავი მანქანის დაკავება
        DEPART        PACKING   ; შეფუთვის რიგის დატოვება
        ADVANCE       V4PTIME   ; შეფუთვის დრო
        RELEASE       MACHINE   ; შემფუთავი მანქანის
                                    განთავისუფლება
        SAVEVALUE     1+,P1     ; შეფუთვა
```

```
          TABULATE  TRANSIT  ; სატრანზიტო დროის ტაბულირება
          TERMINATE          ; სატრანზიტო დროის დასასრული
STOCKOUT  TERMINATE          ; დასასრული

*         MODEL SEGMENT 2

          GENERATE      2700,300,1  ; ელ. საათების გამოშვებათა
                                      შორის ინტერვალი
          ENTER         STOCK,60    ;შეკვეთა ტოლია 60 ცალის

*         MODEL SEGMENT 3

          GENERATE      28800       ; მოდელირების დრო
          TABULATE      NUMBER      ;ყოველდღიური შეკვეთის
                                      ტაბულირება
          SAVEVALUE     1,0         ;შენარჩუნებულია ყოველდღიური
                                      შეკვეთის სიდიდე
          TERMINATE     1           ; დასასრული

*         MODEL SEGMENT 4

          GENERATE    ,,,1,10   ;პირველი შეკვეთის შემოსვლის დრო
          ENTER       STOCK,AMOUNT  ;შეკვეთის მიღება
          TERMINATE                 ;შეკვეთის მიღების დასრულება

*         MODEL SEGMENT 5

          GENERATE                  ; მოდელირების დრო
          TERMINATE     1           ; დასასრული
```

ა მ ო ც ა ნ ა 8
ნავთობბაზის იმიტაციური მოდელირება

ნავთობბაზა ანაწილებს სამი სახეობის საწვავს:

ა) მაზუთი ბინების გასათბობად;

ბ) სამრეწველო დანიშნულების ზეთი;

გ) დიზელის საწვავი ავტომობილებისათვის.

საწვავის თითოეული სახეობისათვის არსებობს ერთი ტუმბო, ხოლო მოთხოვნა საწვავის ყველა ტიპზე ერთნაირია. საწვავზე შეკვეთები თანაბარალბათურად ვარირებს 3000-5000 გალონამდე ზღვრებში 10 გალონი ნამატით. საწვავმზიდების გასავსებად საჭირო დრო წარმოადგენს შემდეგი ფაქტორების ფუნქციას:

1. გადაქაჩვის სიჩქარე (1000 გალონზე 6, 5 და 7 წუთი შესაბამისად);
2. შეკვეთის სიდიდე;
3. ბაზაში არსებული ავტომობილების რიცხვი (ერთ ავტომობილზე 30 დამატებითი წამი);
4. დაყენების დრო (ფიქსირებული დრო 2 წუთი)

ბაზას თავის ტერიტორიაზე შეუძლია განათავსოს მაქსიმუმ 12 ავტომობილი. სპეციალიზებული მოძრავი შემადგენლობის შემოსვლის საშუალო დრო ტოლია 18 წთ-ის და იცვლება შემდეგი ფუნქციის მიხედვით:

ბაზაში ავტომობილების შემოსვლის ტემპი:

სიხშირე	.20	.40	.25	.15
საშუალოკვადრატული გადახრა	.45	.60	1.5	2.0

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს ნავთობბაზაში ერთი სამუშაო დღის განმავლობაში:

ა) სპეციალიზებული ავტომობილების გასვლის (შემოსვლის) დროის განაწილებას;

ბ) ყოველდღიურად რეალიზებული საწვავის რაოდენობას.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                * * * * * * * * * * * * * * * * * *
*                *  CPSS WORLD SIMULATION  *
*                * * * * * * * * * * * * * * * * * *

************************************************************
*   OIL  STORAGE  AND  DISTRIBUTION DEPOT   *
************************************************************
RMULT 5631,39941 ; გენერატორის საწყისი მნიშვნელობების
                   განსაზღვრა

*         FUNCTION DEFINITIONS

ARR       FUNCTION       RN2,C5      ; შემოსვლის სიხშირე
0,0/0.2,.45/.6,1/.85,1.5/1.0,2

PUMPRATE  FUNCTION       P2,L3       ;1000 გალონის ჩატუმბვა
1,6/2,5/3,7

*        VARIABLE DEFINITIONS        ; ცვლადების განსაზღვრა

GALS      VARIABLE       RN1@201+300)#10
TYPE      VARIABLE       RN1@3+1
PUMP      VARIABLE
(FN$PUMPRATE#P1)/1000+S$DEPOT/2+2

*          STORAGE DEFINITIONS

DEPOT     STORAGE   12    ; მაქსიმუმ 12 ავტომობილის განთავსება
```

```
*           TABLE DEFINITIONS

 TRANSIT   TABLE   M1,10,10,20    ; სატრანზიტო დროის
                                    განსაზღვრის ცხრილი
 QTY TABLE  X1,20000,20000,9 ;შეკვეთების განსაზღვრის ცხრილი

*         MODEL SEGMENT 1

           GENERATE  18,FN$ARR ; ავტომობილების შემოსვლა
           ASSIGN  1,V$GALS  ; P1=გალონის ნომერს
           ASSIGN  2,V$TYPE  ; P2=საწვავის ტიპი
           ENTER   DEPOT     ; ავტომობილების მიერ ბაზის
                               დაკავება
           QUEUE   P2          ; საწვავის ტიპის მიხედვით  რიგში
                                 ჩადგომა
           SEIZE   P2          ; სატუმბი სადგურის დაკავება
           DEPART  52          ; რიგის დატოვება
           ADVANCE V$PUMP      ; ჩატუმბვის მომსახურების დრო
           RELEASE P2          ; სატუმბი სადგურის დატოვება
           LEAVE   DEPOT       ; ავტომობილების მიერ ბაზის
                                 განთავისუფლება
           SAVEVALUE  1+,P1    ; საწვავის შეძენის (გაყიდვის)
                                 ტალონის ნომერი
           TABULATE   TRANSIT    ; შეკვეთის დროის ტაბულირება
           TERMINATE             ; ბაზის დატოვება

*         MODEL SEGMENT 2

           GENERATE   480      ; მოდელირების დრო
           TABULATE   QTY      ; გალონების რეალიზაციის
                                 ტაბულირება
           SAVEVALUE  2+,X1    ; რეალიზებული საწვავის შენახვა
           SAVEVALUE  1,0      ; ტუმბოს მრიცხველის განულება
           TERMINATE   1         ; დასასრული
```

ა მ ო ც ა ნ ა 9
საწყობსა და ფილიალებში მარაგების მართვის სისტემის იმიტაციური მოდელირება

ქარხანა ამზადებს დანადგარებს, რომლებიც საჭიროა ნარჩენების საუტილიზაციოდ, რომელთაც ცალობაში ყიდის 200 აშშ დოლარად. საერთო ყოველწლიური მოთხოვნა შეადგენს 20000 დანადგარს. ქარხნის საწყობიდან განაწილება ხორციელდება სამი ფილიალის მეშვეობით. ქარხნის საწყობამდე წარმოებიდან ფილიალების მიერ შეკვეთის შესრულების დრო შეადგენს 4 კვირას. ქარხნის საწყობიდან ფილიალებში მიტანის (გატანის) დრო – 1 კვირას.

მარაგებით მართვის შემოთავაზებული მეთოდი – ეს არის პროდუქციის ეკონომიური ოდენობის ერთჯერადი შეკვეთების სისტემა. საწყისი მარაგები, შეკვეთის წერტილები, პროდუქციის ეკონომიური ოდენობები, ყოველკვირეული მოთხოვნა და სტანდარტული გადახრა ქარხნის საწყობისა და თითოეული ფილიალისათვის ნაჩვენებია ცხრილში.

ცხრილი 12.1

მარაგებით მართვის პარამეტრები

ადგილმდებარეობა	საწყისი მარაგი	შეკვეთის წერტილი	ეკონომიური ოდენობა	კვირეული მოთხოვნა	ყოველკვირეული სტანდარტული გადახრა
საწყობი	3400	2100	2300	–	–
ფილიალი 1	430	240	115	64	24
ფილიალი 2	600	430	165	128	32
ფილიალი 3	1000	630	200	192	48

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს მარაგებით მართვის სისტემას 5 კვირის განმავლობაში.

ა) განსაზღვრეთ მარაგების განაწილება სამ ფილიალსა და ქარხნის საწყობში;

ბ) მოახდინეთ ფაქტიური ყოველთვიური გაყიდვების განაწილების ტაბულირება;

გ) გამოთვალეთ მარაგების საშუალო მნიშვნელობა ფილიალებსა და ქარხნის საწყობში;

დ) როგორ აკმაყოფილებს აღნიშნული სისტემა ფირმის მიერ წარმოებულ მომსახურების პოლიტიკას, რომელიც უშვებს იმ ფაქტს, რომ რვა წლის განმავლობაში შესაძლებელია მარაგების მთლიანად ამოწურვა?

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                 * * * * * * * * * * * * * * * * * *
*                 *  CPSS WORLD SIMULATION  *
*                 * * * * * * * * * * * * * * * * * *
*********************************************************************
*   FACTORY WAREHOUSE AND DISTRIBUTORS INVENTORY   *
*********************************************************************
     RMULT 94521 ;გენერატორის საწყისი მნიშვნელობის განსაზღვრა

*              FUNCTION DEFINITIONS

 SNORM   FUNCTION  RN1,C25 ; ნორმალური განაწილების ფუნქცია
0,-5/0.00003,-4./.00135,-3/.00621,-2.5/.02275,-2./.06681,-1.5
.11507,-1.2/.15866,-1./.21186,-.8/.27425,-.6/.34458,-.4/.42074,-.2
.5,0/.57926,.2/.65542,.4/.72575,.6/.78814,.8/.84134,1/.88493,1.2
.93319,1.5/.97725,2/.99379,2.5/.99865,3/.99997,4/1,5

             INITAL   X1,3400  ; საწყობის საწყისი მარაგი
             INITAL   X2,2100  ; საწყობის შეკვეთის წერტილი
             INITAL   X3,2300  ;საწყობის ეკონომიური ოდენობა
             INITAL   X$STOCK1,430 ;საწყ. მარაგი I ფილიალისთვის
             INITAL   X$STOCK2,600  ; საწყ. მარაგი II ფილ-თვის
             INITAL   X$STOCK3,1000 ; საწყ. მარაგი III ფილ-თვის
             INITAL   X$EOQ1,115  ;ეკონ. შეკვეთა I ფილიალისთვის
             INITAL   X$EOQ2,165 ;ეკონ. შეკვეთა II ფილიალისთვის
             INITAL   X$EOQ3,200 ;ეკონ. შეკვეთა III ფილიალისთვის
             INITAL   X$POINT1,240 ; შეკვეთის წერტ. I ფილ-თვის
             INITAL   X$POINT2,430 ; შეკვეთის წერტ. II ფილ-თვის
             INITAL   X$POINT3,630 ;შეკვეთის წერტ. III ფილ-თვის
DEMAND1      VARIABLE FN$SNORM#24+64
DEMAND2      VARYABLE FN$SNORM#32+128
DEMAND3      VARYABLE FN$SNORM#48+192
TOTAL        VARYABLE P1+P2+P3
SALES        TABLE    X5,200,200,20
REGION_1     TABLE    X$STOCK1,0,40,20
REGION_2     TABLE    X$STOCK2,0,40,20
REGION_3     TABLE    X$STOCK3,0,40,20
FACTORY      TABLE    X,0,200,20
```

```
*                  MODEL SEGMENT 1

*            REORDERING BY FACTORY WAREHOUS

            GENERATE  ,,,1,2       ; შეკვეთის წერტილი
BACKHERE    TEST LE   X1,X2        ; გაყიდული პროდუქცია <=
                                     შეკვეთის წერტილზე?
            ADVANCE   4            ; შეკვეთის შესრულების დრო
            SAVEVALUE 1+,X3        ; შეკვეთის გაზრდა
            TRANSFER  ,BACKHERE    ; შეკვეთის გამეორება

*          MODEL SEGMENT 2

*            REORDERING AT EACH OF THE DISTRIBUTORS

            GENERATE  1,,,1                  ; პირველი განაწილება
DISTR1      TEST L    X$STOCK1,X$POINT1 ; მოთხოვნა < შეკვეთის
                                            წერტილზე?
            ADVANCE    1             ; შესრულების დრო = 1 კვირა
            SAVEVALUE 1-,X$EOQ1      ; I ფილიალის დანაკლისი
            SAVEVALUE STOCK1+,X$EOQ1  ; მოთხოვნის გაზრდა
            TRANSFER  ,DISTR1        ; გადასვლა ახალ მოთხოვნაზე
            GENERATE  1,,,1                ; მეორე განაწილება
DISTR2    TEST L    X$STOCK2,X$POINT2 ; მოთხოვნა < შეკვეთის
                                            წერტილზე?
            ADVANCE   1              ; შესრულების დრო = 1 კვირა
            SAVEVALUE 1-,X$EOQ2        ; II ფილიალის დანაკლისი
            SAVEVALUE STOCK2+,X$EOQ2  ; მოთხოვნის გაზრდა
            TRANSFER  ,DISTR2        ; გადასვლა ახალ მოთხოვნაზე
            GENERATE  1,,,1                ; მესამე განაწილება
DISTR3      TEST L    X$STOCK3,X$POINT3 ; მოთხოვნა < შეკვეთის
                                            წერტილზე?
            ADVANCE  1 ; შესრულების დრო = 1 კვირა
            SAVEVALUE 1-,X$EOQ3    ; II ფილიალის დანაკლისი
            SAVEVALUE STOCK3=,X$EOQ3 ; საწყისი მარაგი =
                                       ეკონომიური შეკვეთა
            TRANSFER  ,DISTR3        ; გადასვლა ახალ მოთხოვნაზე
```

```
*                MODEL SEGMENT 3

*          WEEKLY DEMAND AT EACH DISTRIBUTOR

          GENERATE    1,,,,3        ; პრიორიტეტი აქვს ყოველ-
                                      კვირეულ განაწილებას
          ASSIGN      1,V$DEMAND1 ; P1=I მოთხოვნის განაწილება
          ASSIGN      2,V$DEMAND2 ; P2=II მოთხოვნის განაწილება
          ASSIGN      3,V$DEMAND3 ;P3=III მოთხოვნის განაწილების
          SAVEVALUE   STOCK1-,P1  ; ყოველკვირეული მოთხოვნის
                                      განაწილება
          SAVEVALUE   STOCK2-,P2  ; მოთხოვნის განაწილება ყოველ
                                      2 კვირაში
          SAVEVALUE   STOCK3-,P3  ; მოთხოვნის განაწილება ყოველ
                                      3 კვირაში

          SAVEVALUE   5+,V$TOTAL ;მთლიანი მოთხოვნის განაწილება
          TABULATE    REGION_1    ; I მოთხოვნის განაწილების
                                      ტაბულირება
          TABULATE    REGION_2    ; II მოთხოვნის განაწილების
                                      ტაბულირება
          TABULATE    REGION_3    ; III მოთხოვნის განაწილების
                                      ტაბულირება
          TABULATE    FACTORY     ; ქარხნის პროდუქციის
                                      ტაბულირება
          TERMINATE                ; საწარმოო ციკლის დასასრული

*          MODEL SEGMENT 4
*         MONHLY RECORDING OF SALES

          GENERATE    4,,,,1    ; პრიორიტეტი აქვს ყოველთვიურ
                                  მოთხოვნას
          TABULATE    SALES     ; გაყიდვების ტაბულირება
          SAVEVALUE   5,0       ; შეკვეთის შესრულება
          TERMINATE   1         ; დასასრული
```

ა მ ო ც ა ნ ა 10

T-ებრ გზაჯვარედინზე საბზაო მოძრაობის იმიტაციური მოდელირება

T-ებრ გზაჯვარედინზე ავტომობილები შემოდის საშუალოდ 6, 28 წმ-ის ინტერვალში, რომელიც ექვემდებარება განაწილების ჰიპერექსპონენციალურ კანონს (სტანდარტული გადახრა 8, 40 წმ-ის ტოლია). ამის შემდგომ ავტომობილები ჩრდილოეთის მიმართულებით უხვევენ მარცხნივ და გადიან საავტომობილო ტრასაზე. როდესაც ისინი გადაკვეთენ სამხრეთის მიმართულებით მიმავალ ვიწრო გზას, ცენტრალურ რიგში ადგილი აქვს მოცდენას, რომლის ტევადობაც შეადგენს მაქსიმუმ 8 ავტომობილს. თითოეული ავტომობილი საჭიროებს 3, 6 წმ-ს (ერლანგის **K** = 4) იმისათვის, რომ გადაკვეთოს ვიწრო გზები საჭიროა 4 წმ (ერლანგის **K** = 5) და რომ შეუერთდეს ჩრდილოეთის მიმართულებით მიმავალ ნაკადს. სამხრეთის მიმართულებით მოძრავი ავტომობილები შემოდის ყოველი 50±5 წმ-ის ინტერვალში და საჭიროებს 15±5 წმ-ს, რომ გადაკვეთონ **T**-ებრი გზაჯვარედინი, ჩრდილოეთის მიმართულებით მოძრავი ავტომობილები კი – ყოველი 60±5 წმ-ის ინტერვალში და გზაჯვარედინის გასათავისუფლებლად მათ სჭირდებათ 15±5 წმ.

შეადგინეთ **GPSS** მოდელი, რომელიც 10 წთ-ის განმავლობაში ასახავს საგზაო მოძრაობის სურათს **T**-ებრ გზაჯვარედინზე, აგრეთვე:

ა) განსაზღვრეთ **T**-ებრ გზაჯვარედინზე ჩრდილოეთის მიმართულებით მომხვევი ავტომობილების მოძრაობის დრო;

ბ) წარმოდგინეთ ცხრილის სახით ვიწრო გზების გადაკვეთისა და ნაკადში შესვლის ფაქტიური დრო;

გ) იპოვეთ ვიწრო გზაზე რიგში მდგომი და მარცხნივ მოხვევის მომლოდინე (მარცხნივ მომხვევი) ავტომობილების მაქსიმალური რაოდენობა.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*                * * * * * * * * * * * * * * * * * *
*                *  CPSS WORLD SIMULATION  *
*                * * * * * * * * * * * * * * * * * *
*****************************************
*   TRAFFIC AT A T-JUNCTION   *
*****************************************

*   Erlang SERVISE TIMES ...HYPEREXPONENTIAL ARRIVALS

*   TIME UNIT IS 1/100 SECOND

* არით. ცვლადის განსაზღვრა ჰიპერექსპონენციალურ კანონით
HYPER FVARIABLE (410+((RN2'L'234)#(1343-410)))#FN$XPDIS
f(t)= .234(1/4.100 exp(-t/4.1)+ .766(1/13.43) exp(-t/13.43)

*         STORAGE    DEFINITIONS

 AISLE    STORAGE  8 ; ცენტრ. რიგში ავტომ-ების რაოდენობა

*         TABLE   DEFINITIONS

MARGETIME  TABLE    MP2,100,100,20
CROSSTIME  TABLE    MP1,100,100,20
TRANSIT    TABLE    M1,1000,1000,9
ARRIVALS   TABLE    V$HYPER,200,200,20
```

```
*      FUNCTION DEFINITIONS

XPDIS FUNCTION RN1,C24 ;ექსპონენციალური განაწილების ფუნქცია
 0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915
.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52
.94,2.81/.95,2.99/.96,3.2/.97,3.5/.98,3.9/.99,4.6
.995,5/.3.998,6.2/.999,7/.9998,8

*            MODEL SEGMENT 1

          GENERATE     V$HYPER,,300 ; ავტომობილების
            შემოსვლა ჰიპერექსპონენციალური კანონის თანახმად
          QUEUE        FIRST  ; რიგში ჩადგომა
          GATE SNF     AISLE  ; არის ადგილი გზაჯვარედინზე?
          SEIZE        SOUTZLANE ; სამხრეთის მიმართულებით
                                   გზის მონაკვეთის დაკავება
          DEPART       FIRST  ; რიგის დატოვება
          MARK         1      ; ტრანზაქტის სისტემაში
                                 შესვლის დრო

*            Erlang distribution

          ADVANCE      90,FE$XPDIS  ; ლოდინის დრო
          ADVANCE      90,FN$XPDIS  ; ლოდინის დრო
          ADVANCE      90,FN$XPDIS  ; ლოდინის დრო
          ADVANCE      90,FN$XPDIS  ; ლოდინის დრო
          TABULATE     GROSSE TIME  ; გზაჯვარედინზე დროის
                                      ტაბულირება
          ENTER        AISLE    ; ცენრ. რიგში ავტომ-ების მიერ
                                   გზის მონაკვეთის დაკავება
          RELEASE      SOUTHLANE ; სამ. მიმართულებით გზის
                                    მონაკვეთის დატოვება
          QUEUE        TWO      ; რიგში ჩადგომა
          SEIZE NORTHLANE ; ჩრდ. მიმ-ით გზის მონ-ის დაკავება
          DEPART       TWO      ; რიგის დატოვება
          LEAVE        AISLE  ; ცენტრ. რიგში ავტომ-ების მიერ
                                  გზის მონაკვეთის დატოვება
          MARK         2      ; ტრანზაქტის სისტემაში
                                 შესვლის დრო
```

```
*       Erlang distribution

        ADVANCE     80,FN$XPDIS     ; ლოდინის დრო
        ADVANCE     80,FN$XPDIS     ; ლოდინის დრო
        ADVANCE     80,FN$XPDIS     ; ლოდინის დრო
        ADVANCE     80,FN$XPDIS     ; ლოდინის დრო
        ADVANCE     80,FN$XPDIS     ; ლოდინის დრო
        RELEASE     NORTHLANE       ; ჩრდ. მიმ-ით გზის
                                      მონაკვეთის დატოვება
        TABULATE    MERGETIME      ; ჩრდ. მიმ-ით საერთო
                                      დროის ტაბულირება
        TABULATE    TRANSIT  ; სატრანზიტო დროის
                               ტაბულირება
        TABULATE    ARRIVALS ; ავტომობილების შემოსვლის
                               დროის ტაბულირება
        TERMINATE            ; გზაჯვარედინის დატოვება

*           MODEL SEGMENT 2

        GENERATE    5000,500,,,10  ; გზაჯვარედინზე სამხ.
              მიმ-ით ავტ-ების შემოსვლა 10 წთ-ის ინტერვალით
        SEIZE   SOUTHLANE ;სამხ. მიმ-ით გზის მონ-ის დაკ-ება
        ADVANCE 1200,300  ; გზაჯვარედინზე  მომ. დრო
        RELEASE SOUTLANE  ;სამხ. მიმ-ით გზის მონ-ის გან-ება
        TERMINATE         ; გზაჯვარედინის დატოვება

*           MODEL SEGMENT 3

        GENERATE   6000,500,,,10 ; გზაჯვარედინზე ჩრდ.
              მიმ-ით ავტ-ების შემოსვლა 10 წთ-ის ინტერვალით
        SEIZE   NORTHLANE ;ჩრდ. მიმ-ით გზის მონ-ის დაკ-ბა
        ADVANCE     1200,300  ; გზაჯვარედინზე მომ. დრო
        RELEASE NORTHLANE ;ჩრდ. მიმ-ით გზის მონ-ის გან-ება
        TERMINATE             ; გზაჯვარედინის დატოვება

*           MODEL SEGMENT 4

        GENERATE    6000          ; მოდელირების დრო
        TERMINATE   1             ; დასასრული
```

ა მ ო ც ა ნ ა 11

განსაზღვრული ტიპის საქონლისადმი მომხმარებლის დამოკიდებულების იმიტაციური მოდელირება

ჩატარებულ იქნა გამოკვლევა მომხმარებლის 7 სახის სარეცხი ფხვნილის არჩევის შეფასების თვალსაზრისით. ერთი სახის ფხვნილიდან მეორე სახის ფხვნილზე მომხმარებლის გადასვლის ალბათობა ნაჩვენებია მე-2 ცხრილში. ნავარაუდევია, რომ მომხმარებელი შეიძენს იმავე სახის ფხვნილს (ალბათობა გაცილებით უფრო დიდი გახლავთ), ვიდრე აირჩევს სხვა სახის ფხვნილს (ალბათობა მინიმუმამდეა დაყვანილი). ამიტომ მატრიცის თითოეულ სტრიქონში ალბათობათა ჯამი ტოლია. ბაზრის წილების განაწილების ალბათობა დამოკიდებულია გადასვლების რიცხვზე, რომელთაც ადგილი ჰქონდათ (ანუ თანამიმდევრობითი შესყიდვების რიცხვზე) თითოეული პროდუქტის ბაზრის საწყისი წილისაგან.

მოდელის მიერ შეფასებულ უნდა იქნას ბაზრის წილი, რომელსაც სავარაუდოდ ხანგრძლივი პერიოდის განმავლობაში აღწევს თითოეული პროდუქტი ანუ ის, რომ გადასვლის ალბათობათა მატრიცა სავარაუდოდ უცვლელი დარჩება და ამასთან იმის დაშვება, რომ თავდაპირველად ყველა პროდუქტს გააჩნია ბაზრის თანაბარი წილი.

ცვლილებათა თანამიმდევრობის ზემოაღნიშნული პროცესი შეიძლება წარმოდგენილ იქნეს მარკოვის პროცესით:

სტრიქონიდან სვეტში მომხმარებლის გადასვლის ალბათობა:

ცხრილი 12.2

სოდის შემცველი	შერეული	ბურთულებიანი	თვალის-მომჭრელი სისუფთავე	საუცხოო რეცვა	სწრაფი რეცხვა	საპნის ეფექტი	სულ
.35	.12	.17	.15	.10	.04	.05	1.0
.13	.32	.10	.15	.12	.09	.09	1.0
.15	.14	.25	.14	.17	.08	.07	1.0
.11	.10	0.9	.40	.08	.09	.13	1.0
.05	.12	.16.	.09	.37	.14	.07	1.0
.16	.13	.08	.05	.16	.28	.14	1.0
.08	.10	.09	.10	.07	.13	.43	1.0

ვინაიდან გადასვლის ალბათობათა ცხრილს არ გააჩნია ნულები, ამასთან დაკავშირებით ჩვენ ვახდენთ იმის კონსტატაციას, რომ ყველა მდგომარეობა იმყოფება ერთ ჯაჭვში და წარმოადგენს პერიოდულს. აღნიშნული კი მარკოვის თეორიულ პროცესში უზრუნველყოფს მდგომარეობათა ალბათობების თანხვედრას.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს ბაზარზე თითოეული პროდუქტის რეალიზაციის წილს 5 სამუშაო დღის განმავლობაში.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*              * * * * * * * * * * * * * * * * * *
*              *  CPSS WORLD SIMULATION  *
*              * * * * * * * * * * * * * * * * * *

*****************************************
*   BRAND LOYALTY BY CUSTOMERS    *
*****************************************
RMULT 98851 ; გენერატორის საწყისი მნიშვნელობის განსაზღვრა
CHECK TABLE X$BRAND,1,1,8 ; ბრენდის ცხრილის განსაზღვრა
TRANTIONS MATRIX  ,7,7     ; მატრიცის განსაზღვრა

*         FUNCTION    DEFINITIONS
 POWDER FUNCTION X$BRAND,M7 ; ფხვნილის შერჩევის ფუნქცია
1,FN$SUDS/2,FN$BUBBLES/4,FN#CLEARSHINE
5,FN$CLEANPLUS/6,FN#MARVEL/7,FN#SOAPY

  SUDS   FUNCTION RN1,07      ; სოდის შემცველი
0,390,1/,510,2/,680,3/,810,4/,910,5/,960,6/,1,0,7

  BRANDX FUNCTION RN1,07      ; შერეული
0,1301/,450,2/,550,3/,700,4/,820,5/,910,6/,1,0,7

  BUBBLES FUNCTION RN1,07     ; ბურთულებიანი
0,150,1/,290,2/,540,3/,680,4/,850,5/,930,6/,1,0,7

  CLEARSHINE FUNCTION RN1,07 ; თვალისმომჭრელი სისუფთავე
0,110,1/,210,2/,300,3/,700,4/,780,5/,870,6/,1,0,7

   CLEANPLUS FUNCTION RN1,07 ; საუცხოო რეცვა
0,050,1/,170,2/,330,3/,420,4/,790,5/,930,6/,1,0,7

   MARVEL FUNCTION    RN1,07 ; სწრაფი რეცხვა
0,160,1/,290,2/,370,3/,420,4/,580,5/,860,6/,1,0,7

   SOAPY FUNCTION    RN1,07  ; საპნის ეფექტი
0,080,1/,180,2/,270,3/,370,4/,440,5/,570,6/,1,0,7

   RECORD FUNCTION   X$BRAND,L7   ; ჩანაწერების ნომრები
1,SUDS/2,BRANDX/3,BUBBLES/4,CLEARSHINE/5,CLEANPLUS/6,
MARVEL/7,SOAPY
```

```
*          MODEL SEGMENT 1

           INITIAL     X$BRAND,1; ბრენდის ინიციალიზაცია
           GENERATE    1,,,,2     ; ფხვნილის მოსინჯვა
           TABULATE    CHECK      ; ტაბულირება
           SAVEVALUE   OLDBRAND,X$BRAND    ; ძველი ფხვნილის
                                           სახელის დამახსოვრება
           SAVEVALUE   BRAND,FN$POWDER ;ახალი ფხვნილის ძებნა
           SAVEVALUE   FN$RECORD+,1 ;შერჩეული სახის ფხვნილის
                                      ჩანაწერის გაზრდა
           MSAVEVALUE TRANTIONS+,X$OLDBRAND,X$BRAND,1
                 ; ფხვნილის განახლება ახალი სახის ფხვნილით
           TERMINATE  1           ; დასასრული
```

ა მ ო ც ა ნ ა 12
სამსხმელო საამქროს იმიტაციური მოდელირება

სამსხმელო საამქროში მუშაობს 18 ჩამომსხმელი (ოსტატი). ისინი ასრულებენ შემოსულ შეკვეთებს, რომლებიც საშუალოდ ყოველი მომდევნო საათის ინტერვალში შემოდის და ექვემდებარება შემთხვევითი სიდიდეების განაწილების ექსპონენციალურ კანონს. სამსხმელო საამქროს გააჩნია 8 საათიანი სამუშაო დღე 5 დღიანი სამუშაო კვირით. შეკვეთების 30% წარმოადგენს ახალ შეკვეთებს, ხოლო 70% კი – განმეორებადს. ახალი შეკვეთები მოითხოვს ფორმას, რომელიც მზადდება ფორმის მიმცემი ყალიბების სამსხმელო საამქროში და დასამზადებლად სჭირდება თანაბარალბათური კანონის თანახმად განაწილებული 72±24 სთ დრო. განმეორებადი შეკვეთების შემოსვლა განაწილებულია თანაბარალბათურად და ვარირებს 5±3 სთ დროის ინტერვალში.

ზომების მიხედვით შეკვეთები განაწილებულია თანაბარალბათურად და ვარირებს 6±24 დეტალი. ცალკეული დეტალების წონა იცვალება შემდეგი ცხრილის შესაბამისად.

განაწილება წონის მიხედვით:

სიხშირე	.05	.08	.12	.25	.20	.15	.10	.05
დეტალის წონა	3	6	11	20	28	35	42	50

ჩამოსხმის დრო დეტალის წონის კილოგრამზე შეადგენს 2 წუთს. შეკვეთების შესრულების ვადა განისაზღვრება ჩამოსხმის საერთო დროის მიხედვით პლუს ცვალებადი ტექნოლოგიური დრო, რომელიც თანაბარალბათური განაწილების კანონის დროს ტიპურ შემთხვევაში იცვლება 40±160 სთ-მდე. ოსტატი იცდის ვიდრე მზად არ იქნება შეკვეთის ფორმა. შემდეგ იგი შეკვეთის ვადის შესაბამის დროში ასრულებს ერთ დეტალს. ერთი ჩამომსხმელი მთელ შეკვეთას ბოლომდე ასრულებს. ჩამოსხმა წარმოებს დღეში ერთხელ სამუშაოს დამთავრებამდე ერთი საათით ადრე. როდესაც იწყება ზემოხსენებული პროცესი ყველა ჩამომსხმელი წყვეტს მუშაობას ფორმებთან და უერთდება ჩამოსხმის პროცესს.

შეადგინეთ **GPSS** მოდელი, რომელიც მოახდენს სამუშაოთა იმიტაციას სამსხმელო საამქროში 10 სამუშაო დღის განმავლობაში. განსაზღვრეთ ყველა ჩასატარებელ სამუშაოთა დროის განაწილება.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*              * * * * * * * * * * * * * * * * * *
*              *  CPSS WORLD SIMULATION  *
*              * * * * * * * * * * * * * * * * * *

****************************************
*   FOUNDRY SIMULATION MODEL   *
****************************************

RMULT 85327 ; გენერატორის საწყისი მნიშვნელობის განსაზღვრა

*   TIME UNIT IS ONE MINUTE

P1 = TYPE OF JOB                    ; სამუშაოს ტიპი
P2 = NUMBER IN THE ORDER            ; შეკვეთის ნომერი
P3 = WEIGHT OF A COMPONENT          ; კომპონენტის წონა
P4 = MOLDING TIME PER COMPONENT ; ჩამოსხმის დრო
 P5 = DUE DATE                      ; განმეორებით შეკვეთის დრო
P6 = TOTAL WEIGHT PER ORDER ; დეტალის მთლიანი წონა
P7 = INDEX FOR LOOPING              ; ციკლის მაჩვენებელი


*    FUNCTION    DEFINITIONS

XPDIS FUNCTION RN1,C24 ;ექსპონენციალური განაწილების ფუნქცია

 0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915
.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52
.94,2.81/.95,2.99/.96,3.2/.97,3.5/.98,3.9/.99,4.6
.995,5/.3.998,6.2/.999,7/.9998,8

WEIGHT  FUNCTION RN1,C8 ; კომპონენტების წონა კგ-ში
0.0,3/.13,6/.25,11/.50,20/.70,28/.85,35/.95,42/1.0,50

ORDERTYPE  FUNCTION RN1,D2  ; ახალი შეკვეთა P1=1;
0.3,1/1,0,2                   განმეორებითი P1=2

*        VARIABLE   DEFINITIONS

SIZE     VARIABLE   RN1@19+6       ; შეკვეთის ზომა
```

```
DDATE     VARIABLE   V$MTIME#P2+RN1@121+40+C1
                                   ;განმეორებითი შეკვეთის დრო
MTIME     VARIABLE   (P3#2)          ; ჩამოსხმის დრო
DAY       VARIABLE   (C1/480)        ; დღიური ინდიკატორი
TOTAL     VARIABLE    P3#P2          ; შეკვეთის წონა

*         TABLE DEFINITIONS

TIMES     TABLE    M1,400,400,20 ;შეკვეთის შესრულების დრო
CAST      TABLE    X$WTMOLD,400,400,20 ; შეკვეთის წონა

*              STORAGE DEFINITIONS

 MOLDERS       STORAGE    18    ; ჩამომსხმელების რაოდენობა

*              MODEL SEGMENT 1

            GENERATE 60,FN$XPDIS,,,3 ; შეკვეთების შემოსვლა
            ASSIGN   1,FN$ORDERTYPE  ; სამუშაოს ტიპი
            TEST E   P1,2,NEWJOB       ; მეორდება შეკვეთა?
            ADVANCE  300,180     ; შეკვეთის შესრულების დრო
COMMENCE    ASSIGN   2,V$SIZE    ; შეკვეთის ზომა
            ASSIGN   3,FN$WEIGHT ; კომპონენტების წონა
            ASSIGN   4,V$MTIME   ; კომპონენტების მიხედვით
                                    ჩამოსხმის დრო
            ASSIGN   5,VSDDATE   ;განმეორებითი შეკვეთის დრო
            ASSIGN   6,V$TOTAL   ; დეტალების მთლიანი წონა
            GATE SNF MOLDERS,WAIT ; რამდენი ოსტატია
                                     თავისუფალი
BEGIN       ENTER    MOLDERS     ; ოსტატების დასაქმება
            ASSIGN   7,P2        ; P7=შეკვეთის ნომერი
NEXT        ADVANCE  P4          ; ჩამოსხმის დრო
                                   კომპონენტების მიხედვით
            LOOP     7,NEXT      ; გაიმეორეთ ყოველი
                                   კომპონენტისთვის
            LEAVE    MOLDERS     ; ოსტატების განთავისუფლება
            SAVEVALUE WTMOLD+,P6 ; თით. ჩამოსხმული
                                     შეკვეთის ჯამური წონა
```

```
          UNLINK    1,BEGIN,1  ; შემდეგი შეკვეთის
                                 შესრულებზე გადასვლა
          TABULATE  TIMES      ; შეკვეთის შესრულების
                                 დროის ტაბულირება
          TERMINATE            ; შეკვეთის დასრულება
NEWJOB    ADVANCE   4320,1440  ; ახალი შეკვეთის
                                 შესრულების დრო
          TRANSFER  ,COMMENCE  ; ახალ შეკვეთაზე გადასვლა
WAIT      LINK      1,P5       ; შეკვეთების ლოდინის
                                 ჯაჭვში ჩართვა

*            MODEL SEGMENT 2

          GENERATE  420,,,1,2  ; დეტალების ჩამოსხმის
                                 ციკლის დასაწყისი
AGAIN     SUNAVAIL  MOLDERS    ; დეტალების ჩამოსხმის
                                 ციკლის დონეები
          ADVANCE   60         ; ჩამოსხმის დრო
          SAVAIL    MOLDERS    ; ჩამომსხმელების
                                 განთავისუფლება
          ADVANCE   420        ; ჩამოსხმის დრო
          TABULATE  CAST       ; ჩამოსხმის საერთო
                                 დროის ტაბულირება
          SAVEVALUE TOTCAST+,X$WTMOLD ; ჩამოსხმულის
                                        შენახვა
          SAVEVALUE WTMOLD,0   ; შენახულის განულება
          TRANSFER  ,AGAIN     ; ახალ შეკვეთაზე გადასვლა

*            MODEL SEGMENT 3

          GENERATE  4800,,,,4  ; მოდელირების დრო
          SAVEVALUE V$DAY,X$TOTCAST ; ჩამოსხმის მთლიანი
                                      წონის ტაბულირება
          TERMINATE 1                ; დასასრული
```

ა მ ო ც ა ნ ა 13
ლითონსაჭრელი ჩარხების იმიტაციური მოდელირება

კომპიუტერული ფირმა ლითონსაჭრელი ჩარხებისათვის ამზადებს ლენტებს სპეციალური პროგრამული უზრუნველყოფით. დეტალების ნახაზები ფირმას მიეწოდება დამამზადებლების მიერ. პროგრამისტი ახდენს ნახაზების შესწავლას და წერს დეტალის დამუშავების პროგრამას ნამზადის დამუშავების დროს ჩარხის სამართავად. დაპროგრამება იკავებს შემთხვევითი სიდიდეების განაწილების ექსპონენციალური კანონით განაწილებულ 90 წუთს. შემდეგ პროგრამა შეაქვთ კომპიუტერში, წარმოებს მისი დამუშავება და ჩარხის ლენტზე ინფორმაციის ჩაწერა. აღნიშნული პროცესი იკავებს ექსპონენციალური კანონის მიხედვით განაწილებულ 60 წუთს. ზემოაღნიშნული პროცესის გავლის შემდეგ ლენტი გამოცდისა და რედაქტირებისათვის იტვირთება შესაბამის ჩარხში. ჩატვირთვის პროცესი იკავებს ექსპონენციალურად განაწილებულ 70 წუთს.

კომპიუტერული ფირმის მიზანი მდგომარეობს იმაში, რომ უზრუნველყოს მოცემული დარგის სწრაფი და საიმედო მომსახურება. კლიენტების საუკეთესო მომსახურების თვალსაზრისით ფირმის მიერ შემოთავაზებულია განსახორციელებელ სამუშაოთა დამუშავების სამი შესაძლო ვარიანტი:

ა) თავდაპირველად დამუშავდეს იმ ჩასატარებელ სამუშაოთა ნუსხა, რომელთაც გააჩნიათ დამუშავების ყველაზე მცირე საერთო დრო;

ბ) დამუშავებულ იქნას ის სამუშაოები, რომელთაც გააჩნიათ დამუშავების ყველაზე დიდი საერთო დრო;

გ) თავდაპირველად დამუშავებულ იქნას ის ჩასატარებელი სამუშაოები, რომელთაც გააჩნიათ შესრულების ყველაზე მოკლე ვადა, ანუ ის ღონისძიებები რომლებიც ყველაზე მეტად ექვემდებარება მოთხოვნას.

სამუშაოთა შესასრულებლად საჭირო ტექნოლოგიური დრო არის დამუშავების საერთო დროს (**P4 = P1 + P2 + P3**) პლუს 3±1 დღე. ვადა – არის საათების მიხედვით განსაზღვრული ფარდობითი დრო, როდესაც სრულდება სამუშაოს პლუს დასაშვები ტექნოლოგიური დრო. დაყოვნება განისაზღვრება, როგორც საათობრივად განსაზღვრული ფარდობითი დრო, როდესაც სამუშაო დასრულებულია მინუს აღნიშნულისათვის დადგენილი ვადა.

შეადგინეთ **GPSS** მოდელი, რომლის საშუალებითაც განხორციელდება კომპიუტერული ფირმის მიერ 100 შეკვეთის დამუშავება. აგრეთვე:

1) დამუშავების უმოკლესი დროის მიხედვით გამოთვალეთ რიგის მდგომარეობა.
2) იპოვეთ რესურსის გამოყენების კოეფიციენტი ხელით ჩაწერის, ლენტის მომზადებისა და რედაქტირების დროს.
3) შეადგინეთ ცხრილი, რომელიც ასახავს შეკვეთების დაყოვნებისა და მათი შესრულების დროს.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*              * * * * * * * * * * * * * * * * * *
*              *  CPSS WORLD SIMULATION  *
*              * * * * * * * * * * * * * * * * * *

*************************************************************
*        COMPUTER  NUMERICAL  CONTROL                      *
*   TAPE PROGRAMMING, LOADING AND EDITING                   *
*************************************************************

RMULT 66753 ; გენერატორის საწყისი მნიშვნელობის განსაზღვრა

*      FUNCTION    DEFINITIONS

XPDIS FUNCTION RN1,C24 ;ექსპონენციალური განაწილების ფუნქცია

 0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915
.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52
.94,2.81/.95,2.99/.96,3.2/.97,3.5/.98,3.9/.99,4.6
.995,5/.3.998,6.2/.999,7/.9998,8

* P1 = PROCESS TIME FOR PROGRAMMING ;დაპროგრამების დრო
* P2 =    "     "    "  PUNCHING ; პროგრამის ჩარხის
                                    ლენტზე ჩაწერის დრო
* P3 =    "     "    "  EDITING  ;პროგრამის ჩატვირთვისა
                                     და რედაქტირების დრო
* P4 = SHDRTEST PROCESSING TIME ; დამუშავების უმოკლესი
                                   საერთო დრო
                                   (ტექნოლოგიური დრო)
* P5 = DUE DATE                 ; ვადა (ფარდობითი დრო)
* P6 = LONGEST PROCESSING TIME  ; დამუშავების უგრძელესი
                                  საერთო დრო

SCHEDPARM EQU          4        ; სამუშაოთა რაოდენობა

*           VARIABLE DEFINITIONS ; ცვლადების განსაზღვრა

  VAR1       VARIABLE   P1+P2+P3
  VAR2       VARIABLE   P4+AC1+RN1@160+80
  VAR3       VARIABLE   10000-P4
  LATENESS   VARIABLE   AC1-P5
```

```
*           TABLE    DEFINITIONS

  TRANSIT  TABLE   M1,100,100,20 ; შეკვეთის შესრულების
                                    დროის განსაღვრა
  LATE  TABLE   V$LATENESS,-1000,200,20  ; შეკვეთის
                                დაყოვნების დროის განსაზღვრა

*            MODEL SEGMENT 1

          GENERATE  120,FN$XPDIS  ; ახალი ლენტის შექმნა
          ASSIGN    1,90,XPDIS    ; დაპროგრამების დრო
          ASSIGN    2,60,XPDIS    ; პროგრამის ჩარხის
                                      ლენტზე ჩაწერის დრო
          ASSIGN    3,70,XPDIS    ; ლენტის ჩატვირთვისა და
                                     რედაქტირების დრო
          ASSIGN    4,V$VAR1  ; დამუშავების საერთო დრო
          ASSIGN    5,V$VAR2  ; P5 = ფარდობითი დრო
          ASSIGN    6,V$VAR3  ; უგრძელესი საერთო დრო
          LINK      PROGRAM,P$SCHEDPARM,WRITE
                      ; მომხმარებლის საერთო ჯაჭვში ჩართვა
WRITE     SEIZE     MANUSCRIPT ; ლენტის დაკავება
          ADVANCE   P1         ; დაპროგრამების დრო
          RELEASE   MANUSCRIPT ; ლენტის განთავისუფლება
          UNLINK    PROGRAM,WRITE,1 ; მომხმარებლის
                                  საერთო ჯაჭვიდან გამოსვლა
          LINK      TAPE,P$SCHEDPARM,PUNCH          ;
                      მომხმარებლის საერთო ჯაჭვში ჩართვა
PUNCH     SEIZE     TAPEPUNCH ; ჩარხის დაკავება
          ADVANCE   P2        ; ჩარხის მუშაობის დრო
          RELEASE   TAPEPUNCH ; ჩარხის განთავისუფლება
          UNLINK    TAPE,PUNCH,1 ; მომხმარებლის საერთო
                                    ჯაჭვიდან გამოსვლა
            LINK    EDIT,P$SCHEDPARM,LOADEDIT          ;
                      მომხმარებლის საერთო ჯაჭვში ჩართვა
 OADEDIT  SEIZE    EDIT ; ჩარხის დაკავება
          ADVANCE  P3; ჩატვირთვისა და რედაქტირების დრო
          RELEASE  EDIT ; ჩარხის განთავისუფლება
          UNLINK     EDIT,LOADEDDIT,1 ; მომხმარებლის
                                 საერთო ჯაჭვიდან გამოსვლა
```

```
*                MODEL SEGMENT 2

               TABULATE   LATE     ; შეკვეთების დაყოვნების
                                     დროის ტაბულირება
               TABULATE   TRANSIT  ; შეკვეთების შესრულების
                                     დროის ტაბულირება
               TERMINATE    1       ; დასასრული
```

ა მ ო ც ა ნ ა 14
რიგების ქსელის იმიტაციური მოდელირება

ერთ არხში მომლოდინეთა შემოსვლები ყოველი 5 წმ-ის ინტერვალით განაწილებულია ექსპონენციალური კანონის თანახმად და ტოლია 500 ერთეულის. მომსახურების საშუალო დრო უდრის 3 წმ-ს (დროის 300 ერთეული). მომსახურე არხის გამოყენების საშუალო კოეფიციენტმა შესაბამისად 60% უნდა შეადგინოს.

მომსახურების დროის თანახმად გამოკვლევას ექვემდებარება სამი რეჟიმი:

1. მომსახურების მუდმივი დრო;
2. განაწილების ექსპონენციალურ კანონს დაქვემდებარებული მომსახურების დრო;
3. ერლანგის კანონს დაქვემდებარებული მომსახურების დრო (**K** = 2).

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს 10 წთ-ის განმავლობაში რიგების სტატისტიკას მომსახურების თითოეული ტიპისათვის.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*              * * * * * * * * * * * * * * * * * *
*              *  CPSS WORLD SIMULATION  *
*              * * * * * * * * * * * * * * * * * *

*****************************************
*  QUEUEING THEORY VERIFICATION   *
*****************************************
*   TIME UNIT IS 1/100 OF A SECOND

*              FUNCTION    DEFINITIONS

XPDIS FUNCTION RN1,C24 ;ექსპონენციალური განაწილების ფუნქცია

 0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915
.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52
.94,2.81/.95,2.99/.96,3.2/.97,3.5/.98,3.9/.99,4.6
.995,5/.3.998,6.2/.999,7/.9998,8

*              TABLE    DEFINITIONS

  TRANSIT    TABLE    M1,250,250,20 ; შეკვეთის შესრულების
                                      დრო
  NUMBER     TABLE    Q$EXPON,0,1,20; ცხრილი რიგების
                                       შესახებ
  QCONSTENT QTABLE   CONSTANT,200,200,20; მომსახურების
                                          მუდმივი დრო
  QEXPON     QTABLE   EXPON,200,200,20; ექსპ. კანონით
                        განსაზღვრული მომსახურების დრო
  QERLANG    QTABLE   ERLANG,200,200,20; ერლანგის
                        კანონით განსაზღვრული
                        მომსახურების დრო

*        MODEL SEGMENT 1

  GENERATE   500,FN$XPDIS ;შეკვეთების შემოსვლა 5 წმ-ის
                           ინტერვალით
  QUEUE      CONSTANT     ; რიგში ჩადგომა
  SEIZE      FACILITY1    ; I მოწყობილობის დაკავება
  ADVANCE    300          ; მომსახურების დრო 3 წმ.
```

```
  RELEASE    FACILITY1 ; I მოწყობილობის განთავისუფლება
  DEPAPT     CONSTANT  ; რიგის დატოვება
  TERMINATE            ; რიგიდან წასვლა

*        MODEL SEGMENT 2

  GENERATE   500,FN$    ; შეკვეთების შემოსვლა 5 წმ-ის
                          ინტერვალით
  QUEUE      EXPON      ; რიგში ჩადგომა
  SEIZE      FACILITY2  ; II მოწყობილობის დაკავება
  ADVANCE    300        ; მომსახურების დრო 3 წმ.
  RELEASE    FACILITY2  ; II მოწყობილობის განთავისუფლება
  DEPART     EXPON      ; რიგის დატოვება
  TABULATE   TRANSIT    ; შეკვეთის შესრულების დროის
                          ტაბულირება
  TERMINATE             ; რიგიდან წასვლა

*        MODEL SEGMENT 3

  GENERATE   500,FN$XPDIS ; შეკვეთების შემოსვლა 5 წმ-ის
                            ინტერვალით
  QUEUE      ERLANG       ; რიგში ჩადგომა
  SEIZE      FACILITY3    ; III მოწყობილობის დაკავება
  ADVANCE    150,FN$XPDIS ; მომსახურების დრო ერლანგის
                            მიხედვით
  ADVANCE    150,FN$XPDIS ; მომსახურების დრო ერლანგის
                            მიხედვით
  RELEASE    FACILITY3  ; III მოწყობილობის განთავისუფლება
  DEPART     ERLANG     ; რიგის დატოვება
  TERMINATE             ; რიგიდან წასვლა

*             MODEL SEGMENT 4

   GENERATE     500,FN$XPDIS ; გენერირების დრო
   TABULATE     NUMBER       ; რიგის შესახებ სტატისტიკის
                               ტაბულირება
   TERMINATE    1            ; დასასრული
```

ა მ ო ც ა ნ ა 15
სატელეფონო სადგურის იმიტაციური მოდელირება

კერძო საქალაქთაშორისო ავტომატურ სატელეფონო სადგურს (ასს) გააჩნია 200 სააბონენტო პუნქტი, სიგნალიზაციის 9 მოწყობილობა და ერთი ოპერატორი. სატელეფონო ზარები შემოდის საშუალოდ 150 წმ-ის ინტერვალით, რომლებიც ემორჩილება შემთხვევითი სიდიდეების განაწილების ნორმალურ კანონს სტანდარტული გადახრით 30 წმ. გარედან განხორციელებული ზარების შემოსვლებს (გამოძახებებს) შორის დრო სააბონენტო პუნქტების რაოდენობის უკუპროპორციულად ვარირებს (2500/სააბონენტო პუნქტების რიცხვი) და ემორჩილება განაწილების ექსპონენციალურ კანონს. ქსელის შიგნით განხორციელებული (შიდა) ზარების შემოსვლებს შორის დრო კი - თავისუფალი სააბონენტო პუნქტების რაოდენობის (1260/ თავისუფალი სააბონენტო პუნქტების რიცხვი +1) უკუპროპორციულად. აღნიშნული გამოძახებებიდან აბონენტის ადგილსამყოფელი შესაძლოა იყოს (66, 6%) შიდა ან გარე (33, 3). შიდა ზარების განსახორციელებლად ოპერატორი საჭირო არ არის. შიდა ზარებისათვის საჭიროა სასიგნალო მოწყობილობა და შიდა ხაზი, ხოლო გარე ზარებისათვის – გარეშე ხაზი. სააბონენტო პუნქტების 15% დაკავებულია მათთან დაკავშირების მცდელობით, ხოლო 20% არ პასუხობს. სიგნალიზაციაზე საჭირო დრო ტოლია 7±2 წმ, ხოლო სააბონენტო პუნქტის გავლით ზარის განსახორციელებლად 6±2 წმ. სიგნალის “დაკავებულია” მიღების დროს

აბონენტი იცდის 4±1 წმ. ოპერატორის მომსახურება იკავებს 9±3 წმ-ს.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს კერძო საქალაქთაშორისო ასს-ს მუშაობას 1 სთ-ის განმავლობაში. აგრეთვე:

1. განსაზღვრეთ ოპერატორის, სასიგნალო მოწყობილობის, შიდა და გარე ხაზებისა და სააბონენტო პუნქტების გამოყენების კოეფიციენტი.
2. გააკეთეთ ყოველ წუთს დამუშავებული შიდა და გარე ზარების რიცხვიდან ამონარჩევი.

მოდელის პროგრამას გააჩნია შემდეგი სახე:

```
*               * * * * * * * * * * * * * * * * * *
*               *  CPSS WORLD SIMULATION  *
*               * * * * * * * * * * * * * * * * * *

*********************************************
*    PABX TELEPHONE SYSTEM MODEL    *
*********************************************

*      FUNCTION    DEFINITIONS

XPDIS FUNCTION RN1,C24 ;ექსპონენციალური განაწილების ფუნქცია
0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915
.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52
.94,2.81/.95,2.99/.96,3.2/.97,3.5/.98,3.9/.99,4.6
.995,5/.3.998,6.2/.999,7/.9998,8

SNORM FUNCTION RN1,C25   ;ნორმალური განაწილების ფუნქცია
0,-5/0.00003,-4./.00135,-3/.00621,-2.5/.02275,-2.
.06681,-1.5/.11507,-1.2/.15866,-1./.21186,-.8/.27425,-.6
.  .34458,-.4/.42074,-.2/.5,0/.57926,.2/.65542,.4
.72575,.6/.78814,.8/.84134,1/.88493,1.2/.93319,1.5
.97725,2/.99379,2.5/.99865,3/.99997,4/1,5
```

```
*          TABLE    DEFINITIONS

TRANSIT    TABLE  M1,20,20,20
CALLSINT   TABLE  S$INTLINES,2,2,20 ; შიდა ზარის
                                      განსაზღვრა
CALLSEXT   TABLE  S$EXTLINES,2,2,20 ; გარე ზარის
                                      განსაზღვრა

*          STORAGE DEFINITIONS ; ტევადობების განსაზღვრა

 EXTENSIONS STORAGE   200
 EXTLINES   STORAGE   30
 INTLINES   STORAGE   30
 SIGNALS    STORAGE   9
 OPERATOR   STORAGE   1

*         VARIABLES DEFINITIONS ; ცვლადების განსაზღვრა

INTERNAL  VARIABLE   1260/(1+R$EXTENSIONS)
EXTERNAL  VARIABLE   2500/(R$EXTENSIONS+S$EXTENSIONS)
CALLTIME  VARIABLE   (FN$SNORM#30)+150

*           MODEL SEGMENT 1

*  GENERATE CALLS ORIGINATING INTERNALLY

   GENERATE V$INTERNAL,FN$XPDIS,200 ; შემომავალი ზარები
   ENTER    EXTENSIONS ; სააბონენტო პუნქტის დაკავება
   QUEUE    INSIDE     ; შიდა ზარის რიგში ჩადგომა
   ENTER    SIGNALS    ; სასიგნალო მოწყ-ის დაკავება
   DEPART   INSIDE     ; შიდა ზარის რიგის დატოვება
   ADVANCE  7,2        ; სიგნალიზაციის დრო
   LEAVE    SIGNALS    ; სასიგნალო მოწყობილობის
                                განთავისუფლება
  TRANSFER .333,,UNIT ; განხორციელდება გარე ზარების 33%
```

```
INTINT   TEST GE   R$INTLINES,1,BREAKEOFF ; შიდა ხაზი
                                            დაკავებულია?
         ENTER    INTLINES    ; შიდა ხაზის დაკავება
         ADVANCE  4,1         ; აბონენტის მომსახურების დრო
         TRANSFER .15,,BUSY   ; სააბონენტო პუნქტების 15%
                                დაკავებულია
ALINE    ENTER    EXTENSIONS  ; სააბონენტო პუნქტების
                                დაკავება
         ADVANCE  6,2         ; ზარის განხორციელების დრო
         TRANSFER .2,,NOGOOD  ; ზარების 20% არ პასუხობს
         ADVANCE  V$CALLTIME  ; მომსახურების დრო
NOGOOD   LEAVE    EXTENSIONS  ; სააბონენტო პუნქტების
                                განთავისუფლება
BUSY     LEAVE    INLINES     ; შიდა ხაზის განთავისუფლება
         TRANSFER ,BREAKEOFF  ; კვლავ შიდა ხაზის
                                მოსინჯვა

*  MODEL INTERNAL TO EXTERNAL CALLS

INTOUT   TEST GE   R$EXTLINES,1,BREAKOFF ; გარე ხაზი
                                           დაკავებულია?
         ENTER    EXTLINES     ; გარე ხაზის დაკავება
         ADVANCE  4,1          ; დაკავების დრო
         TRANSFER .200,,NOBODY ; გარე ხაზის 20%
                                 დაკავებულია
         ADVANCE  6,2          ; პასუხის დრო
         TRANSFER .200,,NOBODY ; ზარების 20% არ
                                 პასუხობს
         ADVANCE  V$CALLTIME   ; მომსახურების დრო
         TABULATE TRANSIT      ; სატრანზიტო დროის
                                 ტაბულირება
NOBODY   LEAVE    EXTLINES     ; გარე ხაზის
                                 განთავისუფლება
BREAKOFF LEAVE    EXTENSIONS   ; სააბონენტო პუნქტის
                                 განთავისუფლება
         TERMINATE              ; სააბონენტო პუნქტის
                                  დატოვება
```

```
*                MODEL SEGMENT 2

*     PROCESS CALLS ORIGINATING EXTERNALLY

            GENERATE  V$EXTERNAL,FN$XPDIS ; გამავალი
                                                ზარები
   TEST GE R$EXTLINES,1,NONEFREE  ; გარე ხაზი
                                          თავისუფალია?
            ENTER     EXTLINES    ; გარე ხაზის დაკავება
            QUEUE     OUTSIDER    ; რიგში ჩადგომა
            ENTER     OPERATOR    ; ოპერატორის დაკავება
            DEPART    OUTSIDER    ; რიგის დატოვება
            ADVANCE   9,3         ; ოპერატორის მომს. დრო
            LEAVE     OPERATOR    ; ოპერატორის განთავ-ება
            ADVANCE   4,1         ; მომსახურების დრო
            TRANSFER .15,,ENGAGED ; გარე ხაზის 15 %
                                      დაკავებულია
            ENTER     EXTENSIONS   ; სააბონენტო სადგურის
                                       დაკავება
            ADVANCE   6,2      ; ზარის განხორციელების დრო
            TRANSFER .200,,NOPERSON ; ზარების 20 %  არ
                                        პასუხობს
            ADVANCE   V$CALLTIME  ; ზარის მომს. დრო
            TABULATE  TRANSIT     ; სატრ. დროის ტაბულირება
NOPERSON    LEAVE     EXTENSIONS ; სააბონენტო სადგურის
                                     განთავისუფლება
ENGAGED     LEAVE     EXTLINES ; გარე ხაზის განთავისუფლება
            TERMINATE           ; სააბონენტო პუნქტის
                                    დატოვება

*              MODEL SEGMENT 3

           GENERATE    3600    ; მოდელირების დრო
           TERMINATE   1       ; დასასრული
           GENERATE    60      ; მოდელირების დრო
           TABULATE    CALLSINT ; შიდა ზარების ტაბულირება
           TABULATE    CALLSEXT ; გარე ზარების ტაბულირება
          TERMINATE   1       ; დასასრული
```

ამოცანები დამოუკიდებელი მუშაობისათვის

1. კომპიუტერული სისტემა შედგება ერთი პროცესორისაგან, რომლის მზა ბრძანებათა ბუფერში ბრძანებები შედის 10±5 ნანოწმ-ის ინტენსიურობით. მზა ბრძანებათა ბუფერში რიგის ორგანიზაცია მიმდინარეობს **FIFO** დისციპლინით. ამის შემდეგ ბრძანებები პროცესორში შედის 0, 000016±0,000015 წმ-ის ინტენსიურობით.

   შეადგინეთ **GPSS** მოდელი ამ სისტემის გამოსაკვლევად (რიგისა და პროცესორის დატვირთვის ანალიზისათვის). მოდელირების დროა 5 წთ.

2. კომპიუტერული სისტემა შედგება ერთი პროცესორისაგან, რომლის მზა ბრძანებათა ბუფერში შედის ორი ტიპის ბრძანება. ამასთან ერთად, I ტიპის ბრძანებები შედის 17±8 ნანოწმ-ისა და II ტიპის ბრძანებები – 12±5 ნანოწმ-ის ინტენსიურობით. ამის შემდეგ ბრძანებები მზა ბრძანებათა ბუფერიდან შედის პროცესორში შესასრულებლად: I ტიპის ბრძანებების შესრულების დროა 0, 000023±0, 000013 წმ, ხოლო II ტიპის ბრძანებების — 0, 000015±0, 000005 წმ. აღნიშნულ დიაპაზონებში ბრძანებების მოსვლისა და მათი პროცესორზე მომსახურების დრო განაწილებულია თანაბარალბათურად. ამასთან ერთად,სისტემაში რიგისა და მომსახურების მიხედვით

II ტიპის ბრძანებებს I-თან შედარებით გააჩნიათ მაღალი პრიორიტეტი.

შეადგინეთ **GPSS** მოდელი ამ სისტემის გამოსაკვლევად (რიგისა და პროცესორის დატვირთვის ანალიზისათვის). მოდელირების დროა 10 წთ.

3. კონვეირული პროცესორული სისტემა შედგება ორი პროცესორისაგან, სადაც ბრძანებები მიმდევრობით გადის მომსახურებას, სისტემაში გენერირებული (მოსული ბრძანებები) და პროცესორზე მომსახურებული ბრძანებები დგებიან პროცესორთან მომსახურების რიგში. პირველ პროცესორთან რიგში მისვლის დროა 17±5 წმ, ხოლო მეორეზე – 22±12 წმ. რიგში მისვლის დრო განაწილებულია თანაბარალბათური კანონით.

   შეადგინეთ **GPSS** მოდელი ამ სისტემის გამოსაკვლევად. მოდელირების დროა 1 სთ.

4. კონვეირული პროცესორული სისტემა შედგება სამი პროცესორისაგან, სადაც ბრძანებები მიმდევრობით გადის მომსახურებას, სისტემაში გენერირებული (მოსული ბრძანებები) და პროცესორზე მომსახურებული ბრძანებები დგებიან რიგში შემდგომ პროცესორთან (ბოლო მე-3 პროცესორიდან მომსახურებული ბრძანებები სისტემიდან მიდიან). პირველ პროცესორთან რიგში მისვლის დრო განაწილებულია თანაბარალბათური კანონით და 50±15 წმ-ის ტოლია. პირველ პროცესორზე ერთი ბრძანების მომსახურების დროა 60±25 წმ, მეორე პროცესორზე – 70±30 წმ, ხოლო მესამე პროცესორზე – 80±35 წმ.

შეადგინეთ **GPSS** მოდელი ამ სისტემის გამოსაკვლევად. მოდელირების დროა 3 სთ.

5. მულტიპროცესორული სისტემა შედგება სამი პროცესორისაგან და საერთო მეხსიერებისაგან. იმისათვის, რომ ბრძანებები შესრულდეს თითოეულ პროცესორზე გადის პირველად დამუშავებას (მომზადებას შესასრულებლად), რომლის დროსაც ოპერატიული მეხსიერებიდან ხდება ბრძანების მისამართების მიხედვით ოპერაციის ოპერანდის მომზადება. ამის შემდეგ ოპერანდი ხვდება მზა ბრძანებათა ბუფერში და შემდეგ კი თითოეულ პროცესორზე. ბრძანებათა დამუშავება ბუფერში შედგება შემდეგი ეტაპებისაგან:

I ეტაპზე გენერირდება თითოეული ტიპის 100 ბრძანება.

I ტიპის ბრძანება დეტერმინირებული დროით 37 ნანოწმ-ით გენერირდება.

II ტიპის ბრძანება – 30±15 ნანოწმ-ით გენერირდება.

III ტიპის ბრძანება – 45±15 ნანოწმ-ით გენერირდება.

II ეტაპზე ხდება ბრძანების დამუშავება. მისი დაშლა ხდება ბრძანების კოდად და ბრძანების ოპერანდად, რომლის შესრულების დრო დეტერმინირებულ რიცხვს 30 ნანოწმ-ს შეადგენს და სამივე ტიპის ბრძანებისათვის ტოლია.

III ეტაპზე ბრძანებები ოპერატიულ მეხსიერებასთან ოპერანდების ამოღებაზე რიგში დგებიან.

IV ეტაპი ოპერატიული მეხსიერებიდან ოპერანდების ამოღების დროა, რომელიც ყველა ტიპის ბრძანებისათვის 35 ნანოწმ-ის ტოლია.

V ეტაპზე მომზადებული ბრძანებები მზა ბრძანებათა ბუფერში ხვდებიან ბრძანებათა ტიპის მიხედვით.

VI ეტაპი ხდება ბრძანების შესრულება, რომლის დრო ბრძანებათა ტიპზეა დამოკიდებული:

I ტიპის ბრძანებისათვის შესრულების დროა 40±20 ნანოწმ;

II ტიპის ბრძანებისათვის – 60±25 ნანოწმ;

III ტიპის ბრძანების შესრულების დრო 60±25 ნანოწმ.

VII ეტაპზე ბრძანებები შესრულებულ ბრძანებათა ბუფერში ხვდებიან.

შეადგინეთ **GPSS** მოდელი ამ სისტემის გამოსაკვლევად. მოდელირების დროა 3 სთ.

6. მულტიპროცესორული სისტემა შედგება სამი პროცესორისაგან. იმისათვის, რომ ბრძანება შესრულდეს პროცესორში, საჭიროა შემდეგი ეტაპების შესრულება:

I ეტაპი – ბრძანებათა გენერირება შესრულებაზე;

II ეტაპი – შესასრულებელ ბრძანებათა რიგი;

III ეტაპი – ბრძანების დამუშავება;

IV ეტაპი – რიგი მეხსიერებასთან;

V ეტაპი – ოპერაციული მეხსიერებიდან ბრძანების ოპერანდების ამოღება;

VI ეტაპი – შესრულებისათვის მზა ბრძანებათა ბუფერი;

VII ეტაპი – ბრძანების შესრულება;

I÷VII ეტაპები ყველა პროცესორისათვის ინდივიდუალურია. მათ საერთო აქვთ VIII ეტაპი – იგი პროცესორებს შორის მონაცემთა გადაცემას შეესაბამება.

IX ეტაპი – დასასრული.

I ეტაპზე სამივე ტიპის ბრძანების გენერირების დრო ერთნაირია. იგი დეტერმინირებულ სიდიდეს წარმოადგენს და ტოლია 15±5 წმ-ის.

III ეტაპზე ბრძანების კოდისა და ოპერანდების სამისამართო ველად გაშლის დრო სამივე ტიპის ბრძანებისათვის 17 ნანოწმ-ია.

V ეტაპზე ოპერანდის ამოღების დრო (ოპერანდის მისამართის მიხედვით) 25 ნანოწმ.

VII ეტაპზე ბრძანების შესრულების დროა 17 ნანოწმ (სამივე პროცესორისათვის)

VIII ეტაპზე შედეგების გადაცემის დროა 10 ნანოწმ.

შეადგინეთ **GPSS** მოდელი ამ სისტემის გამოსაკვლევად. მოდელირების დროა 7 წთ.

7. ააგეთ ხაზოვანი ტოპოლოგიის **GPSS** მოდელი 3 პროცესორისათვის. პირველ პროცესორზე (ხაზოვანი ტიპის ტოპოლოგიის დასაწყისი) შესრულებულ ბრძანებათა გენერაციის დინამიკა 100±47 ნანოწმ-ია. ხაზოვანი კომპიუტერით პირველ პროცესორზე ბრძანების შესრულების დროა 120 ±40 ნანოწმ, II პროცესორზე –0,15±0,007 მიკროწმ., ხოლო მესამე კომპიუტერზე ბრძანებათა შესრულების დრო 0,00017±0,00008წმ. არსებობს თითოეულ პროცესორზე შემავალი ბუფერი (რიგები).

შეადგინეთ **GPSS** მოდელი ამ სისტემის გამოსაკვლევად. მოდელირების დროა 16 წთ.

8. ბრძანებები რიგში მოდის 16±4 ინტენსიურობით, რომლებიც სრულდება 20 პარალელურად მომუშავე პროცესორზე. შესრულების დრო განაწილებულია თანაბარალბათურად და 0,06±0,02 მილიწმ-ის ტოლია.

   შეადგინეთ **GPSS** მოდელი ამ სისტემის გამოსაკვლევად. მოდელირების დროა 3 სთ.

9. სასწავლო ლაბორატორიაში განთავსებულია კომპიუტერული ლაბორატორია, რომელიც 10 კომპიუტერისაგან შედგება. კომპიუტერები ყოველთვიურად გამოდის მწყობრიდან. კომპიუტერი დღის განმავლობაში მუშაობს 12 საათის განმავლობაში. მწყობრიდან გამოსვლის დრო განაწილებულია 200±180 სთ. ინტერვალში. მწყობრიდან გამოსული კომპიუტერი ექვემდებარება რემონტს, რომლის დროა 20±5 სთ. მუშაობის დასაწყისში იმყოფება 10 მუშა მდგომარეობაში მყოფი კომპიუტერი. ამასთან ერთად, 2 სარეზერვო და ერთი რემონტის მდგომარეობაში მყოფი კომპიუტერები, რომლითაც ხდება მწყობრიდან გამოსული კომპიუტერების შეცვლა. რემონტს აწარმოებს მხოლოდ ერთი ხელოსანი.

   შეადგინეთ სისტემის შესაბამისი **GPSS** მოდელი. მოდელირების დროა 25 სამუშაო დღე (8 საათიანი).

10. ერთ დეპარტამენტში 3 სასწავლო ლაბორატორიაა, სადაც პირველ მათგანში განთავსებულია 10 კომპიუტერი, მეორეში – 12, ხოლო მესამეში კი – 15. პირველ და მეორე ლაბორატორიებში დგას ყვითელი აწყობის კომპიუტერები, რომელთა მწყობრიდან გამოსვლა ხორციელდება 30±15 დღის განმავლობა-

ში. მესამე ლაბორატორიაში კი – ორიგინალი ამერიკული აწყობის კომპიუტერები 60±20 დღის განმავლობაში. კომპიუტერების ყოველდღიური დატვირთვა შეადგენს 12 სთ-ს. მათ შესაცვლელად მარაგში იმყოფება 3 კომპიუტერი. გასარემონტებელ სამუშაოს 2 ხელოსანი აწარმოებს. თითოეული მათგანი რემონტს 15±7 სთ-ს ანდომებს. შესაკეთებელი კომპიუტერების აღდგენილით შეცვლას სჭირდება 1 სთ.

შეადგინეთ სისტემის შესაბამისი **GPSS** მოდელი 15 სამუშაო კვირის განმავლობაში.

11. სასწავლო ლაბორატორიაში განთავსებულია კომპიუტერული ლაბორატორია, რომელიც შედგება 10 კომპიუტერისაგან, რომლებიც ყოველთვიურად გამოდის მყობრიდან. კომპიუტერების ყოველდღიური დატვირთვა შეადგენს 12 სთ-ს. მწყობრიდან გამოსვლის დრო იცვლება 150±30 სთ ინტერვალით. მწყობრიდან გამოსული კომპიუტერების გარემონტების დრო შეადგენს 15±5 სთ-ს. მუშაობის დასაწყისში 10 მუშა მდგომარეობაში მყოფ კომპიუტერთან ერთად იმყოფება 3 სარეზერვო კომპიუტერი, რომლებითაც წარმოებს მწყობრიდან გამოსული კომპიუტერების შეცვლა. რემონტს აწარმოებს მხოლოდ ერთი ხელოსანი.

შეადგინეთ სისტემის შესაბამისი **GPSS** მოდელი 12 სამუშაო კვირის განმავლობაში. (ამოცანა გაუშვით აგრეთვე იმ შემთხვევებში, როდესაც სარეზერვო კომპიუტერების რაოდენობა ტოლია 4 და 5-ის).

12. სისტემა შედგება მზა ბრძანებათა ბუფერისა და ბრძანებათა შესრულების პარალელური პროცესორისაგან. ბრძანებები სისტემაში შედის 8±4 წმ ინტერვალით და თითოეული პროცესორის შესრულების დრო ვარირებს 40±10 ნანოწმ-ის ინტერვალით.

    შეადგინეთ **GPSS** მოდელი, რომლის დროსაც პარალელურად მომუშავე პროცესორების ცვლილების გათვალისწინებით პროცესორების რაოდენობა იცვლება 5-დან 10-მდე ინტერვალში. მოდელირების დრო შეადგენს 24 სთ-ს.

13. მმართველი მუშა პროცესორი ამზადებს ბრძანებებს შესასრულებლად, რომლის დროსაც მონაცემებს იღებს ფაილური სერვერიდან. ამასთან, მზა ბრძანებებს შესასრულებლად ანაწილებს პარალელურად მომუშავე 256 პროცესორზე. მიღებული შედეგები კომუტატორის დახმარებით იწერება პარალელურად მომუშავე ფაილურ სისტემაში.

    შეადგინეთ სისტემის შესაბამისი **GPSS** მოდელი, როდესაც მოდელირების დრო შეადგენს 24 სთ-ს.

14. პენტიუმის ტიპის კომპიუტერები ჩართულია უნივერსიტეტის გამოთვლით ლაბორატორიებში. მათი მეშვეობით ხდება ამოცანების გაგზავნა ერთიან კლასტერზე შესასრულებლად. პენტიუმებიდან მიცემულ დავალებებს კომუტატორი ანაწილებს 3 შუალედურ კომუტატორზე. ეს უკანასკნელები კი დავალებებს აძლევს პროცესორებს. თითოეული პროცესორის შესრულების დროა 500±50 ნანოწმ.

შეადგინეთ სისტემის შესაბამისი **GPSS** მოდელი, როდესაც მოდელირების დრო შეადგენს 24 სთ-ს.

15. მეცნიერებათა აკადემიის კომპიუტერული ცენტრი ასრულებს რთულ გამოთვლებს, რომელიც ჩართულია აკადემიის სხვადასხვა ინსტიტუტებთან და მათგან იღებს დავალელებს 20±5 ნანოწმ-ის, 15±3 ნანოწმ-ის და 10±2 ნანოწმ-ის ინტენსიურობით.

შეადგინეთ სისტემის შესაბამისი **GPSS** მოდელი, როდესაც მოდელირების დრო შეადგენს 24 სთ-ს.

16. მიკროპროცესორები აწყობის შემდეგ მიაქვთ ტექნიკური კონტროლის ადგილზე 35±15 წთ-ის ინტერვალით, სადაც იმყოფება 3 სპეციალისტი. ამასთან ერთად, თითოეულ სპეციალისტს შესამოწმებლად სჭირდება 25±15 წთ. დრო. გასინჯული მიკროპროცესორების 75 % კარგია და მას უდგამენ კომპიუტერებს, ხოლო გაფუჭებული მიკრო-პროცესორები მიაქვთ გამართვის ადგილზე. ამის შემდეგ მიკროპროცესორები კვლავ გადის ტექნიკურ კონტროლს.

შეადგინეთ **GPSS** მოდელი, რომელიც შეაფასებს ერთი სამუშაო დღის განმავლობაში მიკროპროცესორების რიგის სიგრძეს ტექნიკური კონტროლისა და გამართვის ადგილებზე.

17. კომპიუტერულ ფირმაში მიკროპროცესორების აწყობის გრძელი პროცესი მთავრდება ნოუთბუქში მათი შემოწმების მოკლე პროცესით. ამასთან ერთად, 5 სპეციალისტი იყენებს ერთ ნოუთბუქს (ვინაიდან მისი სიძვირის გამო ფირმას რამდენიმე ნოუთბუქის ყიდვის შესაძლებლობა არ აქვს).

მიკროპროცესორების აწყობისა (**Mikroprocessor, D7**) და ნოუთბუქში მათი შემოწმების პროცესის დროის ინტერვალი განაწილებულია არათანაბარალბათურად (**NoteBook, D5**) დისკრეტული ფუნქციების სახით.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს 3 სამუშაო დღის განმავლობაში სპეციალისტების სინქრონიზებულ მუშაობას ნოუთბუქის გამოყენებისას, რომლის დროსაც ადგილი არ ექნება სპეცილისტების მოცდენებს.

18. სასწავლო დანიშნულების კომპიუტერულ ცენტრში დგას 5 ცალი ინტერნეტში ჩართული კომპიუტერი. ამ ცენტრში მოსული სტუდენტების ნაკადი მოიცემა პუასონის განაწილების სახით დროის საშუალო ინტერვალით 25 წთ, ხოლო სტუდენტების მომსახურების საშუალო დროის ინტერვალი განაწილებულია ექსპონენციალურად და ტოლია 12 წთ-ის. თუ ადგილი კომპიუტერზე არ არის თავისუფალი, მაშინ ზოგიერთი სტუდენტი მიდის სხვაგან, ზოგი კი – დგება რიგში და ელოდება კომპიუტერის განთავისუფლებას.

შეადგინეთ **GPSS** მოდელი, რომელიც 5 სთ-ის განმავლობაში მოდელირების შედეგების მიხედვით შეაფასებს მომსახურებისა და მის გარეშე დარჩენილი სტუდენტების წილს.

19. ერთ-ერთ საწარმოში მესაწყობე კომპიუტერის სპეციალისტებს გაფუჭებულ მიკროპროცესორებს უცვლის ახლით. არსებობს ორი ტიპის მიკროპროცესორები. I და II ტიპის მიკროპროცესორების საჭიროების შემთხვევაში კომპიუტერის სპეციალისტების მოსვლის დროები განაწილებულია უწყვეტი

ფუნქციების (**Specialisti1, C5** და **Specialisti2, C5**) სახით. ამასთან ერთად, I ტიპის მიკროპროცესორების მომსახურებას სჭირდება გაცილებით ნაკლები დრო (**STime1)**, ვიდრე II ტიპის (**STime2**). რაც ნაკლებია მიკროპროცესორების მომსახურების დროებს შორის ინტერვალი, მით მაღალია პრიორიტეტი.

შეადგინეთ **GPSS** მოდელი, რომლის თანახმად შემცირდება რიგში მყოფი ორივე კატეგორიის სპეციალისტის რიცხვი.

20. ერთი არხის მქონე სარემონტო ქვეგანყოფილებაში შედის მწყობრიდან გამოსული კავშირის საშუალებები, რომელთაც ესაჭიროებათ მიმდინარე რემონტი. მწყობრიდან გამოსულ, უწესივრო კავშირის საშუალებათა შემოსვლებს შორის დროის ინტერვალები განაწილებულია თანაბარალბათურად 18±6 სთ-ის ინტერვალით. ამასთან ერთად, რემონტის დრო ასევე განაწილებულია თანაბარალბათურად 16±4 სთ-ის ინტერვალით. რემონტი წარმოებს შემდეგი პირობის თანახმად: „პირველი მოვიდა – პირველი გარემონტდა“.

შეადგინეთ **GPSS** მოდელი, რომელიც განსაზღვრავს სარემონტო ქვეგანყოფილების ფუნქციონირებას 3 დღე-ღამის განმავლობაში (72სთ). აგრეთვე, მოდელმა უნდა უზრუნველყოს რიგის შესახებ სტატისტიკური ინფორმაციის შეგროვება. სამოდელო დროის ერთეულია 1 წთ, ხოლო მოდელირების დრო შეადგენს 72 x 60 = 4320 ერთეულს.

ლიტერატურის სია

1. Максимей И.В. Имитационное моделирования на ЭВМ.- М.: Радио и связь, 1988. - 232 с.
2. Шрайбер Т. Дж. Моделирования на GPSS. - М.: Машиностроение, 1980. -592 с.
3. Руководство пользователя по GPSS World. /перевод с английского/. – Казань: Изд-во „Мастер Лайн“, 2002. - 384с.
4. Асеев А. А. , Боев В. Д. и др. Основы моделирования систем связи и автоматизации на GPSS/PC – Учеб. Пособие. – СПБ.: ВУС, 2000. -230 с.
5. Боев В. Д. Имитационное моделирование на GPSS/PC ситем обеспечения войск.Учеб. Пособие. – СПБ.: ВАУ, 1999. - 186 с.
6. Боев В. Д., Сыпченко Р.П. Компютерное моделирование:Руководство по курсовому проектированию. СПБ.: ВУС, 2002. - 96 с.
7. Методические указания для слушателей ФПКП по моделированию систем и сетей связи на GPSS/PC – Ч. 1. Основы моделирования на GPSS/PC/ Сост.: Воробейчиков Л.А., Сосновиков Г.К. - М.: МИС, 1993.
8. http://www.gpss.ru/paper/dev_yak/3_w.html.
9. Боев В. Д. Моделирование систем – Инструменталные средства GPSS WORLD, Санкт-Петербург , „БХВ - Петербург“, 2004, - 348 с.

10. Норенков И.П. Разработка САПР.- М, МГТУ им.Баумана,1994. - 274 с.
11. Бусленко Н. П. Моделирование сложных систем. Москва, 1978 г. - 314 с.
12. Машинные имитационные эксперименты с моделями экономических систем./ Под ред. Нейлора Т. М. / М.: Мир, 1975 г. - 501 с.
13. Советов Б.Я., Яковлев С. А. Моделирование систем: Учебник для ВУЗов.- М.: Высшая школа, 2001. -235 с.
14. Иващенко А. В., Сыпченко Р. П. Основы моделирования сложных систем на ЭВМ: Учебник. –Л.: ЛВВИУС, 1988. - 272 с.
15. Максимей И. В. Имитационное моделирование на ЭВМ: - М.: Радио и связь,1988. – 232 с.
16. Учебное пособие по GPSS World. / перевод с английского /. – Казань: Изд-во „Мастер Лайн“, 2002. - 270 с.
17. http://www.minutemansoftware.com/general_information.html
18. Томашевский В.Н., Жданова Е. Г. Имитационное моделирование в среде GPSS. – М.: Бестселлер, 2003. – 416 с.
19. Кудрявцев Е. М. GPSS World. Основы имитационного моделирования различных систем. – М.: ДМК Пресс, 2003. – 320 с.

კომპიუტერული უზრუნველყოფა ი. აბულაძის

იბეჭდება ავტორთა მიერ წარმოდგენილი სახით

გადაეცა წარმოებას 30.01.2009. ხელმოწერილია დასაბეჭდად 05.02.2009. ქაღალდის ზომა 60X84 1/16. პირობითი ნაბეჭდი თაბახი 22. ტირაჟი 100 ეგზ.

საგამომცემლო სახლი „ტექნიკური უნივერსიტეტი“, თბილისი,
კოსტავას 77