საქართველოს ტექნიკური უნივერსიტეტი

ხელნაწერის უფლებით

თეოდორე ზარქუა

პროგრამული უზრუნველყოფის მობილურობისა და გაფართოებადობის ზოგიერთი ასპექტი

დოქტორის აკადემიური ხარისხის მოსაპოვებლად
წარდგენილი დისერტაციის
ა ვ ტ ო რ ე ფ ე რ ა ტ ი

სადოქტორო პროგრამა “ინფორმატიკა”, შიფრი 0401

თბილისი
2015 წელი

სამუშაო შესრულებულია საქართველოს ტექნიკურ უნივერსიტეტში
ინფორმატიკისა და მართვის სისტემების ფაკულტეტი
ინტერდისციპლინარული ინფორმატიკის დეპარტამენტი

ხელმძღვანელები: პროფ. ჰამლეტ მელაძე
პროფ. ოლეგ ნამიჩეიშვილი

რეცენზენტები: პროფ. ნანა გულუა
პროფ. ლევან იმნაიშვილი

დაცვა შედგება ---------- წლის ”------” ----------, --------- საათზე
საქართველოს ტექნიკური უნივერსიტეტის ----------
---------- ფაკულტეტის სადისერტაციო საბჭოს კოლეგიის
სხდომაზე, კორპუსი ----------, აუდიტორია ----------
მისამართი: 0175, თბილისი, კოსტავას 77.

დისერტაციის გაცნობა შეიძლება სტუ-ს ბიბლიოთეკაში,
ხოლო ავტორეფერატისა - ფაკულტეტის ვებგვერდზე

სადისერტაციო საბჭოს მდივანი პროფ. თინათინ კაიშაური

ნაშრომის ზოგადი დახასიათება

თემის აქტუალობა და გამოყენების სფერო. დღეს პროგრამული უზრუნველყოფის გამოყენების არემ პრაქტიკულად დაფარა ადამიანის საქმიანობის ყველა სფერო. წინა პლანზე გამოდის გამოყენების ეფექტიანობა, რაც განაპირობებს მზარდ მოთხოვნებს პროგრამული უზრუნველყოფის ხარისხის მიმართ. ეს კი, თავის მხრივ, მიიღწევა, ერთის მხრივ, პროგრამისტისთვის შეთავაზებული საინსტრუმენტო საშუალებათა დახვეწით, მეორეს მხრივ, კი თავად დამპროგრამებლის კვალიფიკაციის ამაღლებით და, აქედან გამომდინარე, მის მიერ დაწერილი კოდის ხარისხის გაუმჯობესებით. წინამდებარე ნაშრომში შედეგიანად არის გამოყენებული ორივე მიდგომა. ნაშრომში მიღებული თეორიული შედეგების საფუძველზე შემუშავებული პროგრამული უზრუნველყოფა არსებითად აძლიერებს დაპროგრამების ენების შესაძლებლობებს, რადგან აწვდის პრობლემურ პროგრამისტს ფუნქციონალური გამოსახულებების სიმბოლურად დამუშავების აპარატს. რაც, თავის მხრივ, განაპირობებს მთელი რიგი ამოცანის კომპიუტერზე გადაწყვეტის თვისობრივად ახალ შესაძლებლობებს. საგულისხმოა, რომ სიმბოლური დიფერენცირების პროგრამული რეალიზაცია თავად წარმოადგენს გაფართოებადი პროგრამული უზრუნველყოფის საინტერესო მაგალითს. იმის გამო, რომ გაწარმოების წესები მოცემულია შიდა ენაზე ჩამოყალიბებული ფორმულებით, უკიდურესად მარტივდება ამავე პროგრამის მიერ რაიმე ახალი ფუნქციონალური გარდაქმნის გათვალისწინება – საკმარისია ახალი გარდაქმნა აღიწეროს შიდა ენაზე და მოთავსდეს ფორმულების ვექტორში. ასევე ძალიან ადვილია პროგრამაში ახალი ელემენტარული ფუნქციის ჩამატება – ამისათვის საკმარისია დაემატოს ეს ფუნქცია ელემენტარული ფუნქციების უკვე არსებულ სიას, დაემატოს შესაბამისი ფორმულა დიფერენცირების წესების განმსაზღვრელ ფორმულების სიას (შიდა ენაზე) და გათვალისწინებულ იქნას რექგირება ამ ფუნქციაზე გამოსახულების მნიშვნელობის განმსაზღვრელ ფრაგმენტში.

დღეს ეჭვგარეშეა, რომ პროგრამისტის პროფესია გახდა ერთ-ერთი ყველაზე მოთხოვნილი და მნიშვნელოვანი. მეტიც, დღეს

დაპროგრამების საფუძვლების ცოდნა მოეთხოვება უკვე იმ საგნობრივი არის სპეციალისტსაც, ვინც წარმოადგენს პროგრამული უზრუნველყოფის მომხმარებელს, რადგან პრობლემური პროგრამების შექმნის **სტანდარტული სქემა, როდესაც პროგრამისტს არ მოეთხოვებოდა სათანადო საგნობრივი არის ცოდნა ვერ უზრუნველყოფს პროგრამული პროდუქტის თანამედროვე მოთხოვნების შესაბამის ხარისხს**. ამ პირობებში ტექნოლოგიური თვალსაზრისით ყველაზე მოწინავე სახელმწიფოებიც კი ცდილობენ საგანგებო ზომები გაატარონ იმისათვის, რომ მოახერხონ მაღალკვალიფიციური კადრების საჭირო რაოდენობით მომზადება, რათა არ აღმოჩნდენ პროგრამისტულ პოზიციებზე სპეციალისტების გარედან მოწვევის აუცილებლობის წინაშე.

ცხადია, ეს საკითხი **აქტუალურია საქართველოსთვისაც**. საკითხის აქტუალობა მნიშვნელოვნად იზრდება იმის გათვალისწინებით, რომ საქართველოს მოსახლეობას ოდითგანვე ახასიათებს მაღალი ინტელექტუალური პოტენციალი და, საზოგადოდ, ლტოლვა ინტელექტუალური საქმიანობისკენ. დასტურად შეიძლება გამოდგეს უმდიდრესი ხალხური შემოქმედება (ხალხური პროზა და პოეზია, უპრეცედენტო ჟღერადობის, სირთულის და მრავალფეროვნების ხალხური სიმღერები, უნიკალური ცეკვები, ხალხური რეწვა, ...). ინტელექტუალური საქმიანობა ქართველი კაცისთვის ყოველთვის მაღალპრესტიჟულად ითვლებოდა. აქედან გამომდინარე, ლაპარაკია იმაზე რომ მაქსიმალურად გამოყენებულ იქნას მოსახლეობის მაღალი ინტელექტუალური პოტენციალი და მიეცეს მას ნორმალურად განვითარების საშუალება. საკითხის აქტუალობა კიდევ უფრო მწვავდება იმის გათვალისწინებით, რომ დღეისათვის არ არსებობს დე ფაქტო სტანდარტად მიჩნეული დაპროგრამების სასტარტო სწავლების მეთოდოლოგია. ასეთად ვერ გადაიქცა ვერც დონალდ კნუტის მიერ შემოთავაზებული ვირტუალურ MIX და MMIX კომპიუტერებზე დამყარებული მეთოდოლოგია და ვერც ნიკლაუს ვირტის მიერ შემოთავაზებული პასკალის სწავლებაზე დამყარებული მეთოდოლოგია. უფრო სწორედ, ეს უკანასკნელი უკვე არ არის დე ფაქტო სტანდარტი, რადგან შეწყდა დაპროგრამების ენა პასკალის ტექნოლოგიური

მხარდაჭერა (სამართლიანობა მოითხოვს შევნიშნოთ, რომ 90-იან წლებში და 2000-იანი წლების დასაწყისში სწორედ ეს მეთოდოლოგია ითვლებოდა ყველაზე უფრო გამართლებულად).

ასეთ პირობებში საკმაოდ ინტენსიურად ხდება სხვადასხვა მიდგომების ჩამოყალიბება. მაგრამ ჩვენი მდგომარეობა ერთიორად გართულებულია იმით, რომ არცერთი სხვა სახელმწიფოს უმაღლესი სკოლა არ არის ჩვენნაირ მდგომარეობაში. ეს ნიშნავს, რომ გარედან თუნდაც აღიარებული მეთოდიკის ბრმად გადმოღება ძირშივე მცდარია, რადგან ის გათვალისწინებულია სულ სხვა სასტარტო პირობებზე.

მოცემულ ნაშრომში შემოთავაზებულია მეთოდიკა, რომელიც ეყრდნობა საქართველოში უკვე აპრობირებულს. შემოთავაზებული მეთოდიკის წინამორბედად ავტორს მიაჩნია აკადემიკოს ილია ვეკუას სახელობის გამოყენებითი მათემატიკის სამეცნიერო კვლევითი ინსტიტუტის ნორჩ მათემატიკოს-პროგრამისტთა სკოლაში წლების განმავლობაში დანერგილი ეგმ ბესმ-6-თვის პოსტის დიალოგური მანქანის რეალიზაციაზე დამყარებული დაპროგრამების საფუძვლების წარმატებული სწავლება.

სამუშაოს მიზანი, კვლევის ობიექტი და მეთოდები, ძირითადი შედეგები და მეცნიერული სიახლე.

ნაშრომის მიზანია ჩატარებული კვლევის შედეგად პრაქტიკულად ღირებული საშუალებების შექმნა, რომლებიც, ერთის მხრივ, შეუწყობს ხელს უფრო ხარისხიანი პროგრამული უზრუნველყოფის შექმნას, მეორეს მხრივ კი გაზრდის დაპროგრამების სასტარტო სწავლების ქმედითობას.

კვლევის ობიექტია ფუნქციონალური გამოსახულებები. გამოკვლეულია მათი წარმოდგენის ხერხები და მათზე შესასრულებელი ოპერაციები. კვლევის ობიექტია ასევე ტიურინგის ვირტუალური მანქანა. გამოკვლეულია მისი, როგორც ბრძანებათა მინიმალური სისტემის პირობებში ალგორითმების უნივერსალური შემსრულებლის, გამოყენების პერსპექტივა დაპროგრამების საფუძვლების სწავლებისთვის.

კვლევის მეთოდები მოიცავს როგორც თეორიულ, ასევე პრაქტიკულ მეთოდებს. კერძოდ, მოცემული ნაშრომის თეორიული შედეგების

მისაღებად გამოყენებული იყო შესწავლა და განზოგადების, ანალიზისა და სინთეზის, აბსტრაგირების, ფორმალიზაციის, აგრეთვე ინდუქციისა და დედუქციის მეთოდები. პრაქტიკული შედეგების მისაღებად გამოყენებულ იქნა ისეთი მეთოდები, როგორიცაა კომპიუტერული მოდელირება, დაკვირვება, შედარება, და იგივე სინთეზი, ინდუქცია და დედუქცია.

ძირითადი შედეგებია:

1. ალგებრულ გამოსახულებათა უფრჩხილებო, ე.წ. პოლონური ჩანაწერების მიმართ შეუღლებულის ცნების შემოღება და ამ ცნებასთან დაკავშირებული კანონზომიერებების დადგენა. ავტორმა მკაცრად მათემატიკურად დაამტკიცა შესაძლებლობა მთლიანად მოიხსნას პოლონური ჩანაწერების პოსტფიქსური და პრეფიქსული ფორმების ურთიერთგარდაქმნის აუცილებლობა. ეს გარემოება კი არსებითია ისეთ ამოცანებში, სადაც წარმოიშვება ფუნქციონალურ გამოსახულებათა არა მარტო რიცხვითი მნიშვნელობების მიღების, არამედ მათი ფუნქციონალური გარდაქმნის საჭიროება. ამ შედეგებზე დაყრდნობით შემუშავდა პროგრამული უზრუნველყოფა, რომელიც აძლევს მომხმარებელს შესაძლებლობას იმუშავოს ფუნქციონალურ გამოსახულებებთან ისევე, როგორც მონაცემთა სტანდარტულ ტიპებთან.
2. შემუშავებულია გამოსახულებათა პოლონური ჩანაწერების შეუღლებულის ცნებასთან დაკავშირებული ალგორითმები, რომლებიც უზრუნველყოფს მათ გარდაქმნას სტანდარტულ პოლონურ ჩანაწერებზე და პირიქით.
3. შემუშავებულია ინფიქსური ჩანაწერიდან პრეფიქსული ჩანაწერის მიღების პირდაპირი ალგორითმი, დამყარებული 2 სტეკის გამოყენებაზე.
4. შემუშავებულია მეთოდიკა, განკუთვნილი პრობლემური პროგრამისტის მიერ ფუნქციონალურ გამოსახულებებზე გაწარმოების გარდაქმნების განხორციელებისთვის დაპროგრამების ე.წ. ალგორითმული ენებიდან, რომელიც ემყარება ამ მიზნით შექმნილ შიდა ენას.

5. შემუშავებულია სპეციალური კლასის საინტერფეისო ნაწილი, რომელიც უზრუნველყოფს პრობლემურ პროგრამისტს ფუნქციონალურ გამოსახულებათა დამუშავების მოსახერხებელი საშუალებებით ალგორითმული ენის დონიდან.
6. შემუშავებულია ტიურინგის ვირტუალური მანქანის მოდიფიკაცია, განკუთვნილი დაპროგრამების საწყისების სწავლებისთვის.

ნაშრომის **მეცნიერული სიახლე** განპირობებულია შემდეგით:

- ავტორის მიერ შემოღებული პოლონური ჩანაწერის შეუღლებულის ცნება და, აქედან გამომდინარე, ამ ცნებასთან დაკავშირებული მის მიერ ჩამოყალიბებული და დამტკიცებული 7 თეორემა და 8 შედეგი სრულიად ახალია;
- სიახლეს წარმოადგენს ავტორის მიერ ჩამოყალიბებული ინფიქსური ჩანაწერიდან პირდაპირ პრეფიქსული ჩანაწერის მიღების ალგორითმი, დამყარებული ერთდროულად 2 სტეკის გამოყენებაზე;
- ახალია შეუღლებულის ცნებასთან დაკავშირებული ყველა ალგორითმი;
- სიახლეს წარმოადგენს ავტორის მიერ შემოთავაზებული შიდა ენა გაწარმოების წესების აღწერისთვის, ასევე სიახლეა რეალიზაცია, რომელიც ეყრდნობა ამ ენას და გამოირჩევა როგორც გაფართოებადობით, ასევე მობილურობით;
- მთლიანობაში, სიახლეა ავტორის მიერ შემუშავებული პროგრამული საშუალებები, რომელთა მეშვეობითაც პრობლემურ პროგრამისტს ეძლევა შესაძლებლობა ალგორითმული ენის დონიდან დაამუშავოს სტრიქონის მეშვეობით ბუნებრივი სახით მოცემული ფუნქციონალური გამოსახულება. დამუშავებაში შედის ამ გამოსახულების არა მარტო მნიშვნელობების გამოთვლა (პარამეტრის გარკვეული მნიშნელობისთვის), არამედ მისი სხვადასხვანაირი ფუნქციონალური გარდაქმნა, მათ შორის დიფენერენცირება;
- სიახლეს წარმოადგენს ავტორის მიერ შემოთავაზებული ტიურინგის ვირტუალური მანქანის მოდიფიკაცია.

ნაშრომის აპრობაცია. დისერტაციაში განხილული საკითხები მოხსენებული იყო საქართველოს ტექნიკური უნივერსიტეტის ინფორმატიკისა და მართვის სისტემების ფაკულტეტის, აგრეთვე საქართველოს საპატრიარქოს წმიდა ანდრია პირველწოდებულის სახელობის ქართული უნივერსიტეტის ინფორმატიკის, მათემატიკის და საბუნებისმეტყველო მეცნიერებათა ფაკულტეტის სემინარებზე, ასევე საერთაშორისო და ადგილობრივ სამეცნიერო ფორუმებზე, რომელთა ნუსხაც თან ერთვის ავტორეფერატს. გარდა ამისა, სადოქტორო პროგრამის გეგმის შესაბამისად მომზადდა და ჩატარდა ორი თემატური სემინარი დისერტაციის ძირითადი შედეგების შესახებ.

ავტორის მიერ შემოთავაზებული დაპროგრამების სასტარტო სწავლების მეთოდიკა დანერგილია და გამოიყენება სასწავლო პროცესში სკოლა-პანსიონ "მთიებში" (2014 წლიდან) და საქართველოს საპატრიარქოს წმიდა ანდრია პირველწოდებულის სახელობის ქართულ უნივერსიტეტში (2011 წლიდან).

ნაშრომის მოცულობა და სტრუქტურა. დისერტაცია მოიცავს რეზიუმეს (ქართულ და ინგლისურ ენაზე), შესავალს, ლიტერატურის მიმოხილვას, სამ თავს, დასკვნას, გამოყენებული ლიტერატურის ნუსხას (28 დასახელება) და 2 დანართს (პროგრამული რეალიზაციების საწყისი ტექსტები). დისერტაცია მოიცავს 119 ნაბეჭდ გვერდს.

დისერტაციის შინაარსი

შესავალში ავტორი მოკლედ მიმოიხილავს პროგრამული უზრუმველყოფის განვითარების თანამედროვე მდგომარეობას და ასაბუთებს მის მიერ შერჩეული თემატიკის აქტუალობას. აქვე არის დასაბუთებული ნაშრომში განხილული საკითხების კავშირი არჩეულ თემატიკასთან.

ლიტერატურის მიმოხილვაში მოყვანილია ნაშრომში განხილული საკვანძო საკითხებისადმი მიძღვნილი პუბლიკაციების გაცნობითი აღწერა და ავტორისეული შეფასებები.

პირველი თავი მიეძღვნა ფუნქციონალურ გამოსახულებათა უფრჩხილებო (პოლონურ) ჩანაწერების შესახებ არსებული ინფორმაციის სისტემატიური ფორმით ჩამოყალიბებას, ინფიქსური ჩანაწერიდან პირდაპირ პრეფიქსულის მიღების ახალი ალგორითმის ჩამოყალიბებას, ამ ჩანაწერების მიმართ შეუღლებულის ცნების შემოღებას, შემოღებულ ცნებასთან დაკავშირებული კანონზომიერებების დადგენას და აქედან პრაქტიკული დასკვნების გამოტანას.

სისტემურ პროგრამირებაში ფართოდ გამოიყენება გამოსახულებათა ჩაწერის უფრჩხილებო ნოტაციები, რომლებსაც პოლონურ ჩანაწერებს უწოდებენ მათი ავტორის, პოლონელი მათემატიკოსის, იან ლუკასევიჩის (*Jan Łukasiewicz)* საპატივცემულოდ.

განასხვავებენ პოლონური ჩანაწერების 2 ნაირსახეობას – პოსტფიქსურს და პრეფიქსულს. პირველ შემთხვევაში ოპერაციის ნიშანი იწერება უშუალოდ შესაბამისი ოპერანდების შემდეგ, ხოლო მეორე შემთხვევაში კი უშუალოდ ოპერანდების წინ.

მაგალითად, ვთქვათ გვაქვს გამოსახულება, ჩაწერილი დაპროგრამების ენებში მიღებული სახით:

(a+b)*c-d/(e-f*g),

სადაც ლათინური ასოებით აღნიშნულია ოპერანდები. გამოსახულების ჩაწერის ასეთ ფორმას, როგორც წესი ინფიქსურს უწოდებენ (რადგან აქ 2-ადგილიანი ოპერაციების ნიშნები ოპერანდებს შორის იწერება).

ასეთ გამოსახულებას შეესაბამება შემდეგი პოსტფიქსური ჩანაწერი:

ab+c*defg*-/-

ამავე გამოსახულების პრეფიქსულ შესატყვისს კი ასეთი სახე აქვს:

-*+abc/d-e*fg

პოსტფიქსური ფორმა მოსახერხებელია გამოსახულების მნიშვნელობის გამოსათვლელად. შესაბამისი ალგორითმი გამოიყურება ასე:

1. ავიღოთ ცარიელი სტეკი;
2. განვახორციელოთ პოსტფიქსური ჩანაწერის მორიგი ელემენტის აღების (წაკითხვის) მცდელობა. თუ ეს მცდელობა წარუმატებელია, ანუ ჩანაწერი უკვე ამოიწურა, შევასრულოთ პუნქტი 5;
3. თუ წაკითხული ელემენტი წარმოადგენს ოპერანდს (ოპერანდის აღნიშვნას), მოვათავსოთ იგი სტეკში და გადავიდეთ პუქნტ 2-ზე;
4. რადგან ჩანაწერის მორიგი ელემენტი ოპერაციის ნიშანია, შევასრულოთ იგი და შედეგი მოვათავსოთ სტეკში. ამისათვის, ამოვიღოთ სტეკიდან ამ ოპერაციის შესრულებისთვის საჭირო ოპერანდების რაოდენობა (ანუ იმდენი ოპერანდი, რამდენ ადგილიანიც არის ეს ოპერაცია), შევასრულოთ ამ ოპერანდებზე (ამ ოპერანდების მნიშვნელობებზე) ხსენებული ოპერაციის შესატყვისი მოქმედება, იმის გათვალისწინებით, რომ ოპერანდების ამოღება ხდებოდა <u>შებრუნებელი მიმდევრობით</u> (ვთქვათ, გამოკლების შემთხვევაში, ჯერ მაკლები, ხოლო შემდეგ კი საკლები) და მიღებული შედეგი მოვათავსოთ სტეკში, რის შემდეგაც გადავიდეთ პუნქტ 2-ზე;
5. ავიღოთ შედეგი სტეკის სარკმელიდან, დავასრულოთ ალგორითმის შესრულება.

ავღნიშნოთ ეს ალგორითმი ასოთი A. როგორც ვხედავთ, არსად არ დაგვჭირდა არც ოპერაციების პრიორიტეტის და არც ფრჩხილების გათვალისწინება. ოპერაციებისგან დაგვჭირდა მხოლოდ უშუალოდ მათი შესრულების წესი, რომელიც, ცხადია, მოიცავს ამ ოპერაციების ოპერანდების რაოდენობას.

ზემომოყვანილი გამოსახულების ab+c*defg*-/- მიმართ ამ ალგორითმის მიყენების დროს სტეკი თანმიმდევრულად მიიღებს შემდეგ მდგომარეობებს:

წაკითხული ელემენტი	a	b	+	c	*	d	e	f	g
სტეკის მდგომარეობა	a	b a	a+b	c a+b	(a+b)*c	d (a+b)*c	e d (a+b)*c	f e d (a+b)*c	g f e d (a+b)*c

წაკითხული ელემენტი	*	–	/	–
სტეკის მდგომარეობა	f*g e d (a+b)*c	e-f*g d (a+b)*c	d/(e-f*g) (a+b)*c	(a+b)*c- d/(e-f*g)

ცხრილი 1. პოსტფიქსური გამოსახულების A ალგორითმით გამოთვლა.

რაც შეეხება გამოსახულების პრეფიქსულ ფორმას, საყოველთაოდ ცნობილია, რომ იგი მოსახერხებელია ფუნქციონალური გარდაქმნებისთვის. ნაშრომში მოყვანილია სათანადო საილუსტრაციო მასალა (გაწარმოების წესების აღწერა პრეფიქსული ნოტაციით).

აქედან შეიძლება გაკეთდეს დასკვნა, რომ გამოსახულების დამუშავების პროცესში აუცილებელია მისი ერთდროულად 2 ფორმის წარმართვა – როგორც პოსტფიქსურის, ასევე პრეფიქსულის, რადგან თითოეულ მათგანს აქვს გამოყენების პრიორიტეტული რეჟიმი. თუმცა, ნაშრომში მიიღება დასკვნა, რომ პრეფიქსული ფორმა სავსებით საკმარისია ყველა სახის დამუშავების სხარტად შესრულებისთვის.

ამ დასკვნის წმინდა პრაქტიკული ღირებულების გაზრდისთვის ყალიბდება ინფიქსურიდან პირდაპირ პოსტფიქსური გამოსახულების მიღების ახალი ალგორითმი:

1. ავიღოთ 2 ცარიელი სტეკი და მივანიჭოთ მათ ნომრები 1 და 2;
2. განვახორციელოთ ინფიქსური ჩანაწერის მორიგი ელემენტის წაკითხვის მცდელობა. თუ ეს მცდელობა წარუმატებელია, ანუ გამოსახულება უკვე ამოიწურა, მაშინ შევასრულოთ პუნქტი 8;

3. თუ წაკითხული ელემენტი ოპერანდია, მოვათავსოთ იგი №2 სტეკში და გადავიდეთ პუნქტ 2-ზე;
4. თუ წაკითხული ელემენტი გახსნილი ფრჩხილია, ჩავწეროთ იგი №1 სტეკში და გადავიდეთ პუნქტ 2-ზე;
5. თუ წაკითხული ელემენტი დახურული ფრჩხილია, მაშინ ამოვიღოთ №1 სტეკიდან ყველა ელემენტი უახლოეს გახსნილ ფრჩხილამდე ჩათვლით. ყველა ამოღებული ელემენტისთვის, გარდა გახსნილი ფრჩხილისა, ამოღების კვალობაზე შევასრულოთ პუნქტი 7 და გადავიდეთ პუნქტ 2-ზე;
6. თუ წაკითხული ელემენტი ოპერაციის ნიშანია, ან ფუნქციის აღნიშვნაა, მაშინ სანამ №1 სტეკის სარკმელში იქნება მოცემული ოპერაციის ან ფუნქციასთან შედარებით უფრო მაღალი ან ტოლი პრიორიტეტის მქონე მოქმედების აღნიშვნა, ამოვიღოთ იგი სტეკიდან და შევასრულოთ მის მიმართ პუნქტი 7, ხოლო როგორც კი სტეკის სარკმელში არ იქნება მოცემულთან შედარებით უფრო მაღალი ან ტოლი პრიორიტეტის მქონე მოქმედების აღნიშვნა (იმ შემთხვევების ჩათვლით, როცა სტეკი ცარიელია, ან სტეკის სარკმელში გახსნილი ფრჩხილია), მოცემული მოქმედების აღნიშვნა მოვათავსოთ №1 სტეკში და გადავიდეთ პუნქტ 2-ზე;
7. მოცემული მოქმედების ნიშნის მიხედვით შევასრულოთ საჭირო ოპერანდების რაოდენობის ტოლი წაკითხვები №2 სტეკიდან, ამოღებულ ელემენტებს წინ მივუერთოთ ეს მოქმედების ნიშანი იმის გათვალისწინებით, რომ ოპერანდების ამოღება ხდებოდა პირუკუ მიმდევრობით და მიღებული გამოსახულება მოვათავსოთ №2 სტეკში, რის შემდეგ დავბრუნდეთ ალგორითმის იმ წერტილში, საიდანაც მოხდა მიმართვა პუნქტ 7-ზე;
8. ამოვიღოთ №1 სტეკიდან ყველა დარჩენილი ელემენტი და ამოღების კვალობაზე შევასრულოთ მათ მიმართ პუნქტი 7. №2 სტეკის სარკმელში მიღებული გამოსახულება ჩავთვალოთ შედეგად და დავასრულოთ ალგორითმის შესრულება.

წაკითხული ელემენტი	(	a	+	b	)	*	c	-	d	/
№1 სტეკის მდგომარეობა	(	(	+ (	+ (		*	*	-	-	/ -
№2 სტეკის მდგომარეობა		a	a	b a	+ab	+ab	c +ab	*+abc	d *+abc	d *+abc

წაკითხული ელემენტი	(	e	-	f	*	g
№1 სტეკის მდგომარეობა	(/ -	(/ -	- (/ -	- (/ -	* - (/ -	* - (/ -
№2 სტეკის მდგომარეობა	d *+abc	e d *+abc	e d *+abc	f e d *+abc	f e d *+abc	g f e d *+abc

წაკითხული ელემენტი	)				┤(გამოსახულება ამოიწურა)	
№1 სტეკის მდგომარეობა	* - (/ -	- (/ -	(/ -	/ -	-	
№2 სტეკის მდგომარეობა	g f e d *+abc	*fg e d *+abc	-e*fg d *+abc	-e*fg d *+abc	/d-e*fg *+abc	-*+abc/d-e*fg

ცხრილი 2. ინფიქსური გამოსახულებიდან პირდაპირ პრეფიქსულის მიღება.

ცხრილი 2 იძლევა ამ ალგორითმის გამოყენების ილუსტრაციას ჩვენს მიერ განხილული გამოსახულების მიმართ.

ზემონახსენები დასკვნა ავტორის მიერ გამომუშავდა მთელი რიგი წმინდა თეორიული შედეგების საფუძველზე. ამ შედეგებს სათავე დაუდო ავტორის მიერვე შემოღებულმა შეუღლებულის ცნებამ.

განსაზღვრა 1. A–ალგორითმის **შეუღლებული** ვუწოდოთ ალგორითმს, რომელიც ასევე მიეყენება პოსტფიქსურ ჩანაწერებს და მიიღება A–ალგორითმიდან, თუ პუნქტ 4-ში ხაზგასმულ ტექსტს ("შებრუნებული მიმდევრობით") შევცვლით ტექსტით "პირდაპირი მიმდევრობით".

A–ალგორითმის შეუღლებული ავღნიშნოთ სიმბოლოთი A*. შევნიშნოთ, რომ განსაზღვრიდან პირდაპირ გამომდინარეობს შემდეგი:

- ალგორითმებს A და A* იდენტური სირთულე აქვთ;
- (A*)* = A, ანუ შეუღლებულობა ურთიერთშებრუნებული ყოფილა.

განსაზღვრა 2. თუ v არის პოსტფიქსური გამოსახულება, მაშინ მისი **შეუღლებული** ვუწოდოთ ისეთ w გამოსახულებას, რომ

$$A^*(w) = A(v)$$

თუ w არის v –ს შეუღლებული, მაშინ ჩავწერთ:

$$w = v^*$$

ავღნიშნოთ სიმბოლოთი R (Reverse - შებრუნებული) სასრული მიმდევრობის შებრუნების (რევერსირების) გარდაქმნა, ხოლო სიმბოლოთი P ავღნიშნოთ გარაქმნა პოსტფიქსურიდან პრეფიქსულში.

დისერტაციაში მკაცრად მტკიცდება, მაგრამ აქ დამტკიცების გარეშე მოვიყვანოთ, შემდეგი დებულებები.

თეორემა 1. ნებისმიერი პოსტფიქსური v ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$C(v) = R(P(v))$$

შედეგი 1.1. ნებისმიერი პოსტფიქსური v ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$P(v) = R(C(v))$$

შედეგი 1.2. ნებისმიერი პრეფიქსული x ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$P^{-1}(x) = C(R(x))$$

ბოლო შედეგიდან პირდაპირ ვასკვნით, რომ პრეფიქსული გამოსახულების შებრუნებული (რევერსირებული) წარმოადგენს მისი შესატყვისი პოსტფიქსურის შეუღლებულს. და რადგან განმარტებით, პოსტფიქსურის შეუღლებულის მნიშვნელობის გამოთვლა ხდება ალგორითმით A* (A-ს შეუღლებულით) ვღებულობთ, რომ **პრეფიქსული გამოსახულების მნიშვნელობის გამოსათვლელად იგი უნდა დამუშავდეს მარჯვნიდან მარცხნივ ალგორითმით A***. ეხლა თუ გავითვალისწინებთ იმას, რომ ალგორითმი A* არაფრით არ ჩამოუვარდება ალგორითმ A-ს, ვაკეთებთ საბოლოოდ ზემოდაპირებულ დასკვნას იმის თაობაზე, რომ **გამოსახულების პრეფიქსული ფორმა სავსებით გაწვდება არა მხოლოდ ფუნქციონალურ გარდაქმნებს, არამედ გამოსახულების მნიშვნელობის გამოთვლასაც**.

შემდეგ მტკიცდება დებულებები.

შედეგი 1.3. თუ v - პოსტფიქსური ჩანაწერია, მაშინ:

$$R(v) = P(v^*)$$

შედეგი 1.4. თუ x – პრეფიქსული ჩანაწერია, მაშინ:

$$R(x) = (P^{-1}(x))^*$$

თეორემა 2. ნებისმიერი პოსტფიქსური v ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$C(v) = P^{-1}(R(v))$$

ამის შემდეგ ავტორს შემოაქვს შეუღლებულის ცნება პრეფიქსული გამოსახულების მიმართ.

განსაზღვრა 3. თუ x არის რაიმე გამოსახულების პრეფიქსული ჩანაწერი, მაშინ მისი **შეუღლებული** ვუწოდოთ ისეთ y გამოსახულებას, რომ

$$C(P^{-1}(y)) = P^{-1}(x)$$

პირდაპირ განმარტებიდან გამომდინარეობს, რომ პრეფიქსულის შეუღლებული ისევ პრეფიქსული ჩანაწერია.

როგორც მოსალოდნელი იყო, მტკიცდება ადრე დამტკიცებული დებულებების სარკისებური ანალოგები.

თეორემა 3. ნებისმიერი პრეფიქსული x ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$C(x) = R(P^{-1}(x))$$

თეორემა 4. ნებისმიერი პრეფიქსული x ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$C(x) = P(R(x))$$

შედეგი 2.1. ნებისმიერი პრეფიქსული x ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$P^{-1}(x) = R(C(x))$$

შედეგი 2.2. ნებისმიერი პოსტფიქსური v ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$P(v) = C(R(v))$$

შედეგი 2.3. ნებისმიერი პრეფიქსული x ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$R(x) = P^{-1}(x^*)$$

შედეგი 2.4. ნებისმიერი პოსტფიქსური v ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$R(v) = (P(v))^*$$

შედეგად ყალიბდება დასკვნა, რომ **პოსტფიქსური ჩანაწერის დამუშავება მარჯვნიდან მარცხნივ, ტოლძალოვანია მისი პრეფიქსული შესატყვისის შეუღლებულის დამუშავებისა**.

ამის შემდეგ მტკიცდება უფრო ზოგადი დებულებები.

თეორემა 5. თუ g არის ნებისმიერი გამოსახულების უფრჩხილებო ფორმა (ან პოსტფიქსური, ან პრეფიქსული), მაშინ ადგილი აქვს შემდეგ თანაფარდობას:

$$C(R(g)) = R(C(g))$$

ეხლა თუ P–ს ქვეშ ვიგულისხმებთ განზოგადოებულ გარდაქმნას, რომელიც პრეფიქსულიდან გვაძლევს პოსტფიქსურს, ხოლო პოსტფიქსურიდან – პირიქით, პრეფიქსულს, მტკიცდება შემდეგი.

თეორემა 6. თუ g არის ნებისმიერი გამოსახულების უფრჩხილებო ფორმა (ან პოსტფიქსური, ან პრეფიქსული), მაშინ მისი შეუღლებულის მისაღებად შეიძლება ან ჯერ მისი შებრუნება და შემდეგ მიღებულისთვის მეორე უფრჩხილებო შესატყვისის მიღება, ან ჯერ საწყისი ჩანაწერის მეორე უფრჩხილებო ფორმის მიღება და შემდეგ კი შებრუნება. ანუ ადგილი აქვს თანაფარდობებს:

$$C(g) = R(P(g))$$

$$C(g) = P(R(g))$$

თეორემა 7. ნებისმიერი უფრჩხილებო g ჩანაწერისთვის ადგილი აქვს თანაფარდობას:

$$R(P(g)) = P(R(g))$$

ანუ, **გარდაქმნა R აღმოჩნდა გადაადგილებადი როგორც გარდაქმნა C–ს მიმართ, ასევე განზოგადებული P გარდაქმნის მიმართაც.**

ამ თავში მიღებული შედეგიდან ზოგიერთი გამოყენებულ იქნა მომდევნო მასალაში, ზოგიერთი კი სამომავლოდ ელოდება გამოყენებას.

თავი 2 მიეძღვნა ფუნქციონალურ გამოსახულებათა დამუშავების ავტომატიზებისთვის განკუთვნილი პროგრამების შემუშავებას, სადაც განსაკუთრებული ადგილი დაეთმო სიმბოლურ დიფერენცირებას.

სიმბოლური დიფერენცირების პროგრამა შესრულდა მხოლოდ 1 ცვლადის ფუნქციებისთვის, რომლებშიც შეიძლება იყოს გამოყენებული 4-ვე არითმეტიკული ოპერაცია, ახარისხება (აღნიშნული "^"-ით) და შემდეგი ელემენტარული ფუნქციები: sin, cos, tg, ctg, arcsin, arcos, arctg, arcctd, sqr, sqrt, exp, ln, lg, s.

აქ ერთპარამეტრიანი ფუნქცია დასახელებით s, (შემოკლება სიტყვისა "signed" - ნიშანი) შემდეგნაირად განისაზღვრება: $s(x) = -x$.

გაწარმოების წესების შესწავლამ მოგვცა შესაძლებლობა გავაკეთოთ შემდეგი დასკვნები ამ წესებით განსაზღვრული მოქმედებების ზოგადი დახასიათებისთვის:

1. გაწარმოების ყოველ ნაბიჯზე აუცილებელია შეგვეძლოს გამოვყოთ ის მოქმედება (ფუნქცია) რომლის მიმართაც ხდება გაწარმოება. ცხადია, ლაპარაკია ყველაზე გარე მოქმედებაზე, ანუ მოქმედებაზე, რომელიც შეიძლება შესრულდეს აღნიშნული გამოსახულების გამოთვლის ბოლოს. ამით ხდება იმ წესის დადგენა, რომელიც აქტუალურია მოცემული ნაბიჯისთვის;

2. აუცილებელია შეგვეძლოს გამოვყოთ აქტუალური მოქმედების (ფუნქციის) ყველა პარამეტრი;

3. როდესაც ცნობილია აქტუალური მოქმედება (ფუნქცია) და მისი ყველა პარამეტრი, საკუთრივ გაწარმოების წესის ჩამოყალიბება შესაძლებელია ამ პარამეტრებისა და მათი წარმოებულების მიმართ

გარკვეული გამოსახულების ტერმინებში. ცხადია, იგივე წესები შეიძლება იყოს გამოყენებული თავის მხრივ ამ პარამეტრების გაწარმოებისთვის – გაწარმოების წესები ბუნებით რეკურსიულია.

ყოველივე ზემოთქმული გვკარნახობს, რომ პროგრამული რეალიზაციის გაადვილების მიზნით მიზანშეწონილია შემუშავდეს შიდა ენა, რომელზედაც ჩამოყალიბდება გაწარმოების წესები. თითოეული წესი უნდა შეესაბამებოდეს ერთ მოქმედებას (ფუნქციას) და ქონდეს სათანადო პრეფიქსული ფუნქციონალური გამოსახულების ფორმის თარგის სახე, რომელშიც გამოიყენება მოქმედებათა (ფუნქციათა) აღნიშვნები, აგრეთვე ზოგიერთი კონსტანტები და გასაწარმოებელი გამოსახულების პარამეტრები და მათი წარმოებულები. აქ ძალიან არსებითია, რომ ეს თარგები იყოს შეძლებისდაგვარად თვალსაჩინო.

გასაწარმეობელი გამოსახულება		შესაბამისი წარმოებული შიდა ენაზე
პრეფიქსული სახით	შიდა ენაზე	
+uv	+ab	+AB
-uv	-ab	-AB
*uv	*ab	+*Ab*aB
/uv	/ab	/-*Ab*aB*bb
^uv	^ab	+**Ab^a-b1**A^abqa
sin u	fa	*gaA
cos u	ga	s*faA
tg u	ha	/Anga
ctg u	ia	s-Anfa
arcsin u	ja	/Ao-1na
arcos u	ka	s/Ao-1na
arctg u	ℓa	/A+1na
arcctd u	ma	s/A+1na
sqr u	na	*2*aA
sqrt u	oa	/A*2oa
exp u	pa	*paA
ℓn u	qa	/Aa

ɬg u	ta	/A*aqD
s u	sa	sA

ცხრილი 3. გაწარმოების წესები შიდა ენაზე.

იმისათვის, რომ გასაგები გახდეს ამ ენაზე ჩამოყალიბებული ზემომოყვანილი გაწარმოების წესები, საკმარისია ითქვას შემდეგი:

1. ამ ფორმულებში არითმეტიკულ მოქმედებათა ნიშნები უცლელადაა გადმოტანილი, ხოლო ახარისხება აღინიშნება სიმბოლოთი ‘^’;
2. ფუნქციები გადანომრილია მცირე ზომის ლათინური ასოებით, დაწყებული ‘f’-ით;
3. აქტუალური მოქმედების პირველი პარამეტრი აღინიშნება სიმბოლოთი ‘a’, ხოლო მეორე პარამეტრი კი სიმბოლოთი ‘b’. მათი წარმოებულები კი აღინიშნება შესაბამისად ‘A’ და ‘B’-თი.

დაგვრჩა შევნიშნოთ, რომ ბოლოსწინა ფორმულაში სიმბოლოთი ‘D’ აღნიშნულია რიცხვი 10. გარდა ამისა, s-ით აღნიშნულია ფუნქცია, რომელიც დააბრუნებს მნიშნელობას, მიღებულს პარამეტრიდან მისი ნიშნის შეცვლით.

ვინადიან პრეფიქსული ჩანაწერი ფაქტობრივად წარმოადგენს მიმდევრობას, რომლის თითოეული წევრი არის ან მოქმედება (ფუნქცია) ან ოპერანდი, ბუნებრივად გამოიყურება ეს მიმდევრობა მოთავსდეს ვექტორში, რომლის თითოეული ელემენტი ან მოქმედებაა ან ოპერანდი. აქედან გამომდინარე, ვექტორის ელემენტის ტიპად შეირჩა შემდეგი სტრუქტურა:

```
struct Telem
{char operand; // 0 ნიშნავს მოქმედებას, 1 ცვლადს, ხოლო 2 რიცხვს
 union { char nomer; // მოქმედებისთვის აღნიშნავს მის ნომერს
       double mnish; // რიცხვისთვის – მისი მნიშვნელობაა
       }
};
```

ამის შემდეგ პროგრამული უზრუნველყოფის შემუშავებისთვის გვრჩება შემდეგი მოქმედებების რეალიზება:

1. გამოსახულების საკვანძო (რომელიც ბოლოს სრულდება) მოქმედების დადგენა. პრეფიქსულ ნოტაციაში ეს უბრალოდ ნიშნავს გამოსახულების საწყისი ელემენტის გარკვევას;
2. თუ საწყისი ელემენტი არ არის მოქმედება (ფუნქცია), მაშინ პასუხის დაბრუნება (1, თუ ცვლადია და 0, თუ კონსტანტაა). ხოლო წინააღმდეგ შემთხვევაში კი შესაბამისი ფორმულის შემცველი სტრიქონის გააქტიურება;
3. მიგნებული ფორმულის მიხედვით პასუხის აგება, რისთვისაც აუცილებელია იქნება საჭირო რაოდენობის პარამეტრების ამოღება და მათი ჩასმა პასუხში ფორმულით განსაზღვრულ ადგილზე, შესაძლოა, გაწარმოების ფუნქციით ზემოქმედების შემდეგ (საკუთარი თავის რეკურსიული გამოძახებით);

საბოლოო პასუხის მიღების წინ სასურველია გამოსახულების გამარტივების პროცედურის (მეთოდის) გამოძახება.

რეალიზების გაადვილებისთვის შემოღებულ იქნა C++ ის სტანდარტული vector – ის მემკვიდრე Vector შემდეგნაირად:

template <class T>

class Vector : public vector <T>

მემკვიდრეს დაემატა კონკატენაციის ოპერაცია, რომელიც აღინიშნა “|”-თი და მეთოდი SubVec, რომლის დანიშნულებაა ქვევექტორის დაბრუნება სრული ანალოგიით იმასთან, რასაც გვაძლევს ქვესტრიქონების მიღებისთვის სტანდარტული მეთოდი Substr.

რეალიზაიამ, რომელიც შესრულდა C++ –ზე მოიცვა შემდეგი შესაძლებლობები:

Tgam Togam(string x, int &res) –

გამოსახულების გადაყვანა ინფიქსურიდან (სტრიქონიდან) შიდა სახეზე (ტიპი **Tgam**). შეცდომის შემთხვევაში მეორე პარამეტრი იძლევა შესაბამისი პოზიციის ნომერს, თუ არა და იგი დააბრუნებს (-1)-ს.

string Tostring(Tgam x) –

გამოსახულების გადაყვანა შიდა სახიდან ინფიქსურზე.

double Gamotvla(Tgam G, double x=1) –

პირველი პარამეტრით მოცემული გამოსახულების გამოთვლა მეორე პარამეტრით განსაზღვრული ცვლადის მნიშვნელობისთვის.

Tgam Dif(Tgam x) –

კორექტული პრეფიქსული გამოსახულებიდან (პირველი პარამეტრი) მისი წარმოებულის შესატყვისი პრეფიქსული გამოსახულების დაბრუნება უშუალოდ ზემომოყვანილ ფორმულებზე დაყრდნობით.

Tgam Gamartiveba(Tgam G) –

ეს ფუნქცია უზრუნველყოფს პრეფიქსული გამოსახულების გამარტივებას.

რეალიზაციის დეტალები ამ თავში დაწვრილებით არის განხილული.

რის შემდეგაც ავტორს მოყავს ფუნქციონალურ გამოსახულებებთან მომხმარებლის მუშაობის მაუზრუნველყოფი კლასის პროექტის (ინტერფეისული ნაწილი) აღწერა სათანადო კომენტარებით.

შემდგომ ამავე თავში აღიწერება პასკალზე დაწერილი მოდული, რომელიც უზრუნველყოფს მრავალი ცვლადის შემცველი გამოსახულების დამუშავებას.

აქ გამოსახულების შიდა წარმოდგენა პოსტფიქსურია და პროგრამაში შეესაბამება ბმული სია. მოდულით გათვალისწინებულია 2 პროგრამული კომპონენტი. პირველი მათგანი წარმოადგენს პროცედურას და მისი სათაურია:

procedure scanfunq(x:string; var y : mimt; var mistake, npoz : byte; var masind : masindp);

აქ პირველი პარამეტრი წარმოადგენს საწყის გამოსახულებას, მოცემულს ინფიქსური სახით სტრიქონის მეშვეობით. მეორე პარამეტრი წარმოადგენს საწყისი გამოსახულების პოსტფიქსურ შესატყვისს, ბოლო პარამეტრი იძლევა ინფორმაციას იმ ცვლადების შესახებ, რომლებსაც იყენებს პროცედურა. შეცდომის შემთხვევაში mistake იძლევა შეცდომის მიზეზის ნომერს, ხოლო npoz კი დააბრუნებს პოზიციის ნომერს, რომლისთვისაც გარკვეული გახდა, რომ საწყისი გამოსახულების შემდგომი დამუშავება აზრს მოკლებულია.

მოდულის მეორე პროგრამული კომპონენტი არის ფუნქცია სათაურით:

function gam(x:mimt; y:maspar):real;

ფუნქცია შიდა სახით მოცემული პირველი პარამეტრისთვის გამოითვლის გამოსახულების მნიშვნელობას მეორე პარამეტრით განსაზღვრული ცვლადების მნიშვნელობებისთვის (მეორე პარამეტრიც

ნამდვილი ტიპის ელემენტებიანი მასივია, რომელშიც ყოველ ელემენტს ზუსტად ერთი შესაძლო ცვლადი შეესაბამება).

ამ მოდულში საწყის გამოსახულებაში დასაშვებია წინასწარ განსაზღვრული რაოდენობის ცვლადების გამოყენება. აგრეთვე შესაძლებელია 4-ვე არითმეტიკული მოქმედების, 2 სახის ახარისხების (ცალკე მთელ ხარისხად და ცალკე წილად ხარისხად, შესაბამისად ნიშნებით # და ^) და შემდეგი ელემენტარული ფუნქციების გამოყენება **sin, cos, tg, arcsin, arccos , arctg, abs, sqr, sqrt, exp, ln, lg .**

ამავე მოდულში რეალიზებულია 3 ოპერანდიანი მოქმედება. ნაშრომში დაწვრილებით არის განხილული ამ მოდულის რეალიზაცია.

თავი 3 მიეძღვნა ტიურინგის ვირტუალური მანქანის მოდიფიკაციას, განკუთვნილს გამოყენებისთვის სასწავლო პროცესში. სწავლებისთვის განკუთვნილი ტიურინგის ვირტუალური მანქანის მოდიფიკაცია შემუშავდა შემდეგ მოსაზრებებზე დაყრდნობით:

1. შენარჩუნებული ყოფილიყო ტიურინგის მანქანის ფუძემდებელი ნაწილები - წრფივი ლენტა, მიმდინარე პოზიციის განმსაზღვრელი თავაკი, და მდგომარეობათა ცხრილის ფაქტობრივი ექვივალენტი.
2. შენარჩუნებულ ყოფილიყო ალგორითმის გამომსახველობით საშუალებათა მინიმალურობა მათი უნივერსალობის პირობებში, თუკი ლენტი უსასრულოა.
3. მდგომარეობათა ცხრილის ექვივალენტის როლი ეტვირთა პროგრამას, დაწერილს შემუშავებული ტიურინგის მანქანის მოდიფიკაციის ბრძანებათა ტერმინებში, მინიმალური დამატებითი საშუალებების ჩართვის პირობებში.
4. რეალიზაცია ყოფილიყო ადვილი გამოყენებისთვის სასწავლო პროცესის სხვადასხვა კომპონენტის ინტერესებიდან გამომდინარე.
5. რეალიზაცია ყოფილიყო მაქსიმალურად თვალსაჩინო.

მოცემულ ნაშრომში წარმოდგენილი ტიურინგის ვირტუალური მანქანის მოდიფიკაცია შედგება 2048 უჯრედიანი წრფივი ლენტისგან, ნიშნულისგან (თავაკისგან) რომელიც მიუთითებს ამ ლენტის მიმდინარე პოზიციას, და იმართება სულ 6 ბრძანებით.

ეს ბრძანებებია:

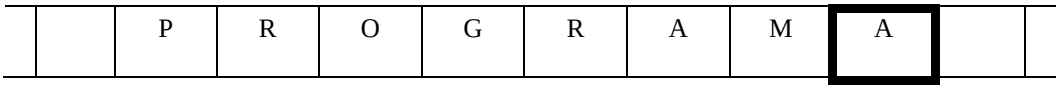
1. **L** - თავაკის გადაადგილება ერთი პოზიციით მარცხნივ;
2. **R** - თავაკის გადაადგილება ერთი პოზიციით მარჯვნივ;
3. **S** - პროგრამის შესრულების დამთავრება (ტიურინგის ვირტუალური მანქანის გაჩერება);
4. **Wს** - ლენტის მიმდინარე პოზიციაში **ს** სიმბოლოს ჩაწერა;
5. **Gჯ** - უპირობო გადასვლა **ჯ** ჯდეზე, ანუ **ჯ** ნიშნულის მქონე ბრძანებაზე;
6. **Ისჯ** - გადასვლა **ჯ** ჯდეზე, იმ შემთხვევაში, თუ ლენტის მიმდინარე პოზიციაში მოთავსებულია სიმბოლო **ს**.

თითოეული ბრძანება იწერება ცალკე სტრიქონში. ასევე ცალკე სტრიქონში იწერება **ჯდე**, რომელიც **გარეგნულად წარმოადგენს არა უმეტეს 5 ნიშნა ათობით მთელ რიცხვს**. **ჯდე იწერება სტრიქონში**, რომელიც მოთავსებულია **უშუალოდ მოსანიშნი ოპერატორის წინ**.
ამის შემდეგ ნაშრომში განიხილება ამ მანქანაზე შესასრულებელი ამოცანები აღმავალი სირთულით. მათგან ყველაზე მარტივია შემდეგი.
ამოცანა 1. სიმბოლოთა მიმდევრობის მარჯვენა კიდეზე გადასვლა, თუ თავაკი ამ მიმდევრობაზეა.
ლენტის სასტარტო მდგომარეობა. ლენტზე მოთავსებულია სიმბოლოთა ნებისმიერი არაცარიელი მიმდევრობა, რომელიც არ შეიცავს ჰარებს, ამ მიმდევრობისგან მარცხნივ და მარჯვნივ ყველა პოზიცია შევსებულია ჰარებით (ჩვენ გთავაზობთ სიტყვა ”ჰარი” გამოვიყენოთ იმ სიმბოლოს აღნიშვნისთვის, რასაც ინგლისურად ეწოდება ”space”, ხოლო რუსულად კი ”пробел”; ”ჰარი” სვანური წარმოშობის სიტყვაა და აღნიშნავს თავისუფალ სივრცეს 2 მოხნულ ყანას შორის).
თავაკის სასტარტო მდგომარეობა. თავაკი განთავსებულია მიმდევრობის მიერ დაკავებული სივრცის შიგნით (ანუ ამ მიმდევრობის ნებისმიერ სიმბოლოზე).
ლენტის დასკვნითი მდგომარეობა. იგივეა, რაც სასტარტო.
თავაკის დასკვნითი მდგომარეობა. თავაკი უნდა განთავსდეს სიმბოლოთა საწყისი მიმდევრობის მარჯვენა კიდეზე.

მაგალითი 1.

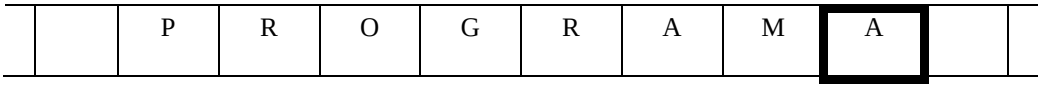
საწყისი მდგომარეობა:

დასკვნითი მდგომარეობა:

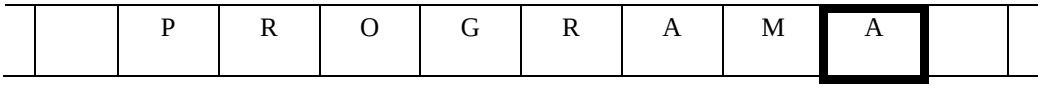


მაგალითი 2.

საწყისი მდგომარეობა:



დასკვნითი მდგომარეობა:



შევნიშნოთ, რომ ტიურინგის ვირტუალურ მანქანაზე ამოსახსნელი ყოველი ამოცანის პირობაში უნდა განისაზღვროს ლენტის და თავაკის სასტარტო მდგომარეობა, აგრეთვე ამავე ლენტის და თავაკის დასკვნითი მდგომარეობა.

აქ ამოხსნა ასე გამოიყურება.

10	20
R	L
I 20	S
G10	

პროგრამის ტექსტი 2 მწკრივად მოგვყავს – ჯერ მარცხენა სრულდება. ნაშრომში ამას მოყვება ამოცანა, რომელიც ოდნავ აღემატება სირთულით განხილულს. და უკვე ამ ამოცანის განხილვა გვიჩენებს ისეთი შეცდომის დაშვების საშიშროებას, როგორიცაა ჩაციკლვა.

ამოცანა 2. სიმბოლოთა მიმდევრობის მარჯვენა კიდეზე გადასვლა, თუ თავაკი თავდაპირველად მიმდევრობაზეა ან მის მარცხნივ.

აქ მოვიყვანთ ამოხნას, რომელიც შეცდომას შეიცავს:

5	I 20
R	G10
I 5	20

10	L
R	S

ეს პროგრამა არასწორად იმუშავებს, თუ ვირტუალური მანქანის სასტარტო მდგომარეობა აკმაყოფილებს დასკვნით პირობას. ამ შემთხვევაში ადგილი ექნება **ჩაციკლვას**.

აქვე შემოთავაზებულია სწორი ამოხსნის გზები. დიახ, ასეთი მარტივი ამოცანაც კი იძლევა შესაძლო ამოხსნათა მრავალფეროვნების ილუსტრირების შესაძლებლობას, რაც აუცილებელია მოსწავლისთვის.

შემდგომ განხილული მაგალითები რთულდება, ხდება განშტოებისა დ ციკლის გაცნობა. რის შემდეგაც ნაშრომში გვამცნობენ სასწავლო მანქანის ასემბლერის გაფართოებას, რომლის იდეა ეკუთვნის ავტორის მოსწავლეს, ბაკალავრიატის სტუდენტ **სანდრო ბარნაბიშვილს.**

ამ გადაწყვეტით მოხდა გადასვლის ოპერაციების (G და I) გაფართოება მათზე დამატებით დაბრუნების მოთხოვნის დაკისრებით, თუ კი ბრძანება დასრულდებოდა სიმბოლოთი R. ასეთ პირობებში გარეგნულად არშეცვლილ ბრძანებას S დაეკისრა ყველაზე ბოლოს ჩამოყალიბებული დაბრუნების მოთხოვნის განხორციელება, თუ კი მისი შესრულების მომენტისთვის ასეთი ერთი მაინც დაუკმაყოფილებელი მოთხოვნა იარსებებდა. წინააღმდეგ შემთხვევაში ეს ბრძანება ჩვეულებრივ ამთავრებს დავალების შესრულებას.

ფაქტობრივად, **ამ გაფართოებამ სასწვლო მანქანის ასემბლერს შემატა ქვეპროგრამის შესაძლებლობა** და აგრეთვე ამან **გახადა შესაძლებელი აღიწეროს რეკურსიული ალგორითმები**.

ნაშრომში განიხილება თქმულის საილუსტრაციო რამდენიმე მაგალითი. აქ მოვიყვანოთ შემდეგი.

ამოცანა 3. ფრჩხილებიანი მიმდევრობის კორექტულობის შემოწმება. ფრჩხილებიანი მიმდევრობის კორექტულობა გაგებულია ჩვეულებრივი აზრით. კონკრეტულად, მრგვალი ფრჩხილებისგან შედგენილ მიმდევრობას ვუწოდებთ კორექტულს (კფმ), თუ იგი აკმაყოფილებს პირობას:

<კფმ>::= | (<კფმ>)

ამ განმარტებით ცარიელი მიმდევრობაც ითვლება კორექტულ მიმდევრობად, მაგრამ ჩვენი ამოცანის პირობით მოცემული მიმდევრობა ცარიელი არ შეიძლება იყოს.

ლენტის სასტარტო მდგომარეობა. ლენტზე განთავსებულია მხოლოდ მრგვალი ფრჩხილებისგან შედგენილი არაცარიელი მიმდევრობა. ლენტის ყველა დარჩენილი პოზიცია შევსებულია ჰარებით.

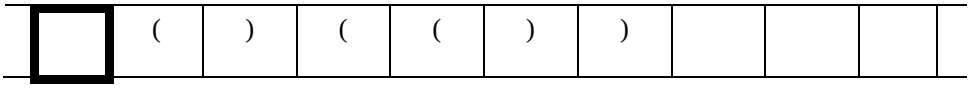
თავაკის სასტარტო მდგომარეობა. თავაკი თავიდან დგას ფრჩხილებისგან შედგენილი მიმდევრობის მარცხენა კიდის მარხენა მეზობელ პოზიციაზე.

ლენტის დასკვნითი მდგომარეობა. დასკვნითი მდგომარეობა იდენტურია სასტარტოსი.

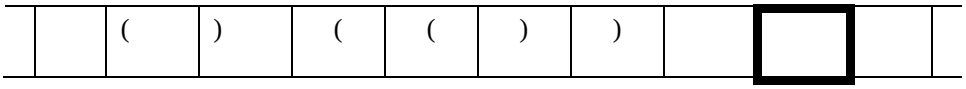
თავაკის დასკვნითი მდგომარეობა. თუ მიმდევრობა კორექტულია, თავაკი უნდა განთავსდეს მიმდევრობის მარჯვენა კიდესგან 2 პოზიციით მარჯვნივ, თუ არა და ნებისმიერ სხვა მდგომარეობაში.

მაგალითი 1.

საწყისი მდგომარეობა:



დასკვნითი მდგომარეობა:



პროგრამა, რომელიც წყვეტს ამ ამოცანას შემდეგნაირად გამოიყურება:

10	I(20
R	R
20	S
I(10R	

ვფიქრობ უდავია, რომ პროგრამა არც ისე ტრივიალურია, თუმცა ზომით საკმაოდ პატარაა.

მოცემული მომენტისთვის არსებობს ტიურინგის სასწავლო მანქანის 2 რეალიზაცია. ორივე შესრულებულია დაპროგრამების ენაზე C# ჩემი ხელმძღვანელობით ავთანდილ რუხაძის და სანდრო ბარნაბიშვილის მიერ. მიმდინარეობს ამ მეთოდიკის შემდგომი დახვეწა.

დასკვნა

წარმოდგენილი ნაშრომის ფარგლებში ჩატარებული კვლევითი სამუშაოს შედეგების შეჯამების სახით შეიძლება შემდეგი დავასკვნათ:

1. შესწავლილია და სისტემაშია მოყვანილი ფუნქციონალურ გამოსახულებათა კომპიუტერში წარმოდგენის ფორმები და მათ დამუშავებასთან დაკავშირებული ალგორითმები;
2. შემუშავებულია ინფიქსური გამოსახულებიდან პრეფიქსული გამოსახულების მიღების ეფექტიანი პირდაპირი ალგორითმი, დამყარებული 2 სტეკის გამოყენების იდეაზე;
3. შემოთავაზებულია ფუნქციონალურ გამოსახულებათა კომპიუტერში წარმოდგენის ახალი ფორმები, შესწავლილია ამ ფორმების თვისებები, მკაცრად მათემატიკურად არის დამტკიცებული შემოთავაზებული და სტანდარტული ფორმების დამაკავშირებელი კანონზომიერებები;
4. მიღებული თეორიული შედეგებიდან გამოტანილია პრაქტიკული მნიშვნელობის მქონე მთელი რიგი დასკვნები, რის საფუძველზეც შემუშავდა ფუნქციონალურ გამოსახულებათა დამუშავების მეთოდიკა, რომელიც მოითხოვს გამოსახულების მხოლოდ პრეფიქსული ფორმის გამოყენებას, როგორც წმინდა ფუნქციონალური გარდამქნებისას, ასევე გამოსახულების რიცხვითი დამუშავების დროს;
5. მკაცრად მათემატიკურად დამტკიცდა, რომ ზუსტად იგივე შეიძლებოდა გაკეთებულიყო გამოსახულებათა მხოლოდ პოსტფიქსურ ფორმაზე დაყრდნობით და ამ თვალსაზრისით, როგორც მოსალოდნელი იყო, უფრჩხილებო ნოტაციის ეს ორი ფორმა თავისი შესაძლებლობებით ტოლძალოვანია;
6. შეიქმნა შემუშავებულ მეთოდიკაზე დამყარებული პროგრამული უზრუნველყოფა;
7. დიფერენცირების გარდაქმნისთვის შემუშავდა რეალიზაცია, რომელმაც მთლიანად დაფარა სიმბოლური დიფერენცირება გაწარმოების წესებს დაქვემდებარებული ერთი ცვლადის ყველა ფუნქციისთვის;

8. შემუშავდა მრავალი ცვლადის ფუნქციონალურ გამოსახულებათა დამუშავებისთვის განკუთვნილი პროგრამული უზრუნველყოფა, რომელშიც ჯერ არ არის გათვალისწინებული სიმბოლური დიფერენცირება;
9. შემუშავდა დაპროგრამების სასტარტო სწავლებისთვის განკუთვნილი ტიურინგის ვირტუალური მანქანის მოდიფიკაცია, რომელიც მიყვანილ იქნა კომპიუტერულ რეალიზაციამდე;
10. შესწავლილი იქნა შემუშავებული რეალიზაციის სასწავლო პროცესში გამოყენებასთან დაკავშირებული საკითხების კომპლექსი;
11. სწავლების ზემონახსენები მეთოდიკა დაინერგა სასწავლო პროცესში;
12. ნაშრომის ფარგლებში შესრულებულმა სამუშაოებმა გამოამზეურა ახალი საკვლევი თემები და წინაპირობები შექმნა სამომავლოდ შემდეგი სამუშაოების წარმოებისთვის:
    12.1. გამოკვლეულ იქნას გამოყენების პერსპექტივის ჭრილში ის ფაქტი, რომ ყველა სახის უფრჩხილებო ჩანაწერი წარმოადგენს გრაფის (ხის) სახით მოცემული ინფორმაციის ალგებრულ წარმოდგენას;
    12.2. დაიხვეწოს და, საჭიროების შემთხვევაში, შეივსოს ავტორის მიერ შემოთავაზებული ფუნქციონალურ გამოსახულებათა დამუშავებისთვის განკუთვნილი კლასის პროექტი და განხორციელდეს მისი რეალიზაცია;
    12.3. შემუშავდეს რამდენიმე ცვლადის შემცველი ფუნქციონალური გამოსახულებების დამუშავებისთვის განკუთვნილი პროგრამული რეალიზაცია, რომელიც მოიცავს კერძო წარმოებულების მიღებას;
    12.4. გამოკვლეულ იქნას გამოსახულებათა შემოკლებისთვის განკუთვნილი უფრო მძლავრი საშუალებების შემუშავების შესაძლებლობა;
    12.5. ავტორის მიერ შემოთავაზებული მიდგომა გამოყენებულ იქნას გაწარმოების შესწავლისთვის განკუთვნილი სასწავლო პროგრამის შესაქმნელად;

12.6. მოისინჯოს მსგავსი რეალიზაციების შესრულება დაპროგრამების სხვა ენებისთვის;

12.7. მოისინჯოს ავტორისეული მიდგომა სიმბოლური დიფერენციირების ამოცანისადმი სიმბოლური ინტეგრების ამოცანის გადასაწყვეტად;

12.8. გამოვლენილ იქნას ამოცანათა კლასი, რომლისთვისაც შემოთავაზებული ინსტრუმენტარი განსაკუთრებით ეფექტიანია;

12.9. შემუშავდეს ტიურინგის ვირტუალური მანქანის მოდიფიკაციის ასემბლერის გაფართოება, რომელიც მოგვცემს შესაძლებლობას შემოღებულ იქნას პრეპროცესინგის ელემენტები. ეს კი, თავის მხრივ, ხელს შეუწყობს მოსწავლეებისთვის სასტარტო ეტაპზე მოდულური დაპროგრამების გაცნობას;

12.10. ტიურინგის სასწავლო ვირტუალური მანქანის რეალიზაცია გატანილ იქნას საიტზე და ხელმისაწვდომი გახდეს გარე მომხმარებლისთვის;

12.11. ტიურინგის სასწავლო ვირტუალური მანქანისთვის გამოკვლეულ იქნას ამოხსნათა სისწორის ავტომატურად შემოწმების შესაძლებლობები;

12.12. ტიურინგის სასწავლო ვირტუალური მანქანისთვის გამოკვლეულ იქნას ტესტების გენერირების ავტომატიზების შესაძლებლობა;

12.13. შეიქმნას საიტი, რომელზედაც 24 საათის განმავლობაში მსურველს მიეცემა შესაძლებლობა გაეცნოს ტიურინგის სასწავლო ვირტუალურ მანქანას და შეასრულოს შესაბამისი სირთულის მაგალითები.

კონფერენციებში მონწილეობა

1. თ.ზარქუა. სამომხმარებლო პროგრამებიდან ფუნქციონალური გამოსახულებების დამუშავების ავტომატიზების საკითხისადმი (მოხსენების თეზისები ქართულ და რუსულ ენებზე). ნიკო მუსხელიშვილის გამოთვლითი მათემატიკის ინსტიტუტი, საქართველოს საპატრიარქოს წმიდა ანდრია პირველწოდებულის სახელობის ქართული უნივერსიტეტი. საერთაშორისო კონფერენცია "ინფორმაციული და გამოთვლითი ტექნოლოგიები", მიძღვნილი პროფესორების ელენე დეკანოსიძის და მურმან წულაძის ხსოვნისადმი. თეზისების კრებული. თბილისი, 2010 წელი, გვ. 52-53.
2. Т.Заркуа. К вопросу о реализации преобразования функциональных выражений на алгоритмических языках. აკადემიკოს ი.ფრანგიშვილის დაბადების 80 წლისთავისადმი მიძღვნილი საერთაშორისო სამეცნიერო კონფერენცია "საინფორმაციო და კომპიუტერული ტექნოლოგიები, მოდელირება, მართვა". მოხსენებათა თეზისები. საქართველო, თბილისი, 1-4 ნოემბერი, 2010 წელი. საგამომცემლო სახლი "ტექნიკური უნივერსიტეტი", 2010, გვ.135.
3. თ.ზარქუა, ა.რუხაძე. ტიურინგის მანქანის მოდიფიკაციის შემუშავება სასწავლო პროცესისათვის. საქართველოს ტექნიკური უნივერსიტეტი საერთაშორისო სამეცნიერო კონფერენცია "მართვის ავტომატიზებული სისტემები და თანამედროვე საინფორმაციო ტექნოლოგიები". მოხსენებათა თეზისები, საგამომცემლო სახლი "ტექნიკური უნივერსიტეტი", თბილისი, 2011, გვ. 196-197.

პუბლიკაციები

1. Т.Заркуа. Автоматизация обработки функциональных выражений на уровне алгоритмических языков. Internet-Edication-Sciense-2010. New Informational and Computer Tecnologies in Education and Sciense. Vinnitsia, 2010, p. 201-206
2. Т.Заркуа. К вопросу об использовании понятия сопряженного по отношению к бесскобочным выражениям. Internet-Education-Sciense-2012. New Informational and Computer Tecnologies in Education and Sciense. Vinnitsia, 2012, p.156-157
3. Т.Заркуа, С.Барнабишвили. К вопросу стартового обучения программированию. Internet-Education-Sciense-2014. New Informational and Computer Tecnologies in Education and Sciense. Vinitsia, 2014, p.269-271.

Abstract

The dissertation “Some aspects of software portability and extensibility” examines theoretical and practical aspects of methodological means intended to improve important parameters of software quality, such as portability and extensibility.

These methods are based on so-called “Polish notation” of representing functional expressions. In terms of this, a software package facilitating work with such expressions has been created. Actually, this can be regarded as an extension to a programming language, which provides programmers with brand new ways of handling substantial class of problems. Two different versions of the mentioned software have been designed.

As for the first version, it was developed with Pascal programming language as an independent module. Reverse Polish Notation is used for internal representation of expressions, implemented by linked lists. The module comprises two main components. The first one is a function, which analyses string containing functional expression written in infix notation and returns an internal representation of the same expression but converted into postfix form, provided that no errors are found. Otherwise, a message containing information about the error is returned. The second component, also a function, evaluates the expression using values of variables, passed as arguments to the function. Generally, the expressions can contain several variables with all four arithmetic operators, two types of exponentiation (with integer and non-integer exponent), some elementary functions and one ternary operator. This particular implementation was part of the software package developed by a department of computer software of Tbilisi State University in 1999. Exhibited within an exhibition “Info-99”, the software was awarded diploma for the 2nd place.

The second version, is designed with C++. In contrast with the first one, this implementation is limited to only single variable expressions with all four arithmetic operators, exponentiation and elementary functions. Vector data structure is used for internal representation of functional expressions. Not only does it support converting string expression into internal representation and vice versa, but also differentiation and simplification. Based on this version, a C++ class could be created, providing a better experience for programmers when dealing with certain sorts of mathematical problems. Description of an interface of this class is also covered in the following thesis.

Within the scope of this dissertation, a conception of teaching computer programming basics at introductory level has also been developed. Based on a modification of Turing Machine, yet preserving the key ideas, this methodology is platform independent, portable and easy to use. Implementations of the concept have been used in classes held at St. Andrew the First-Called Georgian University of the Patriarchate of Georgia and boarding school “IB Mtiebi” since 2014.

The topicality of the subject is provoked by stably high rates of software volume increase and the fact that the computer usage area currently covers virtually every field of human activity.

The principal results of the thesis are the following:

1. The term of a conjugate to parenthesis-free notations of algebraic expressions was introduced and the regularities associated to this term were detected. The author has given a strict mathematical proof that it is possible to withdraw the necessity of convertibility from postfix notation to prefix and vice versa. This is significant for problems which require not only computing the numeral values of functional expressions, but also their

functional translation. On basis of these results, a software was developed which enables the user to operate with functional expressions as with any standard data type.

2. Set of algorithms providing functionality of converting between polish conjugate and standard polish notations has been introduced.
3. The algorithm involving two stacks for directly converting from infix to prefix notation has been designed.
4. The methodology of enabling programmers to perform differentiation of functional expressions using an internal language created for this very purpose.
5. The interface part of special class which enables the user to processing functional expressions from the level of algorithmic language was developed.
6. The conception of modification of Turing Machine intended to facilitate conveying the basics of programming to beginners has been developed.

The following paragraphs suggest the scientific originality of the thesis:

- The term of a conjugate to Polish notation, 7 stated and proven theorems and 8 results introduced by the author are brand new.
- The algorithm of directly obtaining prefix notation of an expression from the corresponding infix notation using 2 stacks is new.
- The new is also all algorithms concerning the conjugate.
- The internal language for describing the rules of differentiation and the corresponding implementation (which is portable and extensible) are completely new as well.
- Overall, the methodology enabling performing transformations of functional expressions (given as a string in a natural form), including differentiation and evaluation (with specified values of variables). Moreover, programmers are able to conduct all these operations from a programming language.
- Finally, suggested modification of Turing Machine.

The practical value of the dissertation consists in the fact that all developed concepts have been implemented as software.