

**Visual Basic-ის ქართული
სახემმდგანელო დამწყებებისათვის**

(ნაწილი 1)

**სანდრო კალატოზიშვილი
a.k.a. - 7th Ghost -**

CyberSky Tech Center

E-Mail: kala0x13@gmail.com

Web: <http://skycent.in/>

წიგნს ვუძღვნი “Your Nightmore...”-ს
If I Don't Live Today, Then I Might Be Here Tomorrow..
See U in Vegas! .. ^^

წინასიტყვაობა

სახემმდევანელოში გადმოცემულია Microsoft Visual Basic ენაში პროგრამების შემუშავების საკითხები. განხილულია VB ენის საფუძვლები, მონაცემთა ტიპები, მმართველი ოპერატორები, სტრუქტურირებული ტიპები. გადმოცემულია ობიექტზე ორიენტირებული დაპროგრამების პრინციპები. განხილულია განსაკუთრებული სიტუაციები, ფაილებთან და კატალოგებთან მუშაობა. წიგნში მოყვანილია მაგალითები და ამოცანები ამოხსნებით.

წიგნი განკუთვნილია დაპროგრამების შესწავლის ნებისმიერი მსურველისათვის, ასევე უკვე გამოცდილი პროგრამისტებისთვის, ვისაც სურს გაიღრმავოს თავისი ცოდნა აღნიშნულ საკითხში. წიგნი დაეხმარება დამწყებ პროგრამისტებს ორიენტაციაში და მასალის მარტივად ათვისებაში.

წიგნში გამოყენებული ნებისმიერი სურათისა თუ ტექსტის გამოყენება, სპეციალური ნებართვის გარეშე, აკრძალულია და სხვისი საკუთრების მითვისება ისჯება კანონით!

All Rights Reserved © CyberSky Tech. 2011-12

<http://skycent.in/>

სარჩევი

წინასიტყვაობა	3
შესავალი	5
პროგრამირება და ლოგიკა	6
ლოგიკის ელემენტები	6
ლოგიკური გამოსახულებები	8
Visual Basic ენის საფუძვლები	10
ცვლადები და მუდმივები	11
ოპერატორები VB ენაში	13
If ოპერატორი	13
For ... Next ოპერატორი	14
Do, Loop, While, Wend ოპერატორები	14
პირველი პროგრამა VB ენაში	16
პროგრამები ToolBox ის ელემენტებით	21
CheckBox-ის გამოყენება	21
ListBox-ის გამოყენება	24
LinkLabel-ის გამოყენება	27
მათემატიკური ოპერატორები VB-ენაში	29
პირველი პროგრამა მათემატიკური ოპერაციებით	29
GUI პროგრამა მათემატიკური ოპერაციებით	33
ფაილებთან მუშაობა	36
ტექსტური ფაილების გახსნა	36
ტექსტური ფაილების შენახვა	39
პირველი ნაწილის დასასრული	40

შესავალი

Visual Basic-ი პოპულარული დაპროგრამების ენაა, რომელიც მიეკუთვნება ვიზუალურ და ობიექტზე ორიენტირებულ დაპროგრამების ენათა რიცხვს და საშუალებას იძლევა მარტივად და მოხერხებულად შეიქმნას Microsoft Windows-ის ოპერაციულ სისტემაზე გაშვებადი პროგრამები.

წიგნში განხილულია მაგალითები Microsoft Visual Basic 2010 ის ფონზე, რომელიც წარმოადგენს ძალიან პოპულარულ Visual Basic .NET პროგრამირების ენის ბოლო ვერსიას.

Visual Basic 2010 საშუალებას იძლევა მარტივად და მოხერხებულად შექმნათ Windows Forms აპლიკაციები, WPF Windows აპლიკაციები, WPF Browser აპლიკაციები, Web აპლიკაციები, მობილური დივაისების აპლიკაციები და სხვადასხვა Web სერვისები.

ეს წიგნი წარმოადგენს აღნიშნული საკითხის ქართულად გადმოცემის ერთერთ პირველ მცდელობას და დაეხმარება დამწყებ პროგრამისტებს პროგრამირების გზის გაკვალვაში. რატომღუნდა არსებობს სხვა ენებიც რომლებიც მეტნაკლებად დახვეწილია VB-სთან შედარებით, მაგრამ შენ თუ ხარ დამწყები .NET პროგრამირებაში, ეს საუკეთესო არჩევანია შენთვის!

წიგნის დაწერის ერთერთ მთავარ მიზანს წარმოადგენს - გადმოვცე ჩემი ცოდნა დამწყებ პროგრამისტებს და ასევე გავიღრმავო მე თვითონ ცოდნა აღნიშნულ სფეროში, რადგან მეც ასევე დამწყები დონის პროგრამისტებს მივეკუთვნები.

რა გვჭირდება სწავლის პროცესში:

- თავისუფალი დრო
- ოპერაციული სისტემა: Windows XP / Vista / 7
- Microsoft Visual Basic 2010
- მონდომება

პროგრამირება და ლოგიკა

პროგრამირება წარმოადგენს არც თუ ისე მარტივ პროცესს პროგრამისტისთვის. პროგრამის წერის პროცესი შეიძლება ძალიან დიდ ხანს გაგრძელდეს და ეს ყოველივე პროგრამისტის ცოდნასა და ლოგიკურ აზროვნებაზეა დამოკიდებული. არსებობს სხვადასხვა ლოგიკური ელემენტები რომელთა ცოდნაც დაეხმარება პროგრამისტს სრულყოფილი და საჭირო ალგორითმის მოფიქრებაში.

ლოგიკა(მათემატიკური ლოგიკის ელემენტები - ბულის ალგებრა)

ლოგიკაში იგულისხმება მსჯელობის ხელოვნების ანალიტიკური თეორია, რომლის მიზანია სწორი მსჯელობის პრინციპების სისტემატიზაცია და კოდიფიკაცია. ლოგიკის მათემატიკურ თეორიაში ფუნდამენტურია გამონათქვამის ცნება. გამონათქვამი ეწოდება თხრობით წინადადებას, რომელიც შეიძლება კლასიფიცირებულ იქნას, როგორც ჭეშმარიტი ან მცდარი.

მათემატიკურ და სხვა მსჯელობებში ხშირად გვხვდება თხრობითი წინადადებები, რომლებიც მიიღება მარტივი თხრობითი წინადადებების შეკავშირებით სათანადო სიტყვების გამოყენებისას, როგორიცაა: „და“, „ან“, „თუ ... მაშინ“, „მაშინ და მხოლოდ მაშინ, როცა“, „არ“.

კონიუნქცია. თუ წინადადება მიღებულია ორი გამონათქვამის ერთმანეთთან „და“ სიტყვით დაკავშირების შედეგად, მაშინ ისიც გამონათქვამია და მას კონიუნქცია ეწოდება. მაგალითად, შემდეგი წინადადება წარმოადგენს კონიუნქციას:

წვიმა მოდის და ქარი უბერავს.

ეს თხრობითი წინადადება გვეუბნება, რომ ერთდროულად ორ ფაქტს აქვს ადგილი: წვიმაც მოდის და მასთან ერთად ქარიც უბერავს.

კონიუნქცია ჭეშმარიტია, როდესაც მასში მონაწილე ორივე გამონათქვამი ჭეშმარიტია. ყველა სხვა შემთხვევაში კონიუნქცია მცდარია.

კონიუნქცია ხასიათდება შემდეგი თვისებებით:

$A \wedge B = B \wedge A$ (კომუტატიურობა)

$(A \wedge B) \wedge C = A \wedge (B \wedge C)$ (ასოციაციურობა)

$A \wedge (\neg A) = \text{false}$ (მცდარი).

დიზიუნქცია. თუ წინადადება მიღებულია ორი გამონათქვამის ერთმანეთთან „ან“ სიტყვით დაკავშირების შედეგად, მაშინ ისიც გამონათქვამია და მას დიზიუნქცია ეწოდება. მაგალითად, შემდეგი წინადადება წარმოადგენს დიზიუნქციას:

ბენზინი გათავდა ან მანქანა გაფუჭდა.

ეს თხრობითი წინადადება გვეუბნება, რომ ორ ფაქტიდან ერთს მაინც აქვს ადგილი: შესაძლოა, რომ მანქანის გაჩერების მიზეზი არის ის, რომ ბენზინი გათავდა, ან ის, რომ მანქანა გაფუჭდა, თუმცა შესაძლოა, რომ ორივეს ერთად ჰქონდეს ადგილი: ბენზინიც გათავდეს და იმავდროულად მანქანაც გაფუჭდეს.

დიზიუნქცია მცდარია, როდესაც მასში მონაწილე ორივე გამონათქვამი მცდარია. ყველა სხვა შემთხვევაში დიზიუნქცია ჭეშმარიტია.

დიზიუნქცია ხასიათდება შემდეგი თვისებებით:

$$A \vee B = B \vee A$$

$$(A \vee B) \vee C = A \vee (B \vee C)$$

$$A \vee (\neg A) = \text{true} \text{ (ჭეშმარიტი).}$$

იმპლიკაცია. თუ წინადადება მიღებულია ორი გამონათქვამის ერთმანეთთან „თუ...მაშინ“ სიტყვებით დაკავშირების შედეგად, მაშინ ისიც გამონათქვამია და მას იმპლიკაცია ეწოდება. მაგალითად, შემდეგი წინადადება წარმოადგენს იმპლიკაციას:

თუ მდინარე აღიდებულია, მაშინ ბანაობა აკრძალულია.

ეს წინადადება გვეუბნება, რომ როცა მდინარე აღიდებულია, მაშინ ბანაობა აკრძალულია, თუმცა ბანაობა შესაძლოა აკრძალული იყოს სხვა რამის გამოც, მაგალითად იმის გამო, რომ წყალი ჭუჭყიანია. სხვანაირად რომ ვთქვათ, წყლის აღიდება იწვევს ბანაობის აკრძალვას, მაგრამ არა პირიქით.

იმპლიკაცია მცდარია, როდესაც მასში მონაწილე პირველი გამონათქვამი (სიტყვა „თუ“-ს შემდეგ) ჭეშმარიტია, ხოლო მეორე გამონათქვამი (სიტყვა „მაშინ“-ის შემდეგ) მცდარი. ყველა სხვა შემთხვევაში იმპლიკაცია ჭეშმარიტია.

ამბობენ, რომ იმპლიკაციაში პირველი გამონათქვამი (სიტყვა „თუ“-ს შემდეგ) არის მეორე გამონათქვამის (სიტყვა „მაშინ“-ის შემდეგ) საკმარისი პირობა, ხოლო მეორე გამონათქვამი არის პირველი გამონათქვამის აუცილებელი პირობა.

ეკვივალენცია. თუ წინადადება მიღებულია ორი გამონათქვამის ერთმანეთთან „მაშინ და მხოლოდ მაშინ, როცა“ სიტყვებით დაკავშირების შედეგად, მაშინ ისიც გამონათქვამია და მას ეკვივალენცია ეწოდება. მაგალითად, შემდეგი გამონათქვამი ეკვივალენტია:

ორი სამკუთხედი ტოლია მაშინ და მხოლოდ მაშინ, თუ მათი სამივე გვერდი ტოლია.

ეს წინადადება გვეუბნება, რომ ორი სამკუთხედის ტოლობას და მათი სამივე გვერდის ტოლობას ან ერთდროულად აქვს ადგილი, ან არა. ანუ არ შეიძლება, რომ ორი სამკუთხედი ტოლი იყოს და მათი სამივე გვერდი კი არა და პირიქით.

ეკვივალენტია ჭეშმარიტია, როდესაც მასში მონაწილე გამონათქვამები ერთდროულად ჭეშმარიტია ან მცდარი. ყველა სხვა შემთხვევაში ეკვივალენტია მცდარია.

უარყოფა. თუ წინადადება მიღებულია გამონათქვამისათვის წინ „არ“ სიტყვის დამატების შედეგად, მაშინ ისიც გამონათქვამია. მაგალითად, შემდეგი წინადადება არის უარყოფა:

არა წვიმა მოდის (მკითხველისთვის უფრო მისაღებია ამ წინადადების ასეთი ფორმულირება – „წვიმა არ მოდის“).

ეს წინადადება ამბობს, რომ „წვიმა მოდის“ – არ არის სწორი. ანუ ამტკიცებს ამ გამონათქვამის საწინააღმდეგოს.

უარყოფა ჭეშმარიტია, როდესაც მასში მონაწილე გამონათქვამი მცდარია. წინააღმდეგ შემთხვევაში უარყოფა მცდარია.

ლოგიკური გამოსახულებები

გამოსახულებას, რომლის მნიშვნელობაც არის ჭეშმარიტი ან მცდარი, ლოგიკური გამოსახულება ეწოდება.

ინფორმატიკაში ლოგიკურ გამოსახულებას ხშირად ლებულობენ შედარების ოპერაციების გამოყენებით. მაგალითად, $x \geq 5$, $y \leq 10$ და ა. შ. იმის მიხედვით, თუ რისი ტოლია x , ამ გამოსახულებების მნიშვნელობა ან ჭეშმარიტია, ან მცდარი.

ცვლადს, რომლის დასაშვებ მნიშვნელობათა სიმრავლე შედგება მხოლოდ ორი ელემენტისაგან: ჭეშმარიტი და მცდარი, ლოგიკური ცვლადი ეწოდება.

ხშირად სიტყვა „ჭეშმარიტის“ ნაცვლად იყენებენ აღნიშვნას T , ხოლო სიტყვა „მცდარის“ ნაცვლად იყენებენ აღნიშვნას F .

ლოგიკური გამოსახულებების და ცვლადების ერთმანეთთან ლოგიკური ოპერაციებით დაკავშირების შედეგად მიიღება ისევ ლოგიკური გამოსახულება.

ლოგიკურ გამოსახულებებში ლოგიკური ოპერაციები ჩაიწერება სპეციალური სიმბოლოების გამოყენებით:

$\wedge$ – კონიუნქცია

$\vee$ – დიზიუნქცია

$\neg$ – ლოგიკური უარყოფა

$\rightarrow$ – იმპლიკაცია

$\leftrightarrow$ – ეკვივალენცია

მაგალითად, თუ P და Q სიმბოლოებით ავლნიშნავთ ლოგიკურ გამოსახულებებს, მაშინ მათი მონაწილეობით ლოგიკური ოპერაციები ასე ჩაიწერება:

$P \wedge Q, P \vee Q, P \rightarrow Q, P \leftrightarrow Q, \neg P, (P \wedge Q) \rightarrow \neg(P \vee Q).$

ცხადია, რომ ლოგიკური უარყოფის გარდა ყველა ოპერაცია ორმონაწილიანია. შევადგინოთ ცხრილი, რომელშიც ჩავწერთ ლოგიკური ოპერაციების მნიშვნელობებს მათი

მონაწილეების ყველა შესაძლო მნიშვნელობებისათვის:

P	Q	$P \wedge Q$	$P \vee Q$	$P \rightarrow Q$	$P \leftrightarrow Q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

Visual Basic ენის საფუძვლები

პირველი პროგრამა:

შევადგინოთ პირველი უმარტივესი პროგრამა, რომელიც ორ რიცხვს შეკრიბავს და შედეგს შეტყობინების სახით ეკრანზე გამოიტანს. პროგრამის განხილვა დაგვეხმარება სწავლის პროცესში.

```
Sub First()
```

```
  'პირველი პროგრამა
```

```
  Dim a, b, c As Integer
```

```
  a = 1
```

```
  b = 2
```

```
  c = a + b
```

```
  MsgBox
```

```
End Sub
```

თუ განვიხილავთ ამ უმარტივეს პროგრამულ კოდს ნაბიჯ-ნაბიჯ, დავინახავთ, რომ VB ენაზე შედგენილი პროგრამა იწყება Sub სიტყვით, რაც წარმოადგენს პროგრამული კოდის დაწყებას და პროცედურის გამოძახებას. Sub-ს მოსდევს უკვე First რაც წარმოადგენს პროცედურის სახელს სახელს. მაგალითად ამ პროცესს ჩვენ ვუწოდებთ სახელი First.

შემდეგ მოდის კომენტარი, რომელიც პროგრამულ კოდში აპოსტროფითაა გამოყოფილი. კომენტარი საჭიროა რათა პროგრამა უფრო ნათლად აღსაქმელი იყოს როგორც სხვისთვის, ასევე შენთვის და დაგეხმაროს დაწერის პროცესში პროგრამის უკეთ აღქმაში. ეგ არ შეუშლის პროგრამის კომპილაციას ხელს, რადგან ვიჟუალ ბეისიკში პროგრამული კოდის კომპილატორი უბრალოდ ახტება ტექსტს, რომელიც აპოსტროფით იწყება. ის შეიძლება იყოს კოდის თავში, ბოლოში, შუაში ან თუნდაც გვერდით.

დაპროგრამების ენის უმარტივეს კონსტრუქციებს ცვლადები და კონსტანტები (მუდმივები) მიეკუთვნება. შემდეგი ნაბიჯია Dim ოპერატორი რომელიც პროგრამულ კოდში ცვლადების შემოტანის ოპერატორია. მას მოსდევს უკვე ცვლადები a, b, და c რომლებსაც მინიჭებული აქვთ ტიპი Integer-ი, რაც მთელი ტიპის რიცხვით მონაცემებს აღნიშნავს. ეს ყველაფერი საჭიროა რათა მოხდეს მეხსიერებაში არეების გამოყოფა შესაბამისი ტიპის მონაცემთა მოსათავსებლად.

შემდეგაა ამ ცვლადებზე მნიშვნელობების მინიჭება: a=1 და b=2, რის შემდეგაც ხდება მარტივად მათი შეკრება და MsgBox-ის საშუალებით მათი ეკრანზე გამოტანა.

და ბოლოს, პროგრამას უნდა ჰქონდეს როგორც დასაწყისი, ასევე დასასრულიც. ამიტომ პროგრამა სრულდება End Sub ოპერატორის მეშვეობით.

ცვლადები და მუდმივები

Visual Basic-ში მონაცემთა სპეციფიკურ კლასს ობიექტთა თვისებების მნიშვნელობები ქმნის. თავისი ფიზიკური არსით ცვლადი მეხსიერების არეა, რომელსაც გარკვეული სახელი (იდენტიფიკატორი) გააჩნია. ცვლადის სახელი უნდა იწყებოდეს ასოთი, მასში დასაშვებია ასოების, ციფრების და ხაზგასმის (“-“) სიმბოლოთა გამოყენება. ცვლადის სახელში სიმბოლოების რაოდენობა არ უნდა აღემატებოდეს 255-ს და დაუშვებელია (Space) ინტერვალის გამოყენება. ცვლადების სახელებად არ შეიძლება დაპროგრამების ენის ოპერატორების და ობიექტის თვისებების აღმნიშვნელი სახელების გამოყენება. მეხსიერება, რომელსაც ცვლადი მიმართავს, სხვადასხვა ტიპის მონაცემებს შეიძლება შეიცავდეს.

VB-ში განასხვავებენ მონაცემებისა და ცვლადების შემდეგ ძირითადი ტიპებს:

1. **Byte** - მთელი ტიპის რიცხვითი მონაცემები 0 – 255 დიაპაზონიდან (მეხსიერებაში იკავებს ერთ ბაიტს ანუ მეხსიერების ერთ უჯრედს)
2. **Integer** - მთელი ტიპის რიცხვითი მონაცემები. 32768 – 32767 დიაპაზონიდან. (მეხსიერებაში იკავებს ორ ბაიტს)
3. **Long** - გრძელი მთელი ტიპის რიცხვითი მონაცემები დიაპაზონიდან 2147483648 – 214748647 (მეხსიერებაში იკავებს ოთხ ბაიტს)
4. **Single** - სტანდარტული სიზუსტის (ექვსი ნიშნით ათობითი წერტილის შემდეგ) როგორც დადებითი ასევე უარყოფითი დიაპაზონიდან $1.4013 \cdot 10^{-45}$ – $3.4028 \cdot 10^{38}$ (მეხსიერებაში იკავებს ოთხ ბაიტს)
5. **Double** - ორმაგი სიზუსტის (თოთხმეტი ნიშანი ათობითი წერტილის შემდეგ) ნამდვილი რიცხვები $4.940656484 \cdot 10^{-324}$ – $1.7976931349 \cdot 10^{308}$ (მეხსიერებაში იკავებს რვა ბაიტს)
6. **String** - ტექსტური სტრიქონული მონაცემები, რომლებიც ყოველთვის ბრჭყალებში თავსდება (მეხსიერებაში ყოველი სიმბოლო ერთ ბაიტში ინახება). გვაქვს მისი კერძო შემთქვევა String*n, სადაც n სტრიქონულ მონაცემში მაქსიმალური დასაშვები რაოდენობაა.
7. **Boolean** - ლოგიკური მონაცემების ტიპი. პროგრამებში ორი შესაძლო მნიშვნელობიდან იღებს ერთ ერთს. კერძოდ 1.(True) ან 0 (False). მეხსიერებაში იკავებს ორ ბაიტს.

8. **Currency** - ნამდვილი რიცხვები წარმოდგენილი ფულადი ერთეულის სტილში (ფულად ფორმატში). ამ ტიპის რიცხვებში მთელი ნაწილი 15 ნიშანს შეიცავდეს ხოლო წილადი ნაწილი - 4 თანრიგს. მეხსიერებაში იკავებს რვა ბაიტს) ძირითადი გამოყენება ფინანსური ხასიათის ინფორმაციის დამუშავების დროს.

9. **Date** – ტიპის მონაცემები შეიცავს ინფორმაციას თარიღისა და დროის შესახებ.

10. **Variant** - მონაცემთა განსაკუთრებული ტიპი, რომლის წინასწარ განსაზღვრული ტიპი არ არსებობს. მან შეიძლება მიიღოს ზემოთ ჩამოთვლილი ნებისმიერი ტიპის მნიშვნელობა. (მეხსიერებაში იკავებს თექვსმეტ ბაიტს რის გამოც მისი გამოყენება პროგრამებში შეზღუდულია). VB.NET ში ის ამოღებულია და მის მაგივრად იხმარება **Object** რომელსაც გააჩნია ზუსტად იგივე მნიშვნელობა.

ცვლადების ასაღწერად არსებობს ორი ხერხი:

1. ცვლადების წინასწარ აღწერა
2. ცვლადების აღწერა შესაბამისი იდენტიფიკატორის გამოყენებით

#1. პირველი ხერხი ხორციელდება Dim და As ოპერატორული სიტყვებით, რომელსაც აქვს შემდეგი სახე:

Dim ცვლადი As ტიპი

Dim ოპერატორით შეიძლება აღწერილი იქნეს რამდენიმე ცვლადი ერთად თუ მას მინიჭებული ექნება ერთნაირი ტიპი. მაგალითად:

Dim a, b, c, d As Integer

#2. მეორე ხერხის გამოყენებით, ცვლადების აღწერა ხდება შესაბამისი იდენტიფიკატორის მეშვეობით, რომელიც ცვლადის ბოლოს მოთავსდება. VB ენაში მხოლოდ ექვს ტიპს აქვს შესაბამისი იდენტიფიკატორი:

ტიპი	Integer	Long	Single	Double	String	Cuurency
იდენტიფიკატორი	%	&	!	#	\$	@

ანუ მაგალითად:

Dim a As Integer იგივეა რაც Dim a%

რაც შეეხება კონსტანტებს (მუდმივები), კონსტანტები ის მუდმივი სიდიდეებია, რომელთა მნიშვნელობების შეცვლა პროგრამის შესრულების პროცესში დაუშვებელია. მათ აღწერას Const ოპერატორი ემსახურება, რომლის ჩაწერის ზოგადი სახე ასეთია:

Const სახელი As ტიპი = მნიშვნელობა

მაგალითად:

```
Const a As Double = 3.14
```

```
Const b As String = "Visual Basic eBook"
```

ოპერატორები VB ენაში

#1. If ოპერატორი

If წარმოადგენს პირობითი გადასვლის ოპერატორს. ანუ მარტივად რომ აგისხნათ, if ოპერატორი ჯერ ამოწმებს მოცემულ პირობას და შემდეგ ასრულებს Then ფუნქციის შემდეგ მოცემულ პროგრამულ კოდს. If ოპერატორის მარტივად გასაგებად, მოვიყვანო რამდენიმე ცხოვრებისეულ მაგალითს. მაგალითად ცხოვრებაში ყოფილა ესეთი მომენტები, როდესაც გვითქვია: თუ დღეს იწვიმებს მაშინ თბილად ჩავიცვავ, ან მაგალითად თუ დღეს მზიანი დღე იქნება მაშინ მოკლევით გავალ გარეთ. სწორედ ასეთი კონსტრუქცია აქვს IF ოპერატორს, რომლის ზოგადი სახე ასეთია:

If პირობა Then ოპერატორი

If ოპერატორს აქვს რამდენიმე სახე, რომლებიცაა: if, elseif და else.

ElseIf წარმოადგენს If ოპერატორის დამატებას, რომლის საშუალებითაც მოწმდება მეორე მოცემული პირობა და შემდეგ სრულდება ოპერაცია. ხოლო რაც შეეხება else-ს, თუ არცერთი ზემოთ მოცემული პირობა არაა სწორი, ამ შემთხვევაში სრულდება ოპერაცია, რომელიც else-ს შემდეგ წერია. მაგალითად:

```
Dim a, b As Integer
```

```
a = 5
```

```
b = 5
```

```
If a > b Then
```

```
MsgBox("a მეტია b-ზე")
```

```
ElseIf a < b Then
```

```
MsgBox("a ნაკლებია b-ზე")
```

```
Else
```

```
MsgBox("a ტოლია b-სი")
```

```
End If
```

ამ შემთხვევაში პროგრამა ამოწმებს a მეტია თუ არა b-ზე, თუ მეტია მაშინ ეკრანზე გამოვა შეტყობინება: "a მეტია b-ზე". ამის შემდეგ იწყება მეორე პირობის შემოწმება a ნაკლებია თუ არა b-ზე, თუ ნაკლებია ეკრანზე გამოვა ასევე შეტყობინება "a ნაკლებია b-ზე" და თუ არცერთი ზემოთ მოცემული პირობა არ აღმოჩნდება ჭეშმარიტი, მაშინ პროგრამა შეასრულებს else-ს შემდგომ დაწერილ ოპერაციას და ეკრანზე გამოვა შეტყობინება "a ტოლია b-სი".

#2. For ... Next ოპერატორი

For არის ოპერატორი რომელიც რაღაც განსაზღვრული რაოდენობით ასრულებს რაღაც პროგრამულ კოდს, ანუ ეს არის ციკლის ოპერატორი რომელსაც აქვს დასაწყისი და აქვს ბაზოლოვება ანუ ცაკლიდან გომოსვლის პირობა. მაგალითად:

```
Dim a As Byte, b As Integer
for a=0 to 50
b=b+1
next i
```

ამ შემთხვევაში ჩვენ ვიძახებთ ციკლს, რომელიც იყენებს ცვლად a-ს, რომელსაც თავიდან მიენიჭება მნიშვნელობა 0 და გაიზრდება 1-ით მანამდე, სანამ მისი მნიშვნელობა არ მიაღწევს 50-ს.

#3. Do, Loop, While, Wend ოპერატორები

Do არის უსასრულო ციკლის ოპერატორი, რომელიც მთავრდება Loop ის მეშვეობით და არის ძალიან მარტივი. მისი ზოგადი სახე ასეთია;

```
Do
ოპერაცია
Loop
```

While და Wend-იც Do, Loop ის მსგავსად ციკლის ოპერატორებია, მაგრამ მათი განსხვავება იმაში მდგომარეობს, რომ While-ს აქვს გამოსვლის პირობები მითითებული და ციკლი გრძელდება მანამდე, სანამ გამოსვლის პირობები ჭეშმარიტია. ანუ მაგალითად:

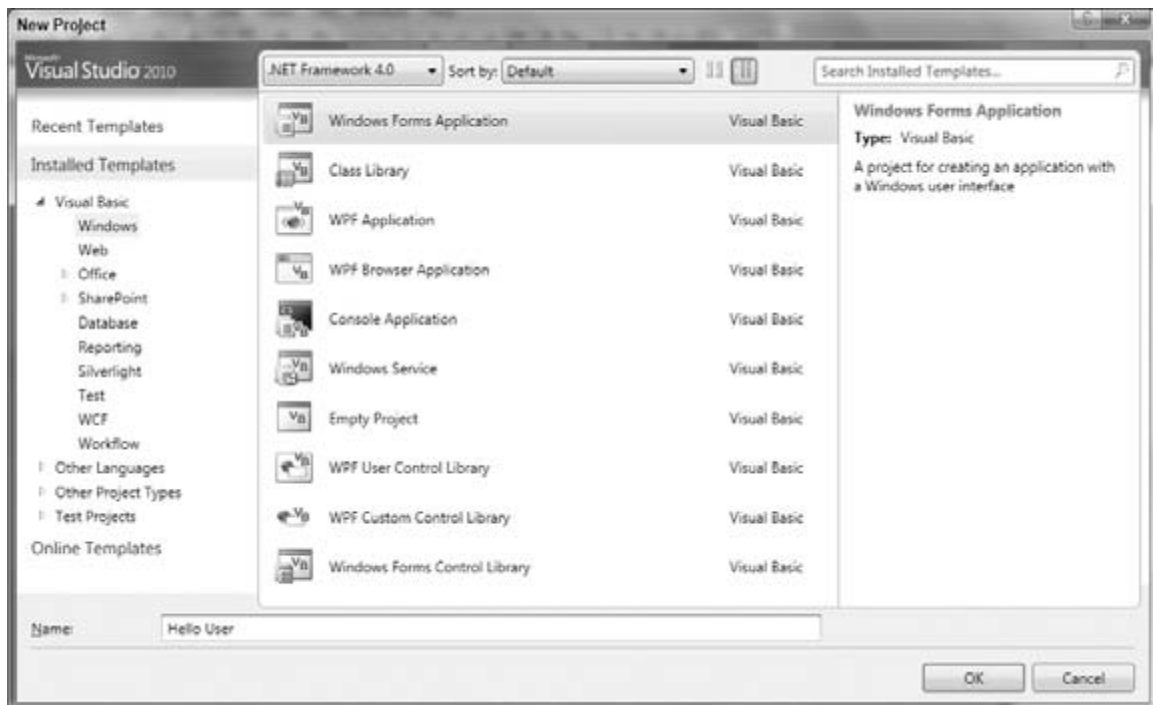
```
While x > y
ოპერაცია
Wend
```

ამ შემთხვევაში ოპერაცა შესრულდება მანამდე, სანამ x იქნება მეტი y-ზე.

პირველი პროგრამა Visual Basic-ში

როგორც უკვე შეცნობილია, ჩვენ გამოვიყენებთ ერთერთ პოპულარულ პროგრამას Visual Basic 2010, რომელიც დღესდღეობით ბოლო ვერსიას წარმოადგენს.

პირველ რიგში საჭიროა დავანსტალიროდ პროგრამა და ამის შემდეგ ეკრანზე გამოჩნდება პიქტოგრამა, რომელზე ორჯერ კლიკით ჩვენ გავუშვებთ მას. პროგრამის გაშვების შემდეგ გაიხსნება ფანჯარა სადაც ჩვენ ვირჩევთ New Project > Windows Forms Application და ვაწკებთ OK-ს.



ამის შემდეგ, ეკრანზე გაიხსნება თქვენი პირველი პროგრამის საწყისი ვიზუალური ფორმა, რომელზეც შესაძლებელია სხვადასხვა ელემენტების, ღილაკების, ტექსტების და ბევრი რამის დამატება.

პროგრამაში სამუშაოს ძირითადი ნაწილი შეიცავს:

- **ToolBox** - საიდანაც ხდება სხვადასხვა პროგრამისთვის საჭირო რესურსის დამატება თქვენს პროგრამაზე. მაგალითად: ღილაკების, ლინკების, ტექსტების, ქონექტორების და ა.შ.
- **Design window** - სადაც ხდება თქვენი პროგრამის ვიზუალური დამუშავება, ინტერფეისის აწყობა, სტილის გალამაზება და ა.შ.
- **Solution Explorer** - ესაა ფანჯარა საიდანაც ხდება თქვენი პროგრამის სხვადასხვა ნაწილების არჩევა. ეს შეიძლება იყოს პროგრამული კოდის, პრიექტის ან სხვა რამის არჩევა.

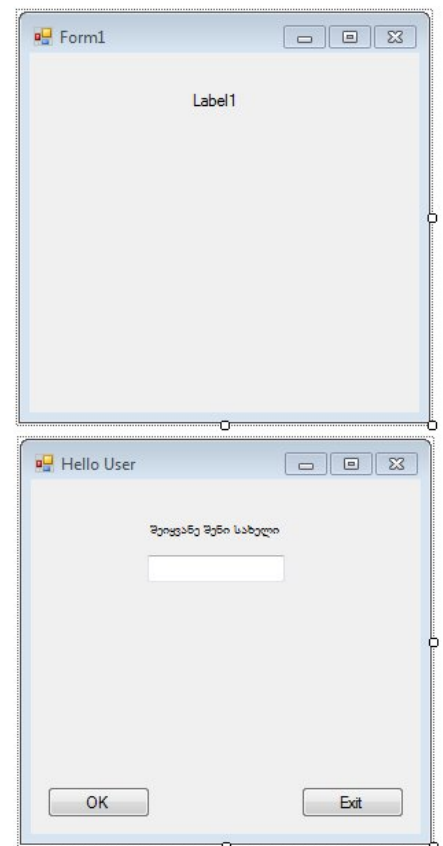
- **Properties** - აქედან ხდება თქვენსმიერ შექმნილ პროგრამაზე სხვადასხვა კონფიგურაციების გასწორება და პროგრამისთვის მორგება. ეს შეიძლება იყოს როგორც დამატებული ტექსტის გამუქება ან ფონტის შერჩევა, ასევე პროგრამის ჩარჩოზე სათაურის დაწერა და ბევრი სხვა რამ.

მაშ ასე, დავიწყეთ ჩვენი პირველი პროგრამის შექმნა, რომელსაც დავარქვათ Hello User.

პირველ რიგში ჩვენს ჩარჩოს დავარქვათ სათაური, რასაც ვაკეთებთ Properties პანელიდან. ამისთვის ჯერ ვაჭერთ ჩვენი პროგრამის საწყის ვიზუალურ ჩარჩოს და შემდეგ Properties პანელში ვეძებთ Text-ს, სადაც Form1-ს ვანაცვლებთ ტექსტით Hello User. რის შემდეგადაც აღნიშნული ტექსტი დაეწერება ჩვენს ჩარჩოს.

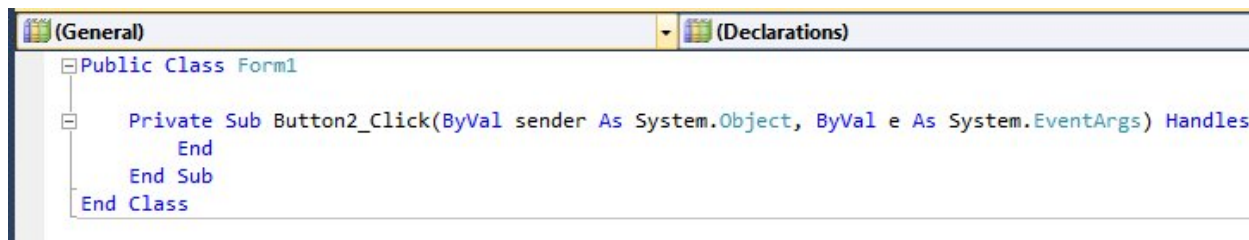
ამის შემდეგ უკვე გადავდივართ პროგრამის ვიზუალური მხარის აწყობაზე.

- 1) დამატეთ ტექსტს Label1 რომელიც ToolBox-ში მდებარეობს და შეუცვალეთ მას სახელი Properties მენიუდან. ამისთვის ჯერ ვამატებთ ტექსტის ფორმას ToolBox-ში Label1-ზე დაკლიკვით და შემდეგ პროგრამის ვიზუალურ ჩარჩოზე დაკლიკვით. ხოლო ტექსტის შესაცვლელად ვხსნით ამ ჩვენ Label1-ს კონტექსტური მენიუთი, ვირჩევთ Properties-ს და შემდეგ Properties მენიუმში ვეძებთ Text-ს და მანდ Label1-ს ვცვლით ამით „შეიყვანე შენი სახელი“
- 2) ამის შემდეგ ჩვენს ფორმას ვამატებთ ToolBox-იდან TextBox-ს და ორ Button ს ანალოგიურად, ვცვლით button-ებზე ტექსტს, ერთს ვაწერთ OK და მეორეს Exit და ვალაგებთ მათ ისე, როგორც სურათზეა.
- 3) ესო, პროგრამის ვიზუალური ნაწილის აწყობას ფაქტობრივად მოვრჩით უკვე. ახლა საქმე მიდგა პროგრამული კოდის დაწერაზე და მის პროგრამასთან ინტეგრაციაზე. იმისთვის რომ ჩვენ განვათავსოთ ჩვენი დაწერილი პროგრამული კოდი ამ პროგრამაში, ჩვენ ის უნდა ჩავსვათ თვითოეულ ობიექტში. ანუ მარტივად რომ აგიხსნათ, OK ღილაკსაც და Exit ღილაკსაც ცალცალკე კოდის დაწერა და ცალცალკე მისი მორგება სჭირდება პროგრამის ვიზუალურ ნაწილში.



ამისთვის ჩვენ ორჯერ სწრაფად ვაწვებით Exit ღილაკს, რითიც ვხსნით ჩვენი ვიზუალური ფორმის პროგრამულ მხარეს, სადაც პროგრამული კოდი უნდა განთავსდეს. პირველი რასაც შეამჩნევთ არის ის, რომ კოდის დაწერის ნახევარს თავად Visual Basic 2010 აწარმოებს, ანუ პროგრამა გეხმარებათ თქვენ კოდის დაწერაში და გიადვილებთ საქმეს. თქვენ დაგხვდებათ უკვე პროგრამული კოდის ნახევარი

დაწერილი და უბრალოდ თვენ მოგიწევთ მასში ჩაწეროთ ფუნქცია End რომელიც პროცესის დასრულებას ნიშნავს და პროგრამას გათიშავს. ანუ Exit ღილაკის პროგრამული ნაწილი უნდა გამოიყურებოდეს ასე:



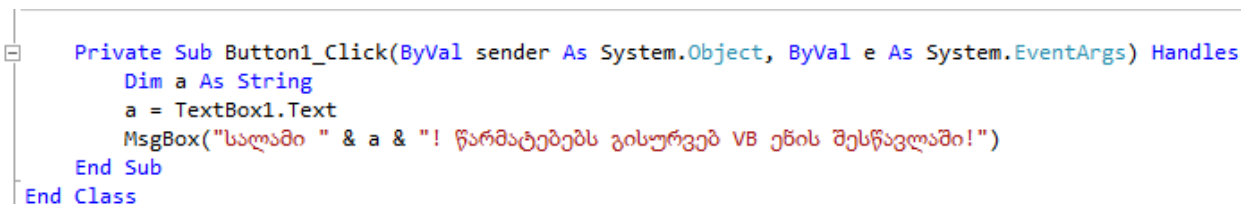
ამის შემდეგ ჩვენ გადავდივართ პროგრამის ძირითადი ნაწილის დაწერაზე, რომელიც უნდა განვითავსოთ OK ღილაკში. ვაწვებით OK ღილაკს ორჯერ სწრაფად და ვიწყებთ პროგრამის წერას.

პროგრამა უნდა დავწეროთ Private Sub-სა და End Sub-ს შორის, რომელიც Visual Basic 2010-მა ავტომატურად დაგვახვედრა გამზადებული.

ჩენი პროგრამა ასეთია:

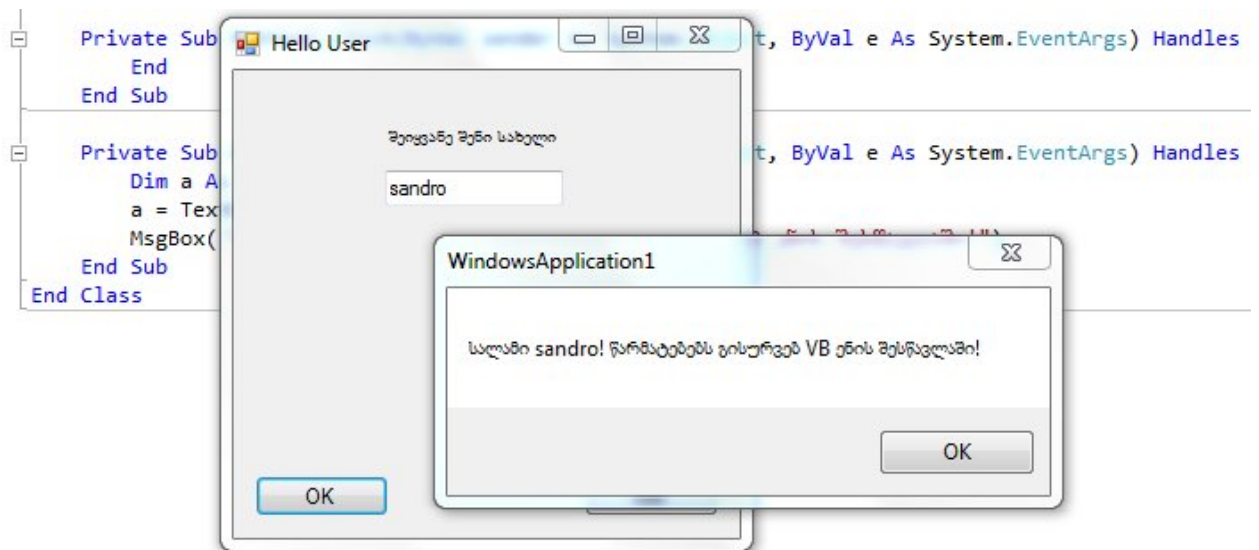
```
Dim a As String
a = TextBox1.Text
MsgBox("სალამი " & a & "! წარმატებებს გისურვებ VB ენის შესწავლაში!")
```

ხოლო მას უკვე OK ღილაკში განთავსებულს. უნდა ჰქონდეს ასეთი სახე:



ამის შემდეგ თქვენი პირველი პროგრამა უკვე მზადაა და მის გასაშვებად საჭიროა დააჭუროთ კლავიატურაზე F5-ს.

თუ ყველაფერი სწორად გააკეთეთ, პროგრამაში სახელის ჩაწერისას და OK ღილაკზე დაჭერისას, ის გამოიტანს ასეთ შეტყობინებას:



ახლა განვიხილოთ ჩვენი პროგრამული კოდი, რაიმე გაუგებარი რომ არ დარჩეს. პროგრამული კოდი ნაბიჯ-ნაბიჯ იშიფრება ასე:

- 1) შემოგვაქვს ტექსტური ცვლადი
- 2) ცვლადს ვანიჭებთ TextBox1-ში აკრეფილ ტექსტის მნიშვნელობას
- 3) შემდეგ კი, წინასწარ გამოზადებულ ტექსტთან ერთად გამოგვაქვს TextBox1-ში აკრეფილი ტექსტი ეკრანზე, MsgBox-ის საშუალებით,

მაშ ასე, პროგრამა უკვე ავაწყვეთ და გავამზადეთ. ახლა კი ჯერი მის შენახვაზე მიდგა. პროექტის შენახვა თუ გვინდა ჩვენ მაღლა მთავარ მენიუმში შევდივართ File-ში და ვაწვებით Save All-ს, რის შემდეგადაც გამოვა ფანჯარა სადაც შეგვყავს პროგრამის დასახელება და ის დირექტორია სადაც ვინახავთ პროექტს. ამის შემდეგ ვაწვებით OK ს და ყველაფერი გაკეთებულია.

რაც შეეხება თვითონ პროგრამას, როგორც ცალკე გამშვებ .exe ფაილს, კომპილაციისას პროგრამა მას ინახავს იმ დირექტორიაში სადაც წელან მივუთითეთ პროექტის შენახვაში. თქვენ უბრალოდ უნდა ნახოთ პროგრამა აღნიშნულ დირექტორიაში სადაც ექნება შემდეგი მდებარეობა: **Visual Studio 2010 > Projects > პროგრამის დასახელება > WindowsApplication1 > bin > Debug** და პროგრამის დასახელება. ამის შემდეგ ამ პროგრამას თქვენ უკვე გადმოაკოპირებთ სამუშაო მაგიდაზე და ყველაფერი რიგზეა.

ახლა ცოტა გავართულოთ ჩვენი პროგრამა და დავამატოთ ფუნქცია, რომლის მეშვეობითაც, როდესაც სახელი არ იქნება შეყვანილი და ცარიელი TextBox-ის შემთხვევაში დავაჭერთ OK-ს, პროგრამამ გამოგვიტანოს შეყობინება - “შეიყვანეთ სახელი”. ამისთვის გამოვიყენოთ If ოპერატორი, და შევიტანოთ პატარა ცვლილებები ჩვენს კოდში:

```

Dim a As String
a = TextBox1.Text
If a = "" Then
MsgBox("შეიყვანეთ სახელი")
Else
MsgBox("სალამი " & a & "! წარმატებებს გისურვებ VB ენის შესწავლაში!")
End If

```

პირველ რიგში კომპილატორი შეამოწმებს If ოპერატორის შემდგომ დაწერილ პირობას და თუ ის ჭეშმარიტია მაშინ შესრულდება Then-ის შემდეგ დაწერილი ოპერაცია. თუ პირობა მცდარია, მაშინ შესრულდება Else-ს შემდეგ დაწერილი ოპერაცია.

პირობაში წერია, რომ ცვლადი a არის ცარიელი ტექსტი, ამიტომ თუ ჩვენ არ დავწერთ არაფერს TextBox1-ში, მაშინ პირობა ჭეშმარიტი იქნება და ეკრანზე გამოვა შეტყობინება "შეიყვანეთ სახელი". ხოლო თუ ჩვენ ჩავწერთ რაიმეს TextBox1-ში, მაშინ პირობა მცდარი გამოვა და ამიტომ შესრულდება Else-ს შემდგომ დაწერილი ოპერაცია და ეკრანზე გამოვა მისალმების შეტყობინება.

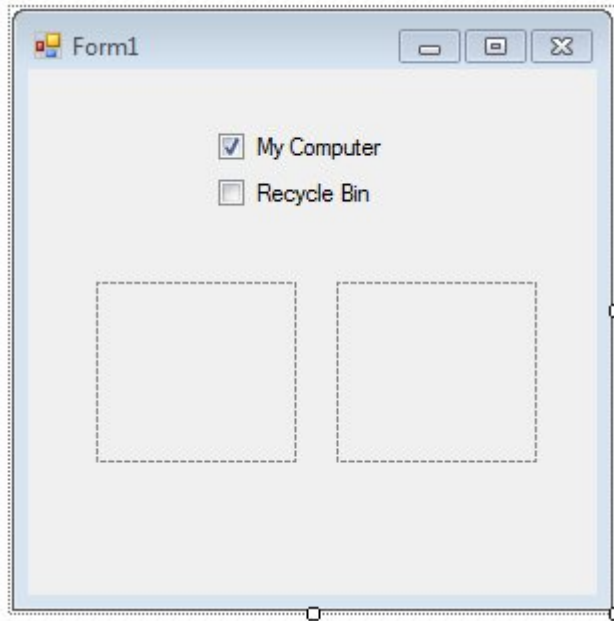
პროგრამები ToolBox-ის სხვადასხვა ელემენტებით

იმისთვის რომ კარგად გავერკვიოთ ToolBox-ის ელემენტებში, მათ დანიშნულებასა და გამოყენებაში, განვიხილოთ სხვადასხვა მაგალითები, რომლებშიც გამოვიყენებთ სხვადასხვა კონკრეტულ ელემენტებს ToolBox-იდან. დავიწყოთ ელემენტი CheckBox-ით.

#1. *CheckBox* ელემენტის გამოყენება

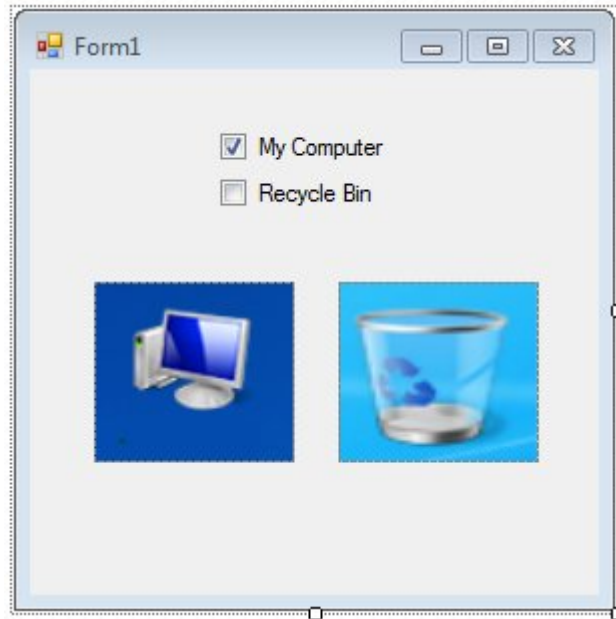
პირველ რიგში:

1. შექმენით ახალი Windows Forms Application პროექტი სახელწოდებით MyCheckBox
2. დააჭირეთ CheckBox-ის ღილაკს ToolBox-ში
3. გადმოიტანეთ CheckBox ელემენტი, პროგრამის საწყის ვიზუალურ ფორმაზე.
4. შემდეგ უკვე გადმოტანილი CheckBox ელემენტი დააკოპირეთ და მიღებული 2 CheckBox ელემენტი განათავსეთ ცენტრში, ერთი მეორის ქვეშ,
5. გამოვიყენოთ PictureBox-ის კონტროლი, ამისთვის დაამატეთ ვიზუალურ ფორმაზე ToolBox-იდან ორი ერთმანეთის ტოლი PictureBox-ი და განათავსეთ ისინი ერთმანეთის გვერდიგვერდ CheckBox-ების ქვევით და ორივე PictureBox-ის Properties ფანჯარაში მოძებნეთ SizeMode, რის შემდეგადაც, ორივეზე დააყენეთ StretchImage.
6. შეცვალეთ CheckBox-ზე სახელები Properties-იდან, პირველზე დააწერეთ ტექსტი “My Computer“ და მეორეზე დააწერეთ ტექსტი “Recycle Bin”. ასევე აირჩიეთ CheckBox1 და Properties ფანჯარაში მოძებნეთ Checked და დააყენეთ მანდ True.



7. მონიშნეთ პირველი PictureBox1 და დააჭირეთ Image-ს გასწვრივ სამ წერტილს, Properties ფანჯარაში. შემდეგ მონიშნეთ Local Recurce და დააჭირეთ Import-ს და შეიტანეთ My Computer-ის აიკონის სურათი (თუ სურათი არ გაქვთ, გააკეთეთ ის Print Screen-ის მეშვეობით)
8. ასევე გააკეთეთ მეორე PictureBox2-ზე, ოღონდ My Computer სურათის მაგივრად შეიტანეთ Recycle Bin-ის აიკონის სურათი და PictureBox2-ის Properties ფანჯარაში მოძებნეთ Visible და დააყენეთ მანდ False.

ამის შემდეგ თქვენი პროგრამა უნდა გამოიყურებოდეს დაახლოვებით ასე:



პროგრამის ვიზუალური ნაწილი ფაქტობრივად დასრულებულია და ახლა საქმე მიდგა მის პროგრამულ ნაწილზე. ჩვენი მიზანია, როდესაც მოვნიშნავთ My Computer-ს, პროგრამაში გამოჩნდეს My Computer-ის სურათი, ხოლო როდესაც მოვნიშნავთ Recycle Bin-ს, ეგეც ანალოგიურად. ხოლო როდესაც CheckBox-ებზე მონიშვნებს მოვხსნით, მაშინ პროგრამამ უნდა გააქროს სურათები. ორჯერ სწრაფად დავაკლიკოთ პირველ CheckBox-სს, ანუ My Computer-ს და გაიხსნება ფორმა სადაც უნდა ჩავსვათ პროგრამული კოდი Private Sub-სა და End Sub-ს შორის. დავიწყეთ პროგრამის წერა. ჩვენი პროგრამა გამოიყურება ასე:

```
If CheckBox1.CheckState = 1 Then
PictureBox1.Visible = True
Else
PictureBox1.Visible = False
End If
```

მოცემულ შემთხვევაში კომპილატორი შეამოწმებს მონიშნულია თუ არა პირველი CheckBox-ი და თუ მონიშნულია, მაშინ გამოაჩენს შესაბამის სურათს, ხოლო თუ არაა მონიშნული, მაშინ გააქრობს სურათს.

პირველი CheckBox-ი მზადაა, ახლა მეორესაც მივხედოთ. ახალი არაფრის მოგონება აღარ მოგვიწევს კოდის დასაწერად, ჩვენ უბრალოდ პატარათი ჩავასწოროთ პირველი CheckBox-ის კოდი და ყველაფერი რიგზეა. დავაჭიროთ ორჯერ სწრაფად მეორე CheckBox-სს ანუ Recycle Bin-ს და ჩავსვათ შიგ შემდეგი კოდი:

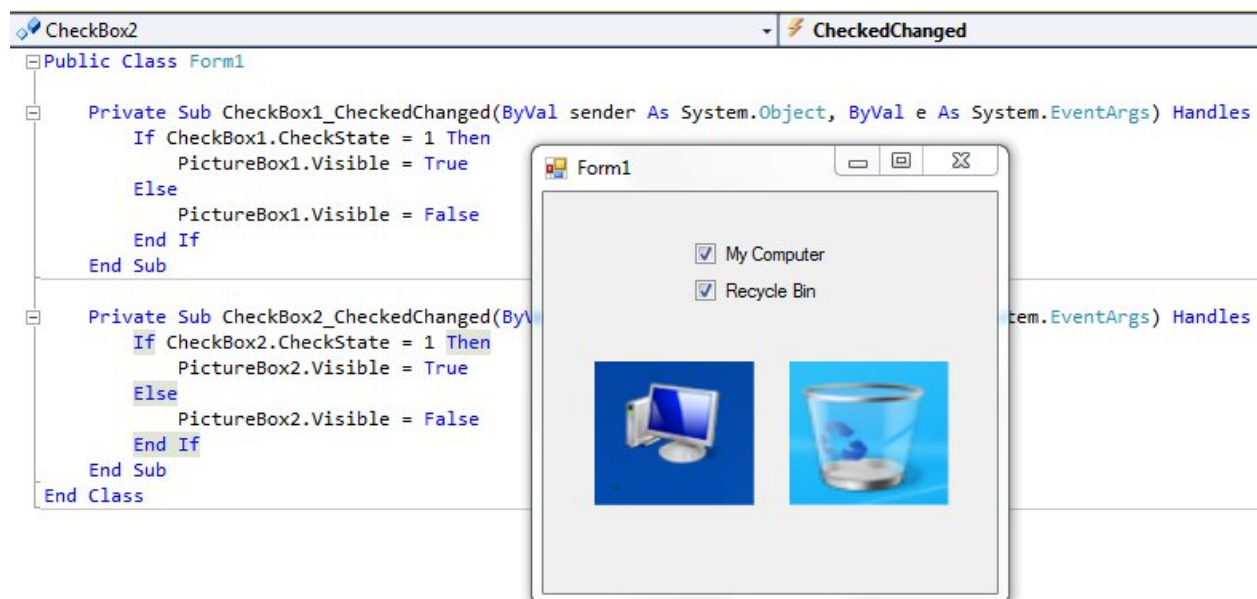
```
If CheckBox2.CheckState = 1 Then
```

```

PictureBox2.Visible = True
Else
PictureBox2.Visible = False
End If

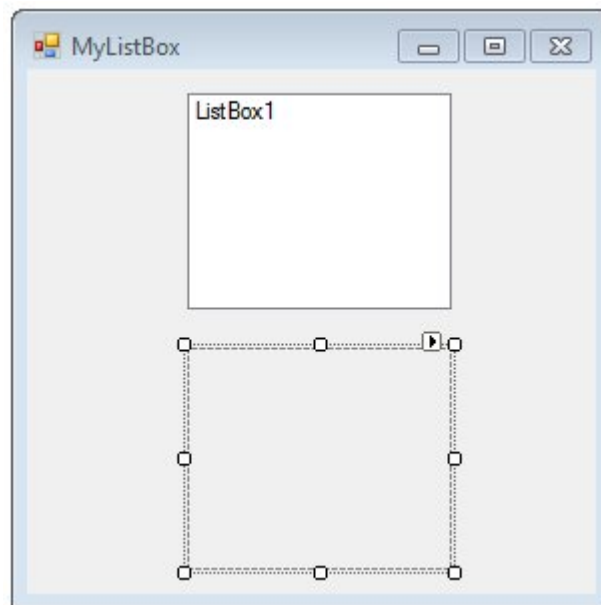
```

ამის შემდეგ ჩვენ მოვრჩით უკვე ჩვენ საქმეს და გავუშვათ პროგრამა შესრულებაზე, რისთვისაც ვაწვებით კლავიატურაზე F5-ს, რის შემდეგაც გამოვა ჩვენი პროგრამა და თუ ყველაფერი ზუსტად გავაკეთეთ მაშინ, როდესაც მოვნიშნავთ My Computer-ს, უნდა გამოჩნეს შესაბამისი სურათი, და როდესაც Recycle Bin-ს, ანალოგიურად. ხოლო როდესაც მოვხსნით მონიშვნებს, მაშინ სურათები უნდა გაქრეს. ჩვენი პროგრამა გამოიყურება დაახლოებით ასე:

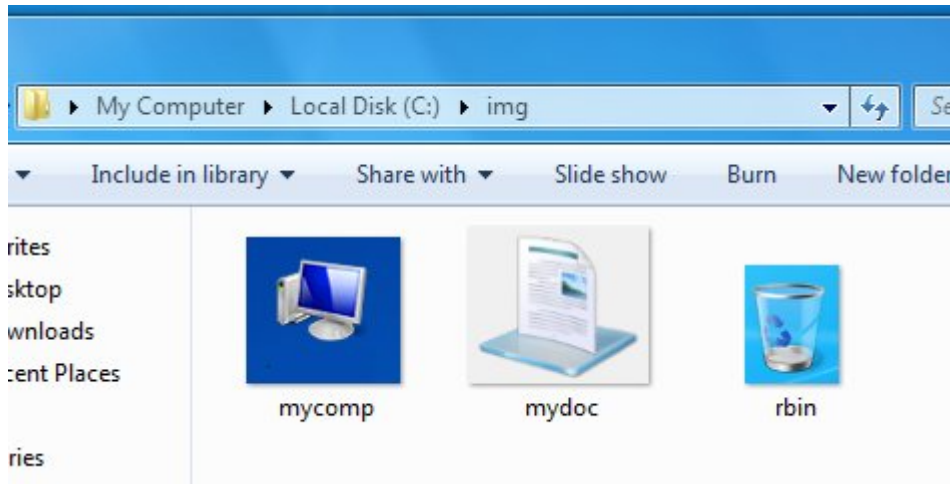


#2. *ListBox-ის გამოყენება*

1. შექმენით ახალი Windows Forms Application პროექტი სახელწოდებით MyListBox
2. ToolBox-იდან გადმოიტანეთ ListBox-ი პროგრამის საწყის გრაფიკულ ფორმაზე და განათავსეთ საშუალო ზომის ლისტბოქსი ფორმის ზედა ნახევარში.
3. გადმოიტანეთ იგივე ზომის PictureBox-ი ToolBox-იდან და განათავსეთ ListBox-ის ქვევით. ამის შემდეგ შედით PictureBox-ის Properties-ში და დააყენეთ `SizeMode – StretchImage`



4. შექმენით C:/ დისკზე საქალაქ სახელწოდებით “img” და ჩააგდეთ My Computer-ის, Recycle Bin-ის და My Document-ის სურათები სახელწოდებით: My Computer – mcomp.png ; Recycle Bin – rbin.png და My Documents – mdoc.png. (თუ სურათები არ გაქვთ გადაიღეთ დესკტოპზე, სკრინშოტის მეშვეობით. პ.ს. წიგნში მოყვანილია აღნიშნული სურათები მაგალითისთვის, რაღათქმაუნდა თქვენ არ ხართ შეზღუდული და შეგიძლიათ გამოიყენოთ ნებისმიერი სხვა თემატიკის სურათი)



5. შედით ListBox-ის Properties მენიუში და Items-ის გასწვრივ დააჭირეთ სამ წერტილს, რის შემდეგაც გაიხსნება ფანჯარა სადაც თქვენ ყოველ ახალ ხაზზე უნდა დაწეროთ სათითაო ელემენტი. My Computer, Recycle Bin და My Documents.
6. ჩვენი მიზანია, ლისტბოქსში ობიექტის არჩევისას, პროგრამამ გამოიტანოს ობიექტის შესაბამისი სურათი PictureBox-სში. ამისთვის ორჯერ სწრაფად ვაჭერთ ფორმაზე ListBox-ს და ვაკოპირებთ შიგ შემდეგ კოდს:

Select Case ListBox1.SelectedIndex

Case 0

PictureBox1.Image = System.Drawing.Image.FromFile _
("c:\img\mycomp.png")

Case 1

PictureBox1.Image = System.Drawing.Image.FromFile _
("c:\img\rbin.png")

Case 2

PictureBox1.Image = System.Drawing.Image.FromFile _
("c:\img\mydoc.png")

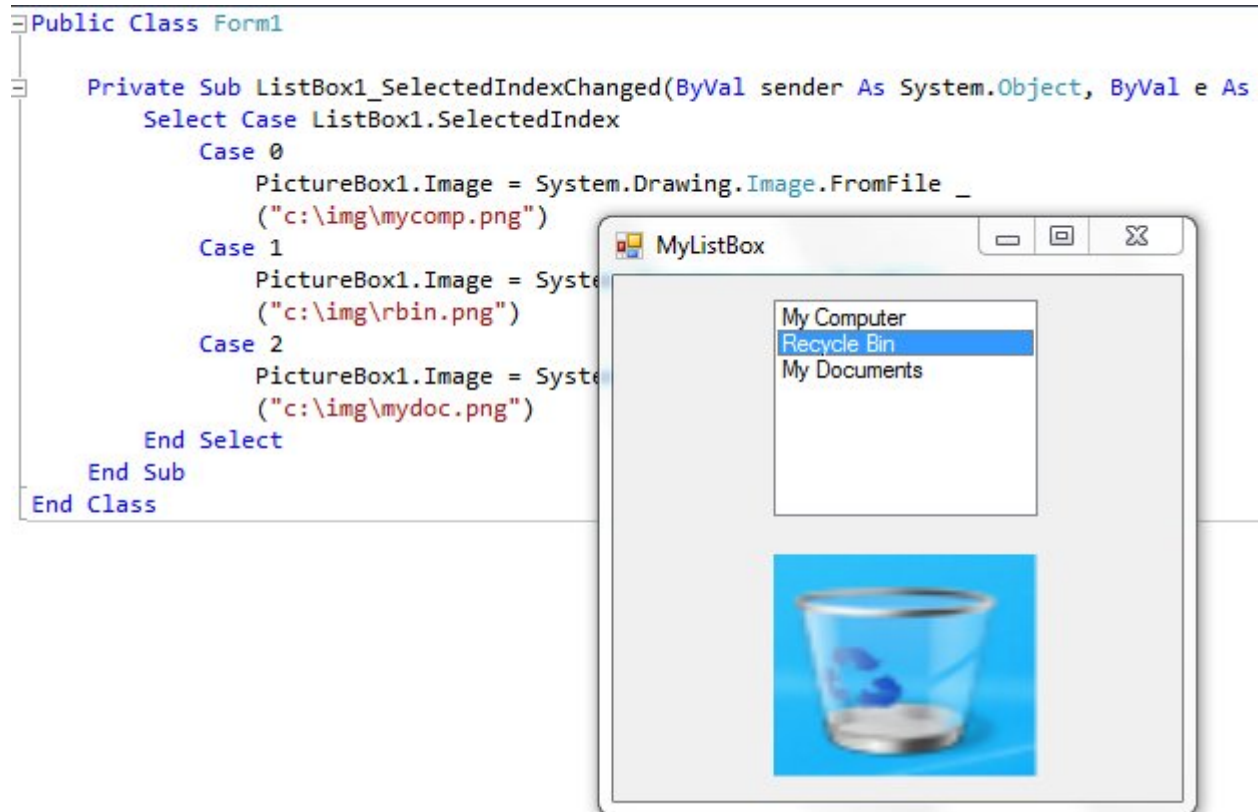
End Select

პროგრამა გადაამოწმებს თუ რომელი ობიექტია მონიშნული და ამის შემდეგ ჩართავს PictureBox-ში შესაბამის სურათს. Case წარმოადგენს ხაზებს ჩვენს ListBox-ში, ანუ პროგრამა აჩვენებს სურათს იმის მიხედვით თუ რომელი ხაზია მონიშნული.

ამის შემდეგ ჩვენი საქმე თითქმის შესრულებულია. მას მხოლოდ გაშვება და აკლია, რისთვისაც თქვენ უნდა დააჭიროთ Start ღილაკს პროგრამაში ან F5-ს

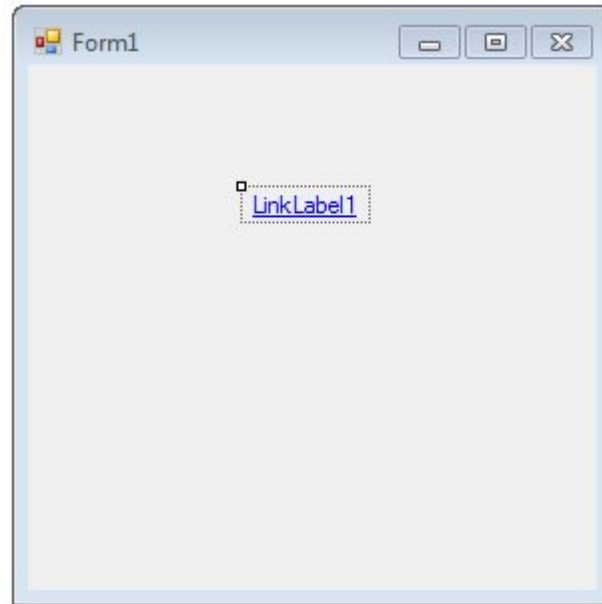
კლავიატურაზე, რის შემდეგაც გაეშვება ჩვენი პროგრამა და თუ ყველაფერი სწორად გავაკეთეთ, თითოეული ობიექტის არჩევისას პროგრამამ უნდა გვაჩვენოს შესაბამისი სურათი, მაგალითად როდესაც დავაჭერთ My Computers, უნდა გაიხსნას My Computers -ს სურათი და ანალოგიურად დანარჩენ ობიექტებზეც. რაღათქმა უნდა თქვენ შეგეძლოთ ეს პროგრამა გაგვეკეთებინათ სულ სხვა თემატიკის და სულ სხვა სურათებით.

ჩვენ პროგრამას, საბოლოო ჯამში, დაახლოვებით ექნება ასეთი სახე:



#3. *LinkLabel-ის გამოყენება*

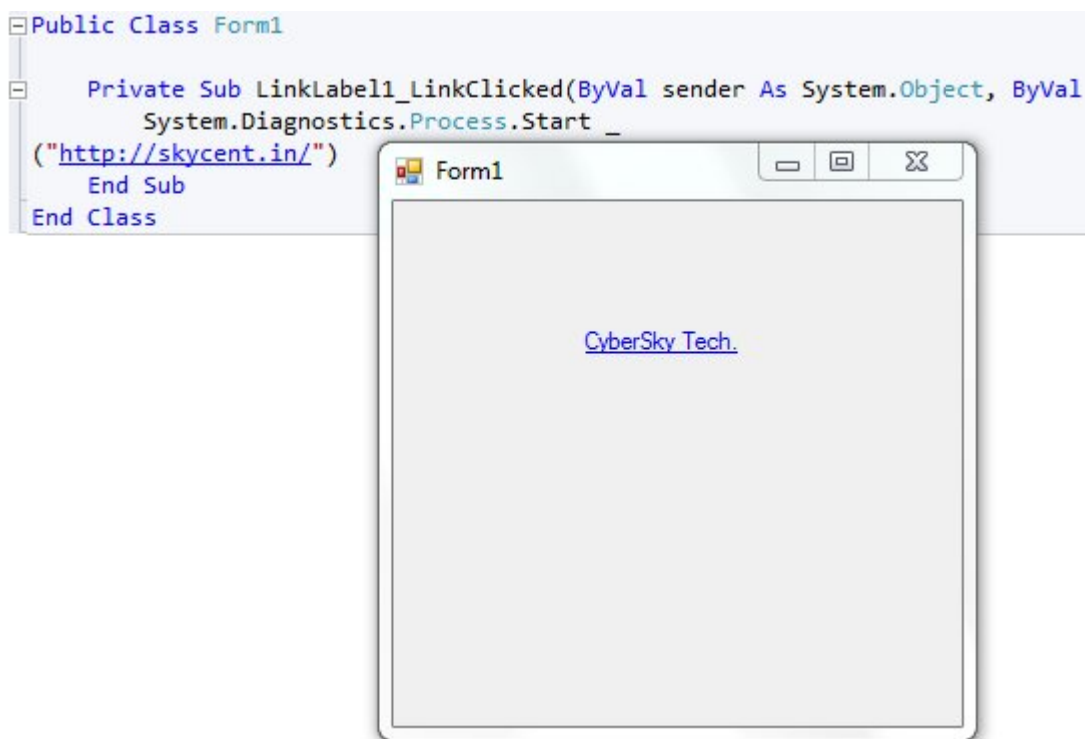
1. გახსენით ახალი Windows Form Application პროექტი Visual Basic 2010 ში და დაარქვით სახელი MyLinkLabel
2. ამის შემდეგ გაიხსნება პროგრამის გარსის საწყისი ჩარჩო, რაზეც დაამატეთ LinkLabel-ი ToolBox-იდან.



3. LinkLabel-ი ლურჯი ფერის, ხაზგასმული ტექსტის ფორმაა, რაზე დაკლიკვისასაც პროგრამამ უნდა შეგვიყვანოს ინტერნეტში იმ საიტზე, თუ რიცი ლინკიცაა ის.
4. შევდივართ LinkLabel-ის Properties ფანჯარაში და ვცვლით text-ს, სასურველი ტექსტით. (ჩვენს შემთხვევაში CyberSky Tech.)
5. იმისთვის რომ ლინკმა იმუშაოს, საჭიროა პროგრამის კოდურ ნაწილში განვათავსოთ შესაბამისი პროგრამულიკოდი, რისთვისაც ორჯერ სწრაფად ვაჭერთ ჩვენს LinkLabel-ს და ვწერთ პროგრამას, რომელიც ძალიან მარტივია (მაგალითი მოყვანილია საიტ skycent.in-ზე, რადთქმუნდა შესაძლებელია მის მაგივრად ნებისმიერი ლინკი ჩაიწეროს):

```
System.Diagnostics.Process.Start _  
("http://skycent.in/")
```

ამის შემდეგ უკვე ჩვენი პროგრამა მზადაა და მხოლოდ გაშვებაა საჭირო. ამისთვის ვაჭერთ კლავიატურაზე F5-ს და თუ ყველაფერი სწორად გავაკეთეთ პროგრამას ექნება დაახლოვებით ასეთი სახე:



ლინკზე დაჭერისას (ჩვენ შემთხვევაში CyberSky Tech.), პროგრამამ უნდა შეგვიყვანოს შესაბამის ლინკზე, რომელიც ჩვენ კოდში მივუთითეთ (<http://skycent.in/>).

ლინკის გახსნისას, პროგრამა იყენებს იმ ბრაუზერს, რომელსაც Default browser-ის ფუნქცია აქვს მინიჭებული თქვენს ოპერაციულ სისტემაში. იმისთვის რომ სხვა ჩვენთვის სასურველი ბრაუზერით გაიხსნას ლინკი, პატარა ცვლილების შეტანა დაგვჭირდება პროგრამულ კოდში. Internet Explorer-ის მაგალითზე, კოდი იქნება ასეთი:

```
System.Diagnostics.Process.Start("Iexplore.exe", _
    "http://skycent.in/")
```

LinkLabel-ის ფუნქციით ასევე შესაძლებელია არამარტო ლინკების, არამედ კონკრეტული ფაილების გახსნაც კომპიუტერში. ამისთვის ვუთითებთ პროგრამას რომლითაც უნდა გაიხსნას და გასახსნელი ფაილის მდებარეობას და ყველაფერი რიგზეა. მაგალითად:

```
System.Diagnostics.Process.Start("Winword.exe", _
    "c:\myletter.doc")
```

მათემატიკური ოპერატორები VB-ენაში

Visual Basic-ში მათემატიკური გამოსახულებების დასაწერად, სხვადასხვა ოპერატორები გამოიყენება:

"+" არის მიმატების ოპერაცია

"-" არის გამოკლების ოპერაცია

"*" არის გამრავლების ოპერაცია

"/" არის გაყოფის ოპერაცია

"^" არის ახარისხების ოპერაცია

"Mod" არის ნაშთით გაყოფის ოპერაცია

"Sqrt(n)" არის n-დან ფესვის ამოღების ოპერაცია

VB.NET-ში, ჩვეულებრივი VB-სგან განსხვავებით ფესვის ამოღება გადატანილია System-ში და მოთავსებულია კლას Math-ში, ამიტომ თუ გვინდა ამ ფუნქციით ვისარგებლოთ, უნდა დავწეროთ მისი სრული მისამართი ოპერაციიდან. ანუ ფესვის ამოღება VB.NET-ში n-დან იქნება ასე: System.Math.Sqrt(n)

მათემატიკური ოპერაციების გამოყენება პროგრამაში არცისე რთულია და მათი გამოყენება მარტივ მაგალითზე, ხდება ასე:

```
Dim a As Integer
```

```
a = 5 + 3
```

```
a = 6 - 4
```

```
a = 2 * 3
```

```
a = 10 / 5
```

```
a = 7 Mod 2
```

```
a = 7 ^ 2 'შვიდის კვადრატში აყვანა
```

```
a = System.Math.Sqrt(25) '25-დან ფესვის ამოღება
```

პირველი პროგრამა მათემატიკური ოპერაციებით

მათემატიკური ოპერატორების უკეთ გაგებისათვის, დავწეროთ პატარა მარტივი პროგრამა, რომელიც კვადრატულ განტოლებას ამოხსნის a, b და c ცვლადების შეტანისას.

პირველ რიგში ჩვენ გვჭირდება პროგრამაში ცვლადების შეტანის ოპერაცია რომლისთვისაც გამოიყენება InputBox-ი. მისი მეშვეობით შეგვიძლია რეალურ რეჟიმში მივანიჭოთ ცვლადს მნიშვნელობა პროგრამაში.

დავიწყოთ პროგრამის წერა. შემოვიტანოთ ცვლადი რომლებსაც შემდეგში მივანიჭებთ მნიშვნელობას.

```
Dim d As Double, a, b, c, x1, x2 As String
a = InputBox("შეიტანეთ a")
b = InputBox("შეიტანეთ b")
c = InputBox("შეიტანეთ c")
```

იმისთვის რომ ჩვენი პროგრამა იყოს სრულყოფილი, პატარა ცვლილებები შევიტანოთ კოდში, რადგან როდესაც ცარიელ InputBox-ზე დააჭერენ OK-ს არ მოხდეს ოპერაციის გაგრძელება, რათა პროგრამა არ გაიქრაშოს. ამისთვის ვიყენებთ ალგორითმული გადასვლის ოპერატორ y-ებს.

```
Dim d As Double, a, b, c, x1, x2 As String
y1:
    a = InputBox("შეიტანეთ a")
    If a = "" Then
        MsgBox("გთხოვთ შეიტანოთ რიცვი")
        GoTo y1
    End If
y2:
    b = InputBox("შეიტანეთ b")
    If b = "" Then
        MsgBox("გთხოვთ შეიტანოთ რიცვი")
        GoTo y2
    End If
y3:
    c = InputBox("შეიტანეთ c")
    If c = "" Then
        MsgBox("გთხოვთ შეიტანოთ რიცვი")
        GoTo y3
    End If
```

პროგრამა შეამოწმებს ცარიელია თუ არა InputBox-ი და მაგის მიხედვით გადავა შემდეგ ოპერაციაზე. თუ ცარიელია ის ხელახლა მოგვთხოვს რიცხვის შეტანას, თუ არა და უბრალოდ გააგრძელებს პროცესს.

იმისთვის რომ ამოვხსნათ კვადრატული განტოლება, ჩვენ გვჭირდება დისკრიმინანტის პოვნა. დისკრიმინანტი ტოლია $b^2 - 4ac$ რიცხვის კვადრატს გამოკლებულ ოთხჯერ ac რიცხვი ($b^2 - 4ac$). ვიჟუალ ბეისიკში ამგვარი ოპერაციის ჩაწერა არც ისე რთულია:

```
d = b ^ 2 - 4 * a * c
```

რის შემდეგაც გადავდივართ უკვე განტოლების ფესვების პოვნაზე, მაგრამ საჭიროა გავითვალისწინოთ ის შემთხვევები თუ როდის აქვს განტოლებას 1 ამონახსნი, როდის 2 და როდის არცერთი. ამისთვის ჩვენ ვიყენებთ if ოპერატორს და ვწერთ კოდს.

```
If d < 0 Then
    MsgBox("განტოლებას ამონახსნი არ აქვს")
ElseIf d = 0 Then
    x1 = -b / (2 * a) : MsgBox("X ტოლია: " & x1)
Else
    'X-ების გამოთვლა და ჩვენება თუ d > 0
    x1 = (-b + System.Math.Sqrt(d)) / (2 * a)
    x2 = (-b - System.Math.Sqrt(d)) / (2 * a)
    MsgBox("X1 ტოლია: " & x1 & vbCrLf & "X2 ტოლია: " & x2)
End If
```

პროგრამა შეამოწმებს დისკრიმინანტს და შემდეგ ისე გადავა სხვა ოპერაციაზე. თუ დისკრიმინანტი ნაკლები იქნება ნოლზე, ის გამოიტანს შეტყობინებას, რომ განტოლებას ამონახსნი არ გააჩნია. თუ დისკრიმინანტი იქნება 0-ის ტოლი, მაშინ ის გამოიტანს ეკრანზე ერთ ამონახსნს, ხოლო თუ მეტი იქნება 0-ზე, მაშინ პროგრამა გამოიტანს ეკრანზე 2 ამონახსნს.

ეს პროგრამა არის კონსოლური სახის და შესაბამისად Visual Studio-ში მისი გაშვება ხდება New project > Console Application-ში. აღნიშნული პროგრამის სრული სახე არის ასეთი:

Module Module1

```
Sub Main()
    'კვადრატული განტოლების ამომხსნელი პროგრამა
    Dim d As Double, a, b, c, x1, x2 As Object
y1:
    a = InputBox("შეიტანეთ a")
    If a = "" Then
        MsgBox("გთხოვთ შეიტანოთ რიცვი")
        GoTo y1
    End If
y2:
    b = InputBox("შეიტანეთ b")
    If b = "" Then
        MsgBox("გთხოვთ შეიტანოთ რიცვი")
        GoTo y2
```

```

End If
y3:
c = InputBox("შეიტანეთ c")
If c = "" Then
MsgBox("გთხოვთ შეიტანოთ რიცვი")
GoTo y3
End If
d = b ^ 2 - 4 * a * c 'დისკრიმინანტის გამოთვლა
'X-ების გამოთვლა
If d < 0 Then
MsgBox("განტოლებას ამონახსნი არ აქვს")
ElseIf d = 0 Then
x1 = -b / (2 * a) : MsgBox("X ტოლია: " & x1)
Else
'X-ების გამოთვლა და ჩვენება თუ d > 0
x1 = (-b + System.Math.Sqrt(d)) / (2 * a)
x2 = (-b - System.Math.Sqrt(d)) / (2 * a)
MsgBox("X1 ტოლია: " & x1 & vbCrLf & "X2 ტოლია: " & x2)
End If
End Sub

End Module

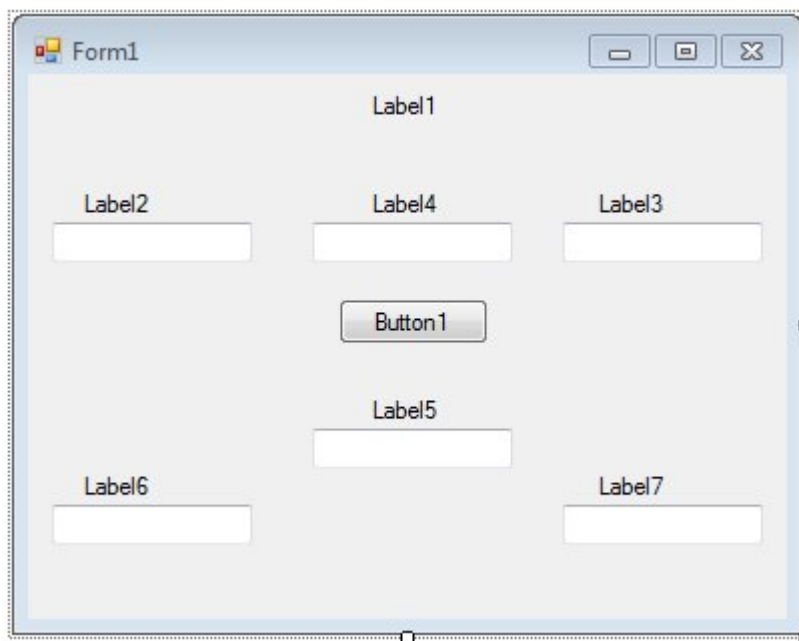
```

P.S. იმისთვის რომ არაფერი დარჩეს გაუგებარი, VB ენაში "vbCrLf"არის ახალ ხაზზე დაწერის ოპერაცია. ჩვენ ის პროგრამაში გამოვიყენეთ რათაგანტოლების ამონახსნები პროგრამას სხვადასხვა ხაზზე დაეწერა.

GUI პროგრამა მათემატიკური ოპერაციებით

წინა პროგრამა, რომელიც დაწერეთ მათემატიკური ოპერაციებით, იყო კონსოლური პროგრამის სახით, ანუ არ გააჩნდა მას გრაფიკული იერსახე. ჩვენ შეგვიძლია იგივე, კვადრატული განტოლების ამომხსნელი პროგრამა გადავწეროთ GUI (რაფიკალ სერ ნტერფაცე)-სთვისაც, სადაც უმნიშვნელო ცვლილებები დაგვჭირდება მხოლოდ კოდში, მარა საჭიროა გრაფიკული გარსის აწყობა/დამუშავება და მისი პროგრამულ კოდთან ინტეგრაცია. შევუდგეთ საქმეს:

1. გახსენით ახალი Windows Form Application პროექტი ვიშუალ ბეისიკში და დაარქვით მას სახელი: Quadratic Operations.
2. ამის შემდეგ გაიხსნება ჩვენი პროგრამის საწყისი გრაფიკული გარსი, სადაც საჭიროა Toolbox-იდან დავამატოთ შემდეგი ელემენტები: 7 ცალი Label, 1 Button და 6 ცალი TextBox. განათავსეთ ისინი როგორც სურათზეა მოცემული (სურათი მოყვანილია მაგალითისთვის, თქვენ რატემაუნდა შეგიძლიათ მოაწყოთ გრაფიკული იერსახე თქვენი გემოვნებისა და შეხედულებების მიხედვით).



3. ამის შემდეგ ვიწყებთ გრაფიკულ ელემენტებზე სახელების დაწერას:
 - Label1 - კვადრატული განტოლების ამომხსნელი პროგრამა
 - Label2 - შეიტანეთ a
 - Label4 - შეიტანეთ b
 - Label 3 - შეიტანეთ c
 - Button1 - გამოთვლა

- Label5 - დისკრიმინანტი
- Label6 – X1
- Label7 – X2

სახელების დარქმევა ხდება შესაბამისი ელემენტის Properties ფანჯარაში Text-ის შეცვლით. სახელებ დარქმეული გრაფიკული გარსი უნდა გამოიყურებოდეს დაახლოვებით ასე:

პროგრამის გრაფიკული მხარე უკვე გაკეთებულია, ახლა საჭიროა მის პროგრამულ მხარეზეც ვიზრუნოთ. რადგან წინა შემთხვევაში უკვე განვიხილეთ დისკრიმინანტის გამომთვლელი პროგრამის კოდი, ახლა აღარ დავწერ განხილვას და პირდაპირ ამ გრაფიკულ გარსზე მორგებულ კოდს შემოგთავაზებთ.

```
Dim d As Double, a, b, c, x1, x2 As String
a = TextBox1.Text
If a = "" Then
    MsgBox("გთხოვთ შეიტანოთ a")
End If
b = TextBox2.Text
If b = "" Then
    MsgBox("გთხოვთ შეიტანოთ b")
End If
c = TextBox3.Text
If c = "" Then
    MsgBox("გთხოვთ შეიტანოთ c")
```

Else

'დისკრიმინანტის გამოთვლა

$d = b^2 - 4 * a * c$: TextBox4.Text = d

'X-ების გამოთვლა

If $d < 0$ Then

MsgBox("განტოლებას ამონახსნი არ აქვს")

ElseIf $d = 0$ Then

$x1 = -b / (2 * a)$: TextBox5.Text = x1

TextBox6.Text = ""

Else

'X-ების გამოთვლა და ჩვენება თუ $d > 0$

$x1 = (-b + \text{System.Math.Sqrt}(d)) / (2 * a)$: TextBox5.Text = x1

$x2 = (-b - \text{System.Math.Sqrt}(d)) / (2 * a)$: TextBox6.Text = x2

End If

End If

იმისთვის რომ პროგრამაში შეიტანოთ კოდი, დააჭირეთ ღილაკ „გამოთვლა“-ს ორჯერ სწრაფად და ჩაწერეთ ეს კოდი Private Sub-სა და End Sub-ს შორის. თუ თქვენ ყველაფერს სოწორად გააკეთებთ, პროგრამის საბოლოო სახე იქნება დაახლოვებით ასეთი:

```
Public Class Form1
```

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
```

```
Dim d As Double, a, b, c, x1, x2 As String
```

```
a = TextBox1.Text
```

```
If a = "" Then
```

```
MsgBox("გთხოვთ შეიტანოთ a")
```

```
End If
```

```
b = TextBox2.Text
```

```
If b = "" Then
```

```
MsgBox("გთხოვთ შეიტანოთ b")
```

```
End If
```

```
c = TextBox3.Text
```

```
If c = "" Then
```

```
MsgBox("გთხოვთ შეიტანოთ c")
```

```
Else
```

```
'დისკრიმინანტის გამოთვლა
```

```
 $d = b^2 - 4 * a * c$  : TextBox4.Text = d
```

```
'X-ების გამოთვლა
```

```
If  $d < 0$  Then
```

```
MsgBox("განტოლებას ამონახსნი არ აქვს")
```

```
ElseIf  $d = 0$  Then
```

```
 $x1 = -b / (2 * a)$  : TextBox5.Text = x1
```

```
TextBox6.Text = ""
```

```
Else
```

```
'X-ების გამოთვლა და ჩვენება თუ  $d > 0$ 
```

```
 $x1 = (-b + \text{System.Math.Sqrt}(d)) / (2 * a)$  : TextBox5.Text = x1
```

```
 $x2 = (-b - \text{System.Math.Sqrt}(d)) / (2 * a)$  : TextBox6.Text = x2
```

```
End If
```

```
End If
```

```
End Sub
```

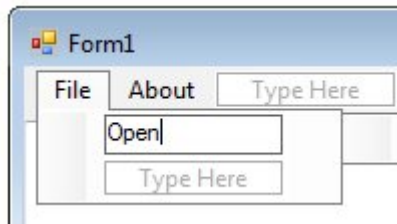
```
End Class
```

ფაილებთან მუშაობა

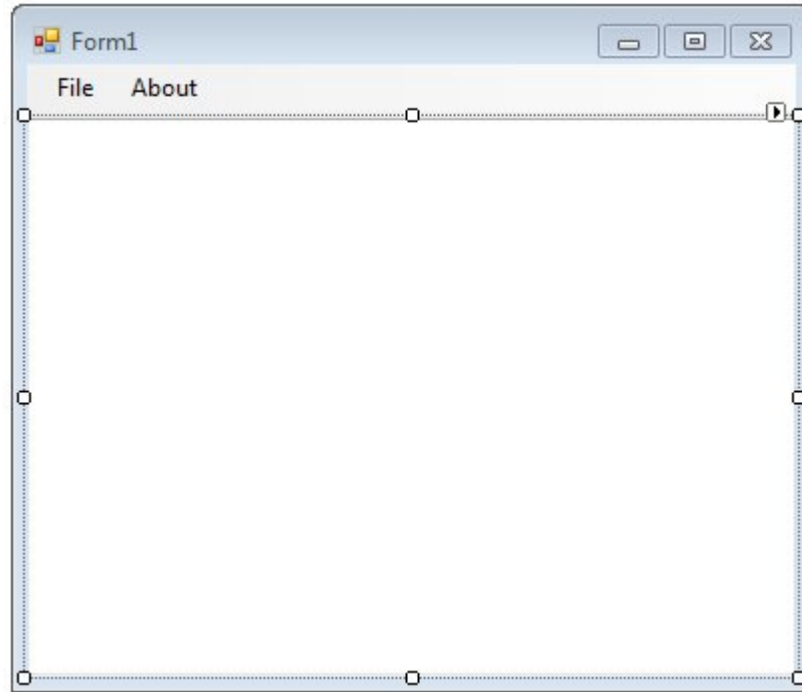
Visual Basic-ენა შესაძლებლობას გვაძლევს ვიმუშაოთ სხვადასხვა ტიპის ფაილებთან. გავხსნათ, დავამუშაოთ და შევინახოთ ისინი. იმისთვის რომ დამწყებებს ზოგადი წარმოდგენა შეეძენათ ამ საკითხის ირგვლივ, განვიხილოთ პატარა მაგალითი, რომელიც მიზნად ისახავს notepad-ის დაწერას vb-ენაში, რომელიც გახსნის და დაამუშავებს, ტექსტური სახის ფაილებს.

#1. ტექსტური ფაილების გახსნა

1. გახსენით ახალი Windows Form Application პროექტი Visual Basic-ში და დაარქვით მას სახელი MyNotepadBeta.
2. ამის შემდეგ გაიხსნება პროგრამის საწყისი გრაფიკული ჩარჩო, სადაც საჭიროა ToolBox-იდან დავამატოთ: OpenFileDialog, TextBox და MenuStrip.
3. როდესაც თქვენი პროგრამის ჩარჩოზე გაჩნდება MenuStrip ფორმა, დააჭირეთ Tupe Here-ს და დაწერეთ მანდ File, შემდეგ კი მაგის გვერდით დააწერეთ About, ხოლო File-ს ქვეშ დაწერეთ Open.



4. ამის შემდეგ, TextBox-ის Properties ფანჯარაში მონახეთ Multiline და მიუთითეთ მანდ True, რის შემდეგაც გაშალეთ TextBox-ი პროგრამის ჩარჩოს მთლიან დარჩენილ ნაწილზე.



5. ამის შემდეგ დააჭირეთ თქვენი ჩარჩოს თავზე, სადაც წერია Form1 და Text-ში დაწერეთ MyNotepad Beta.

ამით პროგრამის გრაფიკული ნაწილი უკვე შესრულებულია, ახლა საქმე მის პროგრამულ ნაწილზე მიდგა. პირველ რიგში ორჯერ სწრაფად დაკლიკეთ About-ზე რომელიც MenuStrip-ზეა და გახსნილ ფანჯარაში, Private Sub-სა და End Sub-ს შორის ჩაწერეთ შემდეგი კოდი.

```
MsgBox("My Notepad Beta" & vbCrLf & "Autor: 7th Ghost")
```

ახლა საქმე პროგრამის მთავარ კოდზე მიდგა, დააჭირეთ File-ს MenuStrip-ზე და შემდეგ დააჭირეთ Open-ს ორჯერ სწრაფად, და დავიწყეთ პროგრამის წერა:

```
Dim text As String = ""
OpenFileDialog1.Filter = "Text files (*.txt)|*.txt"
If OpenFileDialog1.ShowDialog() = DialogResult.OK Then 'ფაილის გახსნის
დიალოგი
    text = My.Computer.FileSystem.ReadAllText(OpenFileDialog1.FileName)
    TextBox1.Text = text 'ფაილის ჩვენება
End If
```

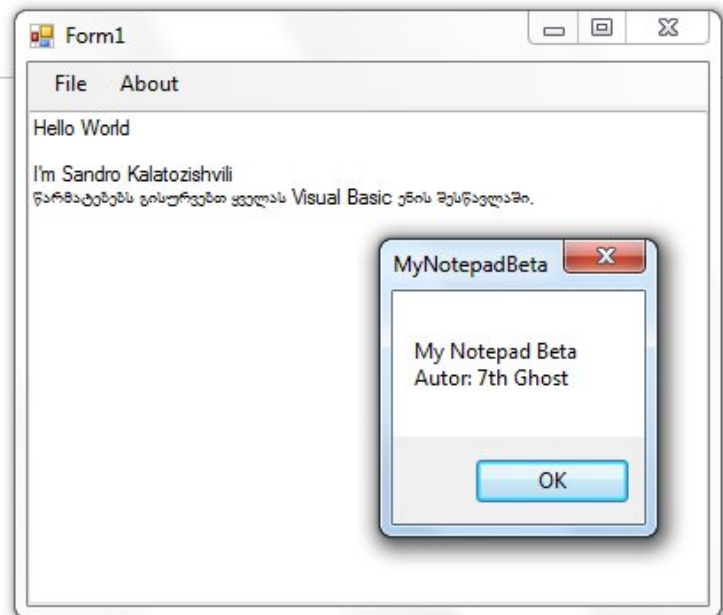
ამის შემდეგ ჩვენი პროგრამა ფაქტობრივად მზადაა და ელოდება გაშვებას. თუ ყველაფერი სწორად გააკეთეთ, პროგრამის საშუალებით შესაძლებელია გახსნათ

ნებისმიერი txt-გაფართოვების ტექსტური ფაილი რომელიც თქვენს კომპიუტერშია. ყველაფრის ბოლოს, თქვენი პროგრამა უნდა გამოიყურებოდეს დაახლოვებით ასე:

```
Public Class Form1

    Private Sub AboutToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles AboutToolStripMenuItem.Click
        MsgBox("My Notepad Beta" & vbCrLf & "Autor: 7th Ghost")
    End Sub

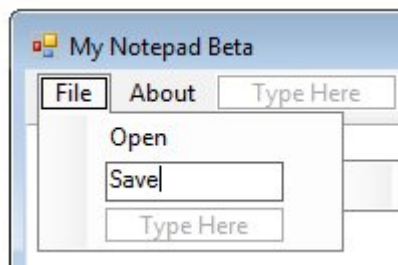
    Private Sub OpenToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles OpenToolStripMenuItem.Click
        Dim text As String = ""
        OpenFileDialog1.Filter = "Text files (*.txt)|*.txt"
        If OpenFileDialog1.ShowDialog() = DialogResult.OK Then 'ფაილის გახსნის დიალოგი
            text = My.Computer.FileSystem.ReadAllText(OpenFileDialog1.FileName)
            TextBox1.Text = text 'ფაილის ჩვენება
        End If
    End Sub
End Class
```



#2. ტექსტური ფაილების შენახვა

წინა მაგალითში, ჩვენ ზოგადად განვიხილეთ თუ როგორ ხდება Visual Basic-ენაში ფაილების გახსნა და დამუშავება, ტექსტური ფაილების მაგალითზე. ახლა განვიხილოთ უკვე გახსნილი, ან ჩვენი დამუშავებული მონაცემების ფაილში გამოტანის/შენახვის ფუნქცია. ამისთვის ჩვენ ახლის არაფრის გაკეთება არ დაგვჭირდება უბრალოდშევიტანოთ პატარა ცვლილებები წინა პროექტ MyNotepad Beta-ში და ყველაფერი რიგზეა.

1. გახსენით წინა პროექტი My Notepad Beta ვიშუალ ბეისიკში და ToolBox-იდან დაამატეთ SaveFileDialog-ი.
2. ამის შემდეგ, დააჭირეთ File-ს MenuStrip-ზე და Open-ის ქვეშ დააჭირეთ Type Here-ს, სადაც დაწერეთ Save.



ამით ჩვენ ფაქტობრივად მოვრჩით გრაფიკული იერსახის გადაკეთება, ახლა პროგრამის კოდზე მიდგა საქმე, რაც არც თუ ისე რთულია. ვაჭერთ File-ს და შემდეგ ორჯერ სწრაფად Save-ს და ვიწყებთ პროგრამის წერას.

```
SaveFileDialog1.Filter = "Text files (*.txt)|*.txt"
If SaveFileDialog1.ShowDialog() = DialogResult.OK Then
    'ფაილის მყარ დისკზე შენახვა
    My.Computer.FileSystem.WriteAllText( _
        SaveFileDialog1.FileName, TextBox1.Text, False)
End If
```

ამის შემდეგ ჩვენი პროგრამა მზადაა და ელოდება გაშვებას. პროგრამის მეშვეობით თქვენ შეგიძლიათ შეასრულოთ Notepad-ის ის უმარტივესი ქმედებები, რომლებიცაა: ფაილის გახსნა, დამუშავება, ტექსტის აკრეფა და ამ ყველაფრის შენახვა. თუ თქვენ ყველაფერი სწორად გააკეთეთ, პროგრამას შეეძლება გახსნას ნებისმიერი txt ფაილი თქვენს კომპიუტერში და დაამუშაოს ის. ასევე შეინახოს დამუშავებული ან თქვენს მიერ აკრეფილი ტექსტი, ახალ ან/და უკვე არსებულ ფაილში.

პირველი ნაწილის დასასრული

„გასრულდა ეს ამბავი ვითა სიზმარი ღამისა“

წიგნის პირველი ნაწილი დასრულებულია!

შეგახსენებთ, რომ წიგნი არის, ელექტრონული სახის, საცდელი გამოცემა.

წიგნის დაწერის ძირითადი მიზნებია:

- დახმარება დამწყები პროგრამისტებისათვის VB-ში ორიენტაციაში.
- საწყისი/საბაზისო ცოდნის შექმნა პროგრამირება/Visual Basic-ში
- ადაპტაცია ვიზუალურ/ობიექტზე ორიენტირებულ პროგრამირებაში
- ამასთანავე, წერის სახით ჩემი ცოდნის გაღრმავება, რადგან თავად დამწყები ვარ.

ეს არის წიგნის პირველი საცდელი ვერსია, ამასთანავე უფასო და ხელმისაწვდომი ყველასათვის. სავარაუდოდ მომავალში იქნება წიგნის სხვადასხვა ნაწილებიც, სადაც ახალ მასალასთან ერთად, უფრო ვრცლად და დაწვრილებით იქნება განხილული ამ წიგნში შემუშავებული და განხილული საკითხები.

წიგნში გამოყენებული ნებისმიერი სურათისა თუ ტექსტის გამოყენება, სპეციალური ნებართვის გარეშე, აკრძალულია და სხვისი საკუთრების მითვისება ისჯება კანონით!

All Rights Reserved © CyberSky Tech. 2011-12

<http://skycent.in/>

წარმატებებს გისურვებთ ცოდნის გაღრმავებაში!

- 7th Ghost -