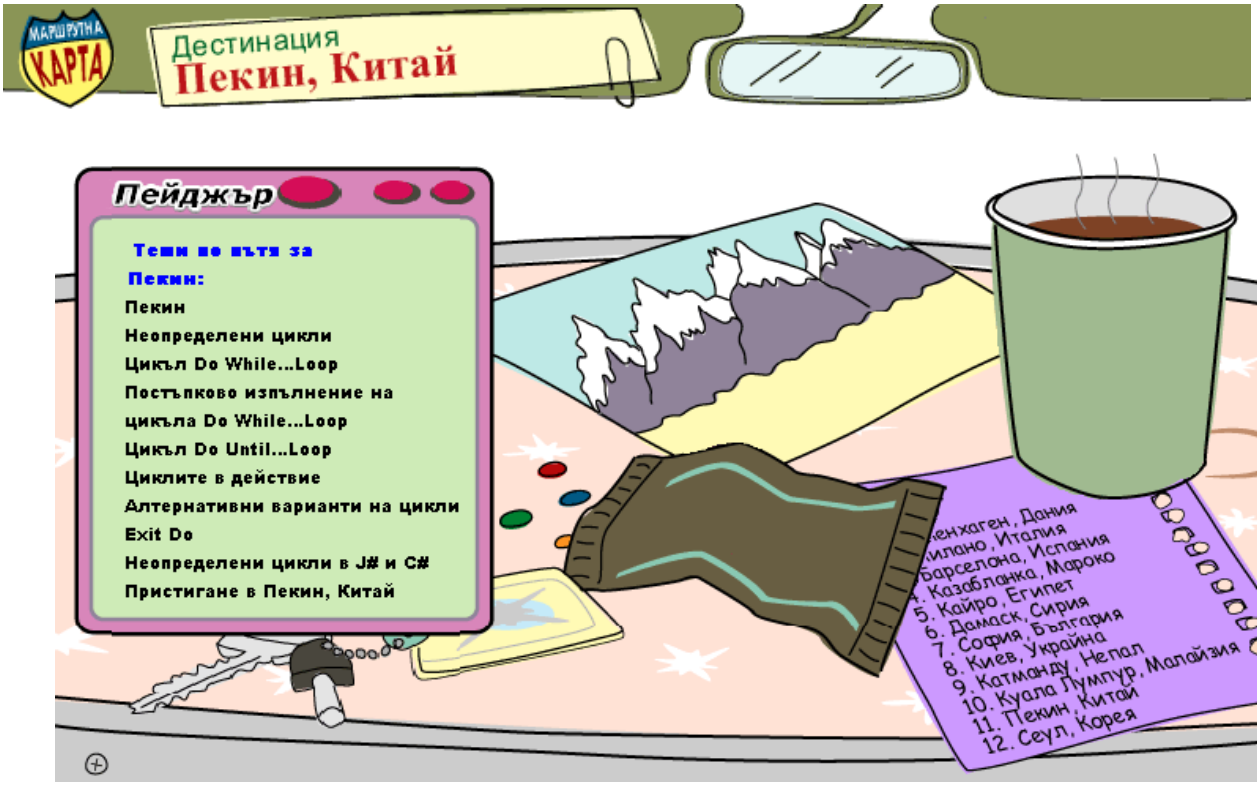
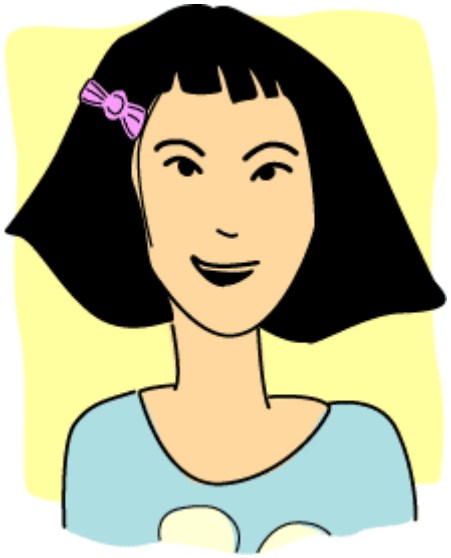

Маршрутна карта
Дестинация
Пекин, Китай
Пейджър
Теми по пътя за
Пекин:
Пекин
Неопределени цикли
Цикъл Do While...Loop
Постъпково изпълнение на
цикъла Do While...Loop
Цикъл Do Until...Loop
Циклите в действие
Алтернативни варианти на цикли
Exit Do
Неопределени цикли в J# и C#
Пристигане в Пекин, Китай
1. Казабланка, Мароко
5. Кайро, Египет
6. Дамаск, Сирия
7. София, България
8. Киев, Украйна
9. Катманду, Непал
10. Куала Лумпур, Малайзия
11. Пекин, Китай
12. Сеул, Корея

Неопределени цикли

Пекин, Китай

Клиф ще кара до нашата следваща дестинация – Пекин, Китай. Ще продължим по живописния път на циклите. По време на последното ни пътуване свършихме чудесна работа като ви запознахме с определените цикли – цикли, които се изпълняват определен брой пъти. На това пътуване отново ще се занимаваме с цикли, които обаче са по-различни. Те се наричат неопределени цикли, тъй като предварително не е известно колко пъти ще се изпълнят. Такива цикли спират изпълнението си, когато се изпълни определено условие.



Ще откриете, че неопределените цикли предоставят друга полезна техника, която всеки програмист трябва да знае как да използва. Съществуват случаи, когато определените цикли не са подходящи, и тогава трябва да се използват неопределените цикли.

Неопределени цикли

Често се случва да правим нещо отново и отново, докато нещо друго не се случи. Това “нещо друго” определя условие, при изпълнението на което спираме да действаме. Неопределените цикли се изпълняват, докато специфично условие стане истина или лъжа (True или False).

Например, миналата вечер обикаляхме около паркинга, докато не намерихме достатъчно голямо място, където да паркираме микробуса. През това време суши-ресторантът затвори и трябваше да потърсим друго място за хранене. Докато обикаляхме да търсим друг ресторант, стомахът ми ръмжеше от глад и не спря, докато не го напълних.

Аналогична е и ситуацията в програмирането. Понякога е необходимо някакъв код да се изпълнява многократно, докато определено условие е истина или лъжа. Например, кодът ви може да пита потребителя за парола отново и отново, докато тя/той не въведе правилна парола. Програмата ви може да изпълнява електронна музика, докато прави изчисления.

Това, което отличава неопределените цикли от определените, е че броя на итерациите за определените цикли е предварително ясен. Неопределените цикли се изпълняват докато не се удовлетвори дадено условие. Съществуват два типа такива цикли: "do while " и "do until". Циклите Do while се изпълняват докато условието е все още истина и спират, когато то стане лъжа. Циклите Do until се изпълняват докато условието е все още лъжа и спират, когато то стане истина. Ключов момент при конструирането на неопределените цикли е определянето на типа, който ще се използва.

Цикъл Do While...Loop

Цикълът "Do While...Loop" изпълнява блок от код, докато определено условие остава истина. След като условието стане лъжа, цикълът приключва и блокът от код повече не се изпълнява. Част от цикъла Do While...Loop е условието, което контролира изпълнението му. Ето синтаксиса му във Visual Basic:

```
Do While (condition)
    code statement 1
    code statement 2
    code statement, etc.
Loop
```

Забележете, че "Do", "While" и "Loop" са ключови думи. Те се визуализират в синьо. Условието се поставя в кръгли скоби. То може да бъде всякакъв булев израз, който връща като резултат стойност True или False, например X < 4. Кодът, който се изпълнява многократно се поставя между ключовите думи Do While и Loop. В началото на всеки пас от изпълнението на цикъла се проверява условието. Ако то е истина, кодът се изпълнява отново, а ако е лъжа – цикълът приключва.

Сега ще ви покажа как да използвате цикъла Do While...Loop. Създайте ново Windows приложение с име DoWhileLoop. Върху формата добавете бутон. Щракнете двукратно върху него, за да създадете обработчик на събитието Button1_Click. Добавете следния код:

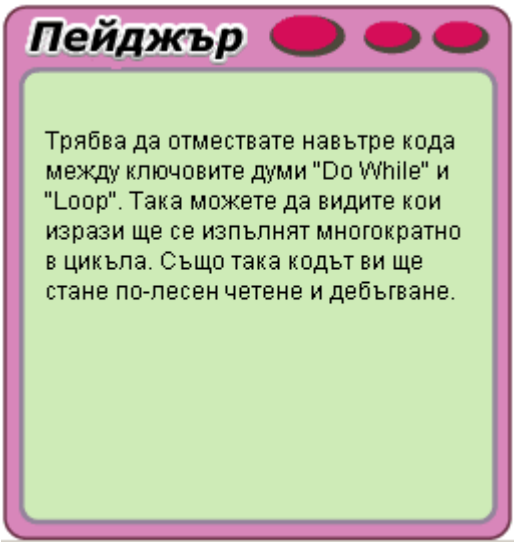
```
Dim WhileValue As Integer
WhileValue = 0
Do While (WhileValue < 2)
    MessageBox.Show(WhileValue)
    WhileValue = WhileValue + 1
Loop
```

Полезен съвет

Условието в цикъла Do може да бъде съставно, като например While (A > 1 and B < C)

Изградете и стартирайте проекта. Натиснете бутона. Появява се диалогов прозорец, който показва 0. Натиснете бутона OK. Диалоговият прозорец показва 1. Натиснете бутона OK. Връщате се към формата и диалогови прозорци повече не се показват.

Как работи този код? Декларирахме променлива с име WhileValue. Инициализирахме я със стойност 0. На всяка следваща итерация стойността ѝ се инкрементира. В началото на цикъла се проверява дали условието WhileValue < 2 е изпълнено. Ако това е така, се визуализира диалогов прозорец и променливата WhileValue се инкрементира. Ако условието е лъжа, цикълът приключва. Цикълът се изпълнява двукратно преди условието да стане лъжа. Първия път диалоговият прозорец показва 0, а вторият път – 1. Третият път стойността на WhileValue е 2. Булевият израз 2 < 2 е лъжа. Цикълът приключва и диалоговият прозорец не се показва.



Полезен съвет

Няма определена причина цикълът For да завършва с Next, а цикълът Do – с Loop. Посъветвайте учениците да гледат на това като на правило.

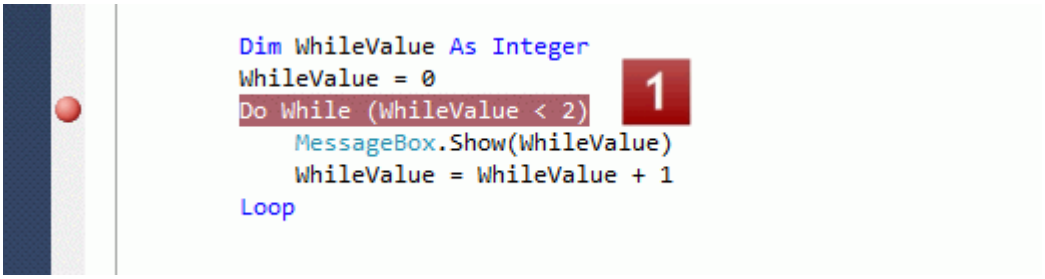
Постъпково изпълнение на цикъла Do While…Loop

Нека да изпълним кода, който написахте, като използваме средствата за дебъгване във Visual Studio 2010 . Изпълнението ще проследим стъпка по стъпка. Следвайте инструкциите:

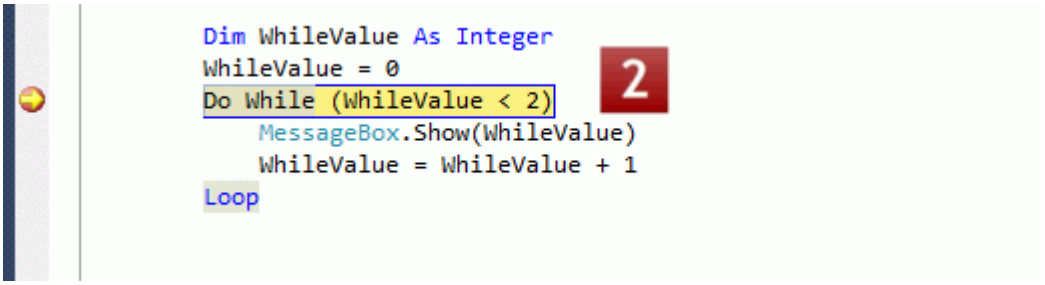
Полезен съвет

Учениците трябва да го направят, а не само да го прочетат.

1. Поставете точка за спиране на реда, който започва с ключовата дума Do While

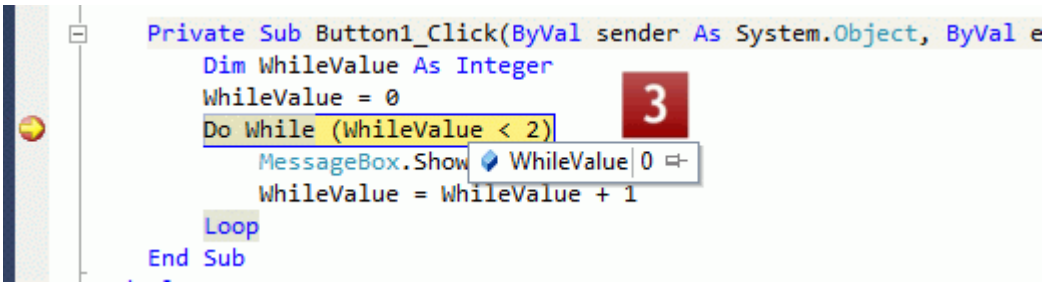


2. Стартирайте приложението. Кодът се изпълнява до точката на прекъсване, а редът, в който е поставена, се маркира в жълто.



3. Натиснете клавиша F11, за да изпълните маркирания ред.

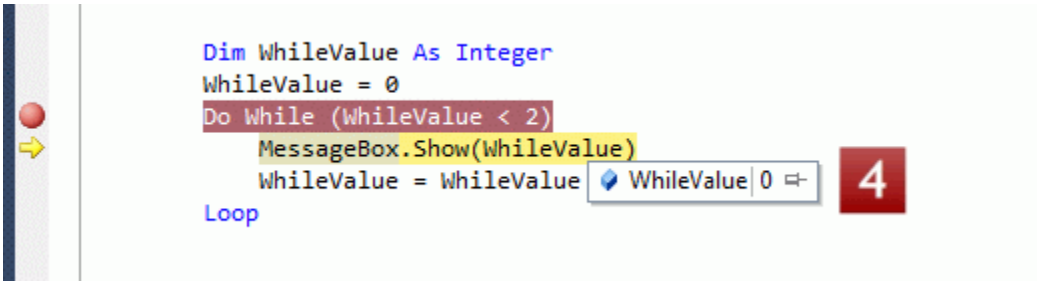
```
Do While (WhileValue < 2)
```



Преместете курсора над променливата WhileValue. Нейната стойност е 0.

4. Натиснете клавиша F11, за да изпълните маркирания ред.

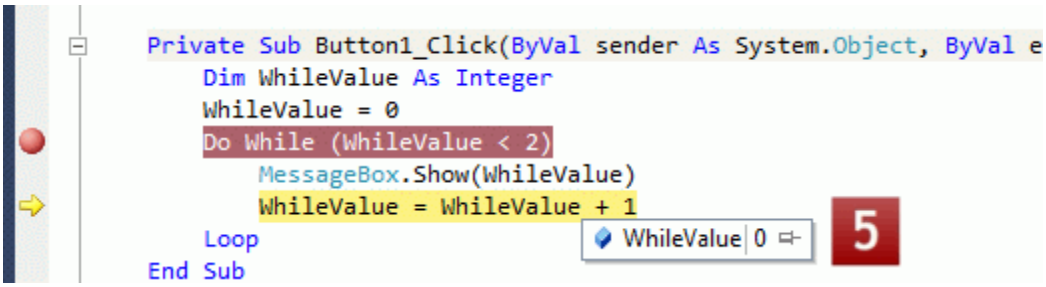
```
MessageBox.Show(WhileValue)
```



Диалоговият прозорец показва 0. Натиснете бутона OK.

5. Натиснете клавиша F11, за да изпълните маркирания ред.

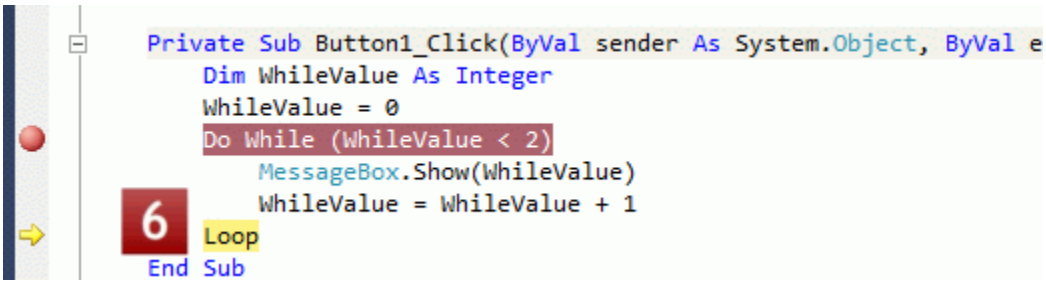
```
WhileValue = WhileValue + 1
```



Стойността на променливата WhileValue се инкрементира.

6. Натиснете клавиша F11, за да изпълните маркирания ред.

```
Loop
```



Цикълът отново се изпълнява.

7. Натиснете клавиша F11, за да изпълните маркирания ред.

`Do While (WhileValue < 2)`

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e
    Dim WhileValue As Integer
    WhileValue = 0
    Do While (WhileValue < 2)
        MessageBox.Show( WhileValue 1
        WhileValue = WhileValue + 1
    Loop
End Sub
```

7

Преместете курсора над променливата WhileValue. Нейната стойност е 1.

8. Натиснете клавиша F11, за да изпълните маркирания ред.

`MessageBox.Show(WhileValue)`

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e
    Dim WhileValue As Integer
    WhileValue = 0
    Do While (WhileValue < 2)
        MessageBox.Show(WhileValue)
        WhileValue = WhileValue + WhileValue 1
    Loop
End Sub
```

8

Диалоговият прозорец показва 1. Натиснете бутона OK.

9. Натиснете клавиша F11, за да изпълните маркирания ред.

`WhileValue = WhileValue + 1`

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e
    Dim WhileValue As Integer
    WhileValue = 0
    Do While (WhileValue < 2)
        MessageBox.Show(WhileValue)
        WhileValue = WhileValue + 1
    Loop                          WhileValue 1
End Sub
```

9

Стойността на променливата WhileValue се инкрементира и става 2.

10. Натиснете клавиша F11, за да изпълните маркирания ред.

Loop

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e
    Dim WhileValue As Integer
    WhileValue = 0
    Do While (WhileValue < 2)
        MessageBox.Show(WhileValue)
        WhileValue = WhileValue + 1
    Loop
End Sub
```

10

Цикълът отново се изпълнява.

11. Натиснете клавиша F11, за да изпълните маркирания ред.

Do While (WhileValue < 2)

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e
    Dim WhileValue As Integer
    WhileValue = 0
    Do While (WhileValue < 2)
        MessageBox.Show WhileValue 2
        WhileValue = WhileValue + 1
    Loop
End Sub
```

11

Преместете курсора над променливата WhileValue. Нейната стойност е 2. Условието WhileValue < 2 вече е лъжа, така е цикълът спира. Диалоговият прозорец не се показва и стойността на променливата WhileValue не се инкрементира.

12. Натиснете клавиша F11, за да изпълните маркирания ред.

End Sub

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e
    Dim WhileValue As Integer
    WhileValue = 0
    Do While (WhileValue < 2)
        MessageBox.Show(WhileValue)
        WhileValue = WhileValue + 1
    Loop
End Sub
```

12

Обработката на събитието, генерирано при натискане на бутона спира. Формата се показва отново.

Сега вече видяхте как работи цикъла Do While...Loop. Нека да видим как работи и цикъла Do Until...Loop.

Цикъл Do Until...Loop

Цикълът Do Until...Loop изпълнява блок от код докато някакво условие не стане истина. Условието може да бъде произволен булев израз, който се изчислява като истина или лъжа. Ето и синтаксиса на този цикъл във Visual Basic:

Полезен съвет

While започва с условие True. Until започва с условие False.

```
Do Until (condition)
    code statement 1
    code statement 2
    code statement etc.
Loop
```

Думите "Do", "Until" и "Loop" са ключови думи. Те са оцветени в синьо. Условието се проверява в началото н цикъла. Ако то все още е лъжа, блокът от код се изпълнява. Когато условието стане истина, цикълът приключва.

Сега ще ви покажа пример, който използва цикъла Do Until...Loop. Готови ли сте да програмирате! Създайте ново Windows приложение с име DoUntilLoop. Върху формата добавете бутон. Щракнете двукратно върху него, за да създадете обработчик на събитието Button1_Click. Добавете следния код:

```
Dim UntilValue As Integer
UntilValue = 0
Do Until (UntilValue > 1)
    MessageBox.Show(UntilValue)
    UntilValue = UntilValue + 1
Loop
```

Изградете и стартирайте проекта. Натиснете бутона. Появява се диалогов прозорец, който показва 0. Натиснете бутона OK. Диалоговият прозорец показва 1. Натиснете бутона OK. Цикълът приключва и формата се показва отново.

Как работи цикълът Do Until...Loop? Първо, декларираме променлива с име UntilValue и я инициализираме с 0. Тя се използва, за да се провери дали цикълът трябва да се изпълни отново. На всяка итерация от цикъла стойността на променливата UntilValue се инкрементира. Изразът Do Until съдържа условие, което проверява дали стойността на променливата UntilValue е по-голяма от 1. На първия пас стойността на променливата UntilValue е 0. Булевият израз е лъже, така че блокът от код се изпълнява. Диалоговият прозорец се визуализира и стойността на променливата UntilValue се инкрементира. На втория пас стойността на променливата UntilValue е 1. Булевият израз все още е лъжа и блокът от код отново се изпълнява. На третия пас стойността на променливата UntilValue е 2. Булевият израз 2 > 1 е истина и цикълът спира.

Циклите в действие

А сега да разгледаме няколко примера с циклите Do While...Loop и Do Until...Loop, които използват булеви изрази и не зависят пряко от стойността на брояча на цикъла. Първият пример използва цикъла Do While...Loop с булев израз, който проверява дали стойността на определена променлива е все още по-голяма или равна на нула. Когато стойността на тази променлива стане по-малка от нула, цикълът приключва.

Създайте ново Windows приложение с име LoopsInAction. Върху формата добавете бутон. Задайте стойност "CountDown" за свойството Text на бутона. Щракнете двукратно върху него, за да създадете обработчик на събитието Button1_Click. Добавете следния код:

```
Dim LoopIndex As Integer = 0
Dim WhileController As Integer = 167
Do While (WhileController >= 0)
    LoopIndex = LoopIndex + 1
    WhileController = WhileController - (23 *
LoopIndex)
Loop
MessageBox.Show(WhileController)
```

Изградете и стартирайте проекта. Натиснете бутона "CountDown". Появява се диалогов прозорец, който показва 63. Как работи този код? Декларирахме две променливи, LoopIndex и WhileController, и ги инициализирахме. Променливата LoopIndex се инкрементира всеки път, когато цикълът се изпълнява. Заедно с това се изчислява новата стойност на променливата WhileController като се използва определен израз. Новата стойност на променливата WhileController се получава като стойността на променливата LoopIndex се умножи по 23 и се извади от текущата стойност на променливата WhileController. Всеки път, когато цикълът се изпълнява, стойността на променливата WhileController намалява. Трудно е да се предскаже колко пъти ще се изпълнява цикълът преди стойността на променливата WhileController да стане по-малка от 0 и цикълът да приключи. Това няма значение, тъй като булевият израз, който контролира цикъла, използва стойността на променливата WhileController, а не на променливата LoopIndex.

Вторият пример използва цикъла Do While...Loop с булев израз, който проверява дали определен радио-бутон е маркиран. Друг радио-бутон се маркира на всяка итерация от цикъла.

В приложението LoopsInAction добавете втори обикновен бутон Задайте стойност "While" за свойството Text на бутона. Добавете три радио-бутона. Щракнете двукратно върху обикновения бутон, за да създадете обработчик на събитието Button2_Click. Добавете следния код:

```
Dim LoopIndex As Integer
= 0
RadioButton1.Checked =
True
Do While
(RadioButton3.Checked =
False)
    LoopIndex = LoopIndex + 1
    If LoopIndex = 1 Then
        RadioButton1.Checked = True
    ElseIf LoopIndex = 2 Then
        RadioButton2.Checked = True
    ElseIf LoopIndex = 3 Then
        RadioButton3.Checked = True
    End If
Loop
MessageBox.Show(LoopIndex)
```

Изградете и стартирайте приложението. Забележете, че първият радио-бутон е селектиран. Натиснете бутона "While". Третият радио-бутон се маркира т кода и се появява диалогов прозорец с номера на итерацията от цикъла. Ако очите ви са достатъчно бързи или ако използвате бавен компютър, ще видите, че вторият радио-бутон се маркира при изпълнението на цикъла.

Ето какво се случва. Декларирахме променлива с име LoopIndex, която съхранява броя на изпълненията на цикъла.Тя обаче, не се използва в булевия израз, който контролира цикъла. Вместо това променливата определя кой от радио-бутоните да се селектира на всяка итерация от цикъла. Булевият израз на цикъла Do While...Loop проверява дали третият радио-бутон е маркиран. Когато това се изпълни, цикълът приключва.

Третият пример използва цикъла Do Until...Loop. Булевият израз сравнява свойството Text на текстово поле със символна променлива, която се променя на всяка итерация от цикъла. Върху формата добавете текстово поле. Не променяйте стойността на свойството му Text. Добавете трети бутон и задайте стойност "TextMatch" за свойството му Text. Щракнете двукратно върху бутона, за да създадете обработчик на събитието Button3_Click. Добавете следния код:


```
Dim MatchText As String = ""
Dim LoopIndex As Integer = 0
Do Until (MatchText =
TextBox1.Text)
    LoopIndex = LoopIndex + 1
    If LoopIndex = 2 Then
        MatchText = "Text"
    End If
    If LoopIndex = 3 Then
        MatchText = "Box1"
    End If
    If LoopIndex = 4 Then
        MatchText = "TextBox1"
    End If
Loop
MessageBox.Show("Number " & LoopIndex & " is a match.")
```

Полезен съвет

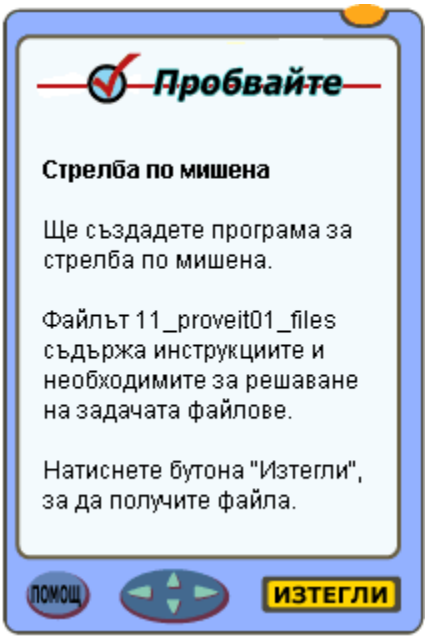
Името на променливата LoopIndex е произволно избрано. То може да бъде LoopCounter, LoopVariable, X, или някакво друго.

Изградете и стартирайте приложението. Натиснете бутона "Text Match". Появява се диалогов прозорец, който показва стойността на променливата LoopIndex.

Кодът използва цикъла Do Until...Loop. Променливата LoopIndex определя стойността на променливата MatchText. Булевият израз сравнява променливата MatchText със свойството Text на текстовото поле. На четвъртия пас от изпълнението на цикъла променливата LoopIndex има стойност 4, а променливата MatchText – "TextBox1". Булевият израз става истина, когато е изпълнено условието "TextBox1"="TextBox1", при което цикълът приключва.

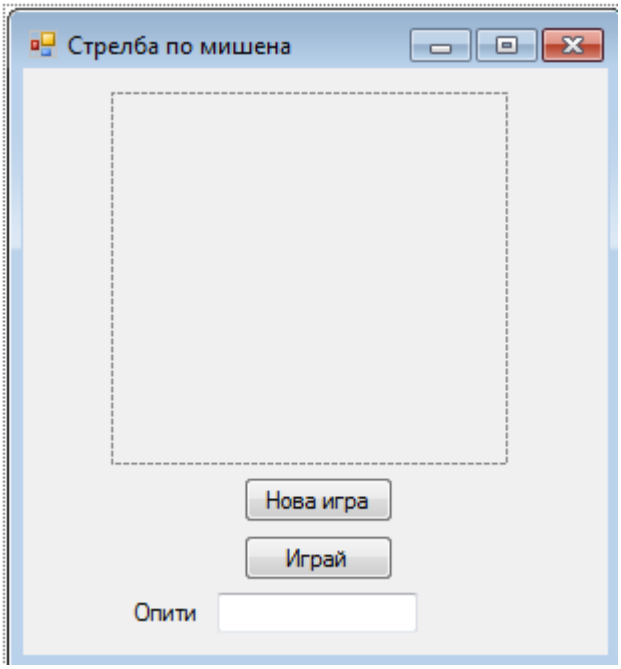
Какво би станало, ако стойността на свойството Text на текстовото поле беше установена на "The Purple Pit". Можете ли да добавете още един оператор "If..Then", така че кодът да продължи да работи?

11 Пробвайте 01 Стрелба по мишена



Винаги съм искал да посетя стрелбище. Стрелбата по мишена е забавна, но мисля, че компютърния й вариант е по-безопасен. Нека да създадем игра, в която компютърът ще опитва да уцели мишената.

Използвайте кода и формата в шаблон 01 от секция 11. Формата изглежда по следния начин:



Предоставяме ви изходния код за бутона ”Нова игра”, тъй като той съдържа графични аспекти, които не са включени в курса. Вашата задача е да напишете кода за бутона “Играй”.

Целта на играта е да преброите броя на опитите, необходими за да се уцели точката в центъра на мишената. Тази точка е с размер 1 пиксел. Тъй като полето, което визуализира картинката на мишената, е с размер 200 x 200 пиксела, то центъра на мишената е с координати 100 x 100.

Генерирайте случайно число за целочислената променлива X, която се използва за определяне на ширината. Генерирайте случайно число за целочислената променлива Y, която се използва за определяне на височината. Кодът за генериране на случайно число в диапазона от 0 до 200 е следния:

```
Dim MyRandomGenerator As System.Random
```

```
MyRandomGenerator = New System.Random
Dim RanNum As Integer

 ' Generate random value between 0 and 200 - not
including 200.
 RanNum = MyRandomGenerator.Next(0, 200)
```

Дефинирайте цикъл, който ще приключи, когато X и Y станат равни на 100.

Пребройте броя на итерациите в цикъла. Визуализирайте резултата в текстово поле.

Следният код изчертава точката, която обозначава изстрела върху мишената:

```
g.DrawEllipse(MyPen, New Rectangle(X, Y, 1, 1))
```

Ако програмата работи правилно, я покажете на учителя си.

Полезен съвет

Ще се появят въпроси. Вижте обяснителните бележки за очакваните въпроси.

Алтернативна варианти на цикли

Циклите Do While...Loop и Do Until...Loop могат да бъдат структурирани в алтернативни варианти, при които условието е в края на цикъла, вместо в началото му. Така блокът от код се изпълнява поне веднъж. След това се проверява условието. Алтернативните варианти са представени от циклите Do...Loop While и Do...Loop Until. Ето пример с цикъла Do...Loop While:

```
Dim WhileValue As Integer
WhileValue = 0
Do
    MessageBox.Show(WhileValue)
    WhileValue = WhileValue + 1
Loop While (WhileValue < 2)
```

Полезен съвет

Поставянето на условието в края на цикъла гарантира изпълнението му поне веднъж

Ето пример и с цикъла Do...Loop Until:

```
Dim UntilValue As Integer
UntilValue = 0
Do
    MessageBox.Show(UntilValue)
    UntilValue = UntilValue + 1
Loop Until (UntilValue > 1)
```

Забележете, че и в двата случая условието се записва с ключовата дума Loop, а не с ключовата дума Do. Кодът между ключовите думи Do и Loop се изпълнява поне веднъж. След това се проверява условието, за да се определели дали цикълът ще се повтори. В примера по-горе ако променливата UntilValue е инициализирана с 5, диалоговият прозорец ще покаже 5 преди да е направена проверката на условието.

Exit Do

Спомняте ли си, когато Клиф ви показа как да прекъсвате цикъла For...Next като използвате оператора Exit For? Е, по подобен начин можете да прекъсвате и циклите Do While...Loop, Do Until...Loop, Do...Loop While и Do...Loop Until. Вместо оператора "Exit For", обаче се използва оператора "Exit Do". За да се управлява изпълнението на оператора Exit Do, се използва булев израз, дефиниран с оператора If...Then. Ако булевият израз е истина, операторът Exit Do се изпълнява и цикълът приключва. Използвайте този механизъм, когато искате да излезете от цикъла преди да е удовлетворено условието му за приключване.

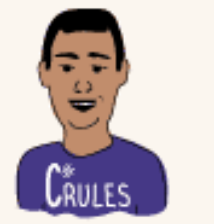
Да напишем код за изход от цикъла Do Until...Loop. Създайте ново Windows приложение с име ExitDo. Върху формата добавете бутон. Щракнете двукратно върху него, за да създадете обработчик на събитието Button1_Click. Добавете следния код:

```
Dim UntilValue As Integer
UntilValue = 0
Do Until (UntilValue > 10)
    MessageBox.Show(UntilValue)
    UntilValue = UntilValue + 1
    If UntilValue = 5 Then
        Exit Do
    End If
Loop
MessageBox.Show("Exit Value= " & UntilValue)
```

Изградете и стартирайте проекта. Натиснете бутона. На всеки пас от изпълнението на цикъла стойността на променливата UntilValue се инкрементира и диалогов прозорец показва номера на текущата итерация (0,1,2,3 и 4). Натискайте бутона OK всеки път. Когато променливата UntilValue стане 5, условието на оператора If...Then се удовлетворява. Изпълнява се оператора Exit Do. Цикълът приключва, а изпълнението на кода продължава от първия ред след оператора Loop Until. Този ред показва диалогов прозорец със съобщение "Exit Value= 5". Натиснете бутона OK, за да се върнете към формата.

Същият Exit Do оператор може да се използва и в цикъла Do While...Loop, при което цикълът отново ще приключи преди булевият израз от условието му да се изчисли като лъжа.

Неопределени цикли в C#



Нека да разгледаме неопределените цикли в C#.
В C# можете да напишете код, който има същото предназначение както циклите Do While...Loop и Do...Loop While във Visual Basic. C# няма алтернативни варианти за циклите Do Until...Loop или the Do...Loop Until.

Ето код на C#, който използва цикъла while (ключовите думи while и loop не започват с главни букви). Условието за прекратяване на цикъла се проверява в неговото начало:

```
int WhileValue=0;
while (WhileValue<5)
{
    MessageBox.Show(WhileValue.ToString());
    WhileValue=WhileValue+1;
}
```

Ето код на C#, който използва цикъла do...while (ключовите думи do, while и loop не започват с главни букви). Условието за прекратяване на цикъла се проверява в края на цикъла:

```
int WhileValue=0;
do
{
    MessageBox.Show(WhileValue.ToString());
    WhileValue=WhileValue+1;
}
while(WhileValue)
```

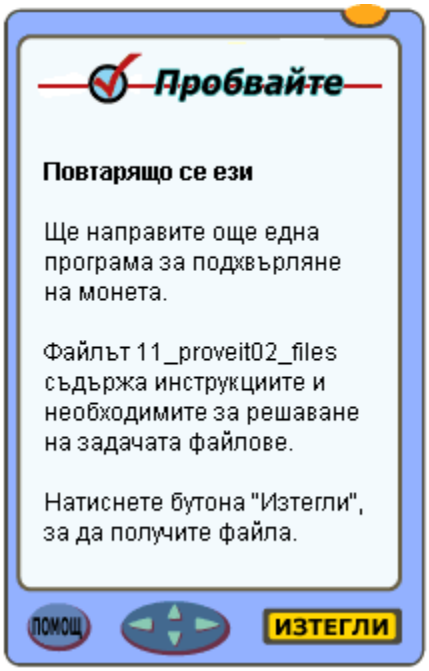
В C# няма еквивалент на цикъла Do...Until Loop във Visual Basic.

Пристигане в Пекин, Китай

Ето, че сме в Пекин! Това беше кратка, но интересна част от пътуването ни! Научихме как да използваме четири различни типа неопределени цикли. Можете да дефинирате цикли, които се изпълняват докато определено условие е истина или докато определено условие не стане истина. Можете да задавате условието за прекратяване на цикъла в края или в началото му. Когато условието се проверява в края на цикъла, блокът от код се изпълнява поне един път. Ще откриете, че различните типове цикли взаимно се допълват. Те могат да правят неща, които цикълът For...Next не може.

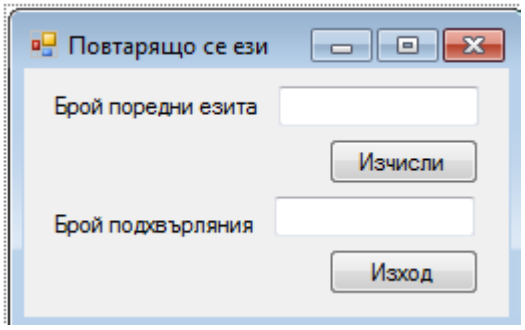
Да видим какво ви е приготвил старият джобен компютър!

11 Пробвайте 02 Повтарящо се ези



В една от предишните задачи се забавлявахме с подхвърлянето на монета. Нека да направим още една програма. Замисляли ли сте се някога колко хвърляния трябва да направите, за да получите последователно три пъти ези? А 4? А 5?

Формата, която трябва да създадете, изглежда по следния начин:



Кодът който генерира случайно число 0 или 1 е следният:

```
Dim MyRandomGenerator As System.Random
  MyRandomGenerator = New System.Random
  Dim RanNum As Integer

  ' Generate random value between 0 and 2 -
       not including 2.
  RanNum = MyRandomGenerator.Next(0, 2)
```

Приемете, че 1 съответства на ези, а 0 на тура. Съхранявайте броя на последователно генерираните единици в първото текстово поле. При генериране на 0 започвайте от начало. Визуализирайте броя на опитите във второто текстово поле.

При тестването прилежните програмисти поставят в цикъла оператор за условен преход if, с който показват съобщение и спират цикъла, ако броячът нарасне прекалено много.

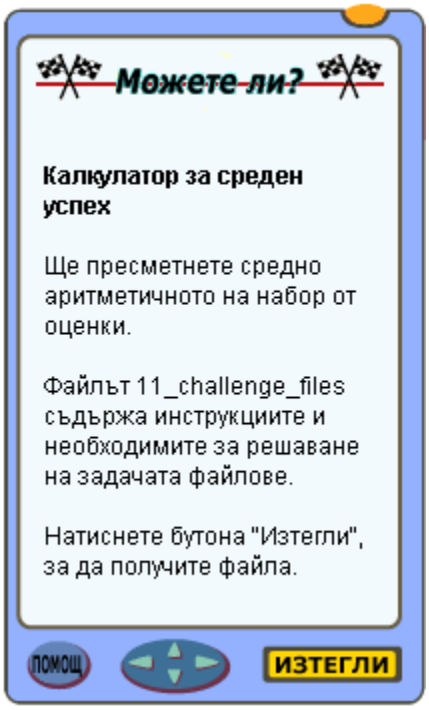
Ако програмата работи правилно, я покажете на учителя си.

Полезен съвет

Тази програма може би е малко полесна от предходната.

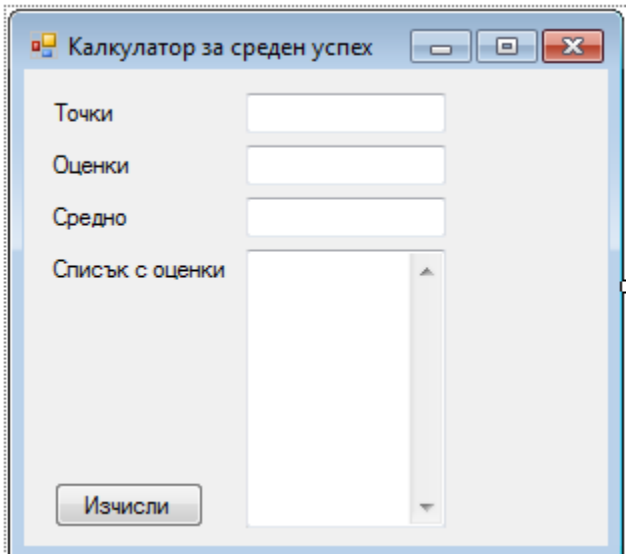
Вижте следващото ви предизвикателство, комплименти за Комисията по състезанията! Сигурен съм, че знанията ви за циклите ще ви бъдат полезни.

11 Можете ли? Калкулатор за среден успех

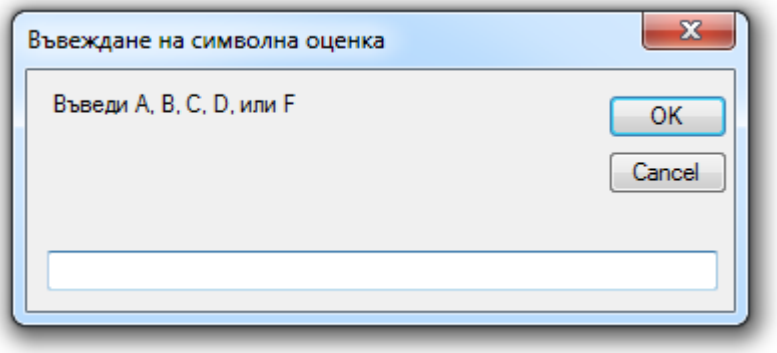


Забавлявам се по време на пътуването, а то е към своя край. След това ще трябва да се върна отново в училище, притеснявайки се за средния си успех. Една програма за изчисляване средно аритметичното на набор от оценки би ми била полезна. Нека да напишем такава програма.

Създайте форма подобна на показаната по-долу:



Програмата използва функцията InputBox, която създава форма подобна на показаната по-долу.



Вие не трябва да я създавате.Visual Basic ще направи това автоматично.

Тази функция е нова за вас така, че ще ви дадем целия код, от който се нуждаете.

Дефинирайте променлива, която да съхранява оценките по следния начин:

```
Dim GradeIn As String = "None Entered"
```

Създайте цикъл, в който ще използвате следния ред код:

```
GradeIn = InputBox("Въведи Enter A, B, C, D, или or
F", "Въведи символна оценка", "")
```

Потребителят ще въвежда толкова оценки, колкото са необходими. Когато приключи с въвеждането ще натисне бутона Cancel. В противен случай променливата GradeIn ще бъде празна. Празна стойност се обозначава с кавички.

A съответства на оценка 4
B съответства на оценка 3
C съответства на оценка 2,
D съответства на оценка 1
F съответства на 0.

Полезен съвет

Цикълът приключва, когато GradeIn = “”

Игнорирайте всички други символи и числа.Потребителят може да въвежда малки и големи букви.

Визуализирайте всичко въведено в текстово поле.

След като оценките са въведени пресметнете средния успех.

Продължение / Обобщение

Представете средния успех в символен вид.

Осигурете въвеждане на положителни и отрицателни оценки като например B+ и C-. Използвайте системата за оценяване във вашето училище или таблицата по-долу:

1.0	D -
1.3	D
1.7	D +
2.0	C -
2.3	C
2.7	C +
3.0	B -
3.3	B
3.5	B +
3.8	A -
4.0	A

Ето и специалния тест за проверка на знанията ви.