Vježba 3

Programiranje 1 (550) - ak. god. 2024./2025.

Strukture za ponavljanje

FOR petlja

FOR petlja je jedna od struktura za ponavljanje određenog dijela programskog koda koja se ponavlja određeni broj puta na temelju vrijednosti brojača:

VB

```
For <brojac> [As <tip>] = <start> To <kraj> Step <korak>
        ' Kod koji treba ponoviti vise puta
Next
```

Ova petlja koristi numeričku varijablu brojača kojoj se vrijednost povećava (ili smanjuje) svakim prolaskom kroz petlju. Početna i krajnja vrijednost određuju koje sve vrijednosti varijabla brojača može poprimiti, a opcionalna naredba Step vrijednost koraka određuje koliko se svakim prolaskom kroz petlju varijabla brojača povećava ili smanjuje: ako je korak pozitivan, prolaskom kroz petlju varijabla brojača se povećava, a ako je negativan se smanjuje. Ako korak nije naveden, podrazumijeva se vrijednost od 1 i nije ga potrebno eksplicitno navoditi. Primjer:

VB

```
For brojac As Integer = 3 To 15 Step 2
        Console.Write("{0} ", brojac)
Next
Console.WriteLine()
```

Ispis navedenog koda bit će: 3 5 7 9 11 13 15.

Petlju je moguće i prijevremeno prekinuti izvršavanjem naredbe `Exit For` unutar koda petlje. Za to se najčešće koristi IF naredba da izađemo iz petlje ukoliko je neki uvjet za to zadovoljen.

WHILE petlja

WHILE petlja ponavlja blok koda neodređeni broj puta dok je neki uvjet zadovoljen, tj. dok je njegova vrijednost `True`:

VB

```vb
While <uvjet>
        ' Naredbe za ponavljanje
End While
```

WHILE petlja provjerava uvjet prije svakog prolaska kroz petlju, pa tako nemamo garanciju da će se kod unutar petlje izvršiti barem jednom, ako je uvjet inicijalno `False`.

Info

Izvršavanje petlje ne mora nikad niti završiti, ako je uvjet uvijek `True`. To se naziva beskonačna petlja.

Primjer WHILE petlje u jednostavnom programu za računanje srednje vrijednosti unesenih brojeva dok korisnik ne unese nulu, što prekida unosi i ispisuje prosjek:

VB

```vb
Sub Main(args As String())
    Dim broj As Integer
    Dim suma As Integer
    Dim counter As Integer = 0
    Console.WriteLine("Unosite brojeve, te nulu za kraj")
    broj = Console.ReadLine()
    While broj <> 0
        suma += broj
        counter += 1
        broj = Console.ReadLine()
    End While
    Console.WriteLine("Srednja vrijednost: {0:F3}", suma / counter)
End Sub
```

Ukoliko je potrebno, WHILE petlju je moguće prijevremeno prekinuti naredbom `Exit While`.

DO-LOOP petlja

DO-LOOP je struktura za ponavljanje određenog dijela koda koja ima nekoliko varijacija, ali ono što je zajedničko svima je da se u nekom trenutku provjerava uvjet, slično kao kod WHILE petlje.

VB

```vb
Do While <uvjet>
        ' Kod za ponavljanje
```

```
3   Loop
```

VB

```
1   Do
2           ' Kod za ponavljanje
3   Loop While <uvjet>
```

Ovo su dvije petlje kod kojih je jedina razlika ta što će se kod ove druge kod unutar petlje garantirano izvršiti barem jednom, budući da će se uvjet evaluirati tek na kraju petlje. Prva od njih je u osnovi ista kao obična WHILE petlja, s malo drugačijom sintaksom.

Osim ove dvije, postoje i još dvije varijante:

VB

```
1   Do Until <uvjet>
2           ' Kod za ponavljanje
3   Loop
```

VB

```
1   Do
2           ' Kod za ponavljanje
3   Loop Until <uvjet>
```

One su u svojoj suštini iste kao i prethodne dvije, uz to da se uvjet evaluira na drugačiji način. Dakle, kod ove petlje se izvršava sve dok uvjet ne postane `True`, tj. dok je `False`. Kao i u prethodnom slučaju, druga petlja garantira izvođenje koda barem jednom.

Sve ove petlje je moguće prijevremeno prekinuti unutar koda petlje korištenjem naredbe `Exit Do`.

Rukovanje iznimkama

Iznimka (engl. *exception*) ukazuje na problem koji se može nastati za vrijeme izvođenja programa, a koji nije prisutan prilikom prevođenja. To se uobičajeno javlja kada se izvršava neka naredba ili niz naredbi ili operacija za koje ne možemo sa sigurnošću unaprijed reći da će se uspješno izvršiti. Neki od tipičnih scenarija u kojima se može javiti iznimka su:

- pokušaj dijeljenja s nulom
- dohvaćanje datoteka s interneta, što može biti neuspješno ako padne internet veza
- korisnik treba unijeti broj, a on unese nešto što nije broj ili unese broj koji je veći od maksimalnog za definirani tip podatka

VB.NET nam omogućava **strukturirano** rukovanje iznimkama, te se za to koristi `Try-Catch-Finally` programska struktura.

VB

```
Try
        ' naredbe
Catch
        ' naredbe
End Try

Try
        ' naredbe
Catch <varijablaIznimke> As <tipIznimke>
        ' naredbe
Finally
        ' naredbe
End Try
```

`Try` blok sadrži naredbe koje pratimo hoće li se dogoditi iznimka. Ako se iznimka dogodi, prekida se izvršavanje naredbi u `Try` bloku, te započinje izvršavanje naredbi u `Catch` bloku. Ako se sve naredbe u `Try` bloku izvrše uspješno, naredbe `Catch` bloka se **ne izvršavaju**. Ukoliko je definiran blok `Finally` i naredbe u njemu, one se izvršavaju uvijek nakon završetka `Try` bloka ako se nije dogodila iznimka, odnosno nakon završetka `Catch` bloka ako se dogodila iznimka.

S obzirom da `Catch` blok može hvatati iznimke različitih tipova, moguće je da ova programska struktura sadrži više od jednog Catch bloka, ali svaki mora biti s različitim tipom iznimke.

U sljedećem jednostavnom primjeru, imamo cijeli broj koji je postavljen na maksimalni iznos za svoj tip podataka. Kada taj broj pokušamo uvećati za 1, podignut će se iznimka zbog prekoračenja:

VB

```
Dim x As Short = Short.MaxValue
Try
        x += 1
Catch ex As OverflowException
        Console.WriteLine(ex.Message)
Finally
        Console.WriteLine(x)
End Try
```

Primijetite da u gornjem primjeru `Catch` blok očekuje iznimku tipa `OverflowException` (prekoračenje) i da ispisuje poruku koju ta iznimka sadrži. Ovako napisan `Catch` blok će

uhvatiti samo taj tip iznimke, dok ostale tipove neće. Zato ako želimo da budu uhvaćene sve iznimke, bez obzira na tip, navodi se samo `Catch` naredba bez varijable i tipa iznimke.

Neki od najčešće korištenih tipova iznimki su dani u sljedećoj tablici uz opise u kojim scenarijima se koriste.

Klasa iznimke	Opis
`IndexOutOfRangeException`	Iznimka koja se referira na pristup nepostojećem elementu niza.
`OverflowException`	Iznimka kod prekoračenja, kao rezultat matematičkih operacija kada se rezultat više ne može pohraniti u željeni tip jer je maksimalna vrijednost prekoračena.
`DivideByZeroException`	Iznimka kod dijeljenja s nulom
`InvalidCastException`	Iznimka kod nepravilne pretvorbe podataka između tipova
`OutOfMemoryException`	Iznimka zbog nedovoljne količine slobodne memorije
`StackOverflowException`	Iznimka zbog greške prekoračenja stoga (*stack overflow*), do koje najčešće dolazi zbog neograničene rekurzije ili beskonačnih petlji.

Zadaci

1. Učitati N cijelih brojeva (N unosi korisnik). Izračunati i ispisati njihovu aritmetičku sredinu.
2. Učitati N cijelih brojeva (N unosi korisnik). Ispisati najmanji i najveći od unesenih brojeva.
3. Učitavajte brojeve s konzole sve dok se upisuju neparni brojevi. Izračunajte aritmetičku sredinu, uzimajući samo brojeve veće od 10, a manje od 50 u izračun.
4. Učitavati cijele brojeve s konzole sve dok se upisuju pozitivni brojevi. Ispisati koliko je korisnik upisao prostih brojeva.
5. Korisnik unosi prirodni broj. Provjerite unos i ispišite odgovarajući poruku ako uneseni broj nije prirodni. Ispišite znamenke unesenog broja jednu ispod druge, počevši od one s najmanjom vrijednosti.
6. Korisnik unosi broj `N`, potrebno je napraviti ispis kao na predlošku.

Primjer za N=5

```
+++++
++++
+++
```

```
++
+
```

7. Ispišite sve brojeve između `M` i `N` čiji je zbroj znamenki djeljiv s brojem `B`. Sve brojeve unosi korisnik putem konzole.
8. Učitati N cijelih brojeva (N unosi korisnik). Izračunati i ispisati aritmetičku sredinu unesenih *parnih* brojeva osim onih koji završavaju s 4.
9. Prebrojite koliko brojeva unutar zadanog raspona (koji unosi korisnik putem konzole) ima znamenku jedinice vrijednosti 9.
10. Učitati 10 troznamenkastih brojeva i za svaki od učitanih brojeva ispisati aritmetičku sredinu njegovih znamenki.
11. Učitavati brojeve dok se ne unese 0. Ispisati zbroj učitanih brojeva, te one koji su djeljivi s 5 ili sa 7.
12. Učitavati troznamenkaste brojeve dok je zbroj znamenki paran. Ispisati koliko je brojeva učitano.
13. Napišite jednostavni kalkulator. Korisnik unosi dva broja (Integer), te oznaku za operaciju +,-* ili /. Potrebno je uhvatiti iznimke ako dođe do dijeljenja s nulom ili do prekoračenja dozvoljenih vrijednosti za tip Integer. Na kraju je potrebno ispisati rezultat. Na internetu pronađite koje iznimke se koriste u navedenim situacijama te ih iskoristite u rješenju.