



# Programiranje kroz aplikacije

Kontrola toka programa

Nizovi

# Kontrola toka u VBA

- **If** naredba
- **Select Case** naredba
- **GoTo** naredba
- **For** petlja
- **While** petlja
- **Do While** petlja
- **Do Until** petlja

# If naredba

- Opšti oblik naredbe:

If Uslov1 Then

    Instrukcije1

Elsef Uslov2 Then

    Instrukcije2

...

Elsef UslovN Then

    InstrukcijeN

Else

    PodrazumevaneInstrukcije

End If

- Instrukcije1, ..., InstrukcijeN se uvek pišu u zasebnom redu, ispod odgovarajućeg uslova, tj. **ne smeju biti u istom redu sa uslovima.**
- Else grana je opcionalna, tj. ne mora se navoditi ako nema potrebe.
- End If završava If naredbu.

# Primer sa If naredbom

- Napisati funkciju **Ocena** koja za argument ima broj poena koje je student dobio na ispitu i vraća ocenu koju je student dobio znajući da važi relacija:

Ocena =  $\left\{ \begin{array}{l} \text{A, BrojPoena} \geq 90 \\ \text{B, BrojPoena} \geq 80 \\ \text{C, BrojPoena} \geq 70 \\ \text{D, BrojPoena} \geq 60 \\ \text{E, BrojPoena} \geq 50 \\ \text{F, BrojPoena} < 50. \end{array} \right.$

```
Function Ocena(BrojPoena As Integer) As String
  If BrojPoena >= 90 Then
    Ocena = "A"
  ElseIf BrojPoena >= 80 Then
    Ocena = "B"
  ElseIf BrojPoena >= 70 Then
    Ocena = "C"
  ElseIf BrojPoena >= 60 Then
    Ocena = "D"
  ElseIf BrojPoena >= 50 Then
    Ocena = "E"
  Else
    Ocena = "F"
  End If
End Function
```

# Jednolinijska If naredba

- U slučaju kada postoji samo jedan uslov i mali broj instrukcija, pogodnije je koristiti jednolinijsku **If** naredbu, čiji je oblik:

**If Uslov Then Instrukcije1 Else Instrukcije2**

- U jednolinijskom obliku, **End If** se ne navodi!
- I ovde je **Else** deo opcion, tj. može se pisati:

**If Uslov Then Instrukcije**

## Primeri:

```
If (X Mod 2 = 0) Then MsgBox "Broj X je paran"
```

```
If (X Mod 2 = 0) Then MsgBox "Broj X je paran" Else _  
    MsgBox "Broj X je neparan"
```

# Select Case naredba

- Opšti oblik naredbe:

```
Select Case Testnilzraz
  Case Izraz1
    Instrukcije1
  ...
  Case IzrazN
    InstrukcijeN
  Case Else
    PodrazumevaneInstrukcije
End Select
```

- **Testnilzraz** određuje koja će se **Case** grana izvršiti.
- **Case Else** grana je opcionalna.
- **End Select** završava **Select Case** naredbu.

# Primer sa Select Case naredbom

- Napišimo funkciju `Ocena` koristeći `Select Case` naredbu.

```
Function Ocena(BrojPoena As Integer) As String
  Select Case BrojPoena
    Case Is >= 90
      Ocena = "A"
    Case Is >= 80
      Ocena = "B"
    Case Is >= 70
      Ocena = "C"
    Case Is >= 60
      Ocena = "D"
    Case Is >= 50
      Ocena = "E"
    Case Else
      Ocena = "F"
  End Select
End Function
```

Upotreba ključne reči `Is` radi specificiranja vrednosti!!!

```
Case 90 To 100
  Ocena = "A"
Case 80 To 90
  Ocena = "B"
...
```

Upotreba ključne reči `To` radi specificiranja opsega vrednosti!!!

# Case uslovi

- U **Case** uslovu, više vrednosti se razdvaja zarezima:

**Case** 10, 20, 50

- Dozvoljene su i kombinacije vrednosti sa ključnim rečima **Is** i **To**:

**Case** 13, 20 **To** 30, 50 **To** 60, **Is** > 90

- U **Case** uslovu, mogu se koristiti i stringovi. Na primer:

**Case** "aaa" **To** "bbb"

odgovara svim stringovima između "aaa" i "bbb", a to su "aaaa", "abcd", "babin zub" itd.



# GoTo naredba

- **GoTo** je naredba bezuslovnog skoka (grananja), tj. **GoTo** bezuslovno prebacuje izvršavanje programa na određenu instrukciju procedure koja mora započeti **labelom**.
- Labela je oznaka koja jednoznačno određuje instrukciju na koju se ide naredbom **GoTo**. Labela može biti **tekstualni string praćen dvotačkom (:)** ili **broj bez dvotačke**.

If Uslov Then **GoTo Lab** Else **GoTo 1000**

Naredbe

**Lab:** Naredbe

Naredbe

**1000** Naredbe

- **GoTo** ne može da grana van procedure!
- U VBA proceduri može postojati proizvoljan broj različitih labela.
- **GoTo** ćemo uglavnom koristiti kod upravljanja greškama.

# For petlja

- Brojačka **For** petlja ima oblik:

```
For Brojac = Pocetak To Kraj Step Korak  
  Naredbe  
Next Brojac
```

- **Korak** može biti pozitivan i negativan, a može se i izostaviti. Podrazumevano je 1.
- **Next Brojac** se skraćeno može zapisati sa **Next**.
- Forsirani izlazak iz **For** petlje se vrši sa **Exit For**, čime se iz petlje bezuslovno izlazi i prelazi na izvršenje prve naredbe nakon petlje.
- Šta rade sledeće dve petlje?

```
S = 0  
For I = 1 To 99  
  S = S + I  
Next I
```

```
S = 0  
For I = 1 To 99 Step 2  
  S = S + I  
Next
```

# Do While petlja

- **Do While** je programska konstrukcija kod koje se naredbe (telo petlje) izvršavaju sve dok je ispunjen uslov petlje. Dolazi u dva oblika:

**Do While Uslov**  
Naredbe  
Loop

**Do**  
Naredbe  
Loop While **Uslov**

- U prvom obliku se može desiti da se u petlju uopšte ne uđe. U drugom obliku se petlja izvršava minimum jedanput.
- Forsirani izlazak iz **Do While** petlje se vrši naredbom **Exit Do**.
- Sabrati prirodne brojeve manje od 100 pomoću obe **Do While** petlje.

```
S = 0 : K = 1
Do While K < 100
  S = S + K
  K = K + 1
Loop
```

```
S = 0 : K = 0
Do
  K = K + 1
  S = S + K
Loop While K < 99
```

# Do Until petlja

- **Do Until** petlja se izvršava sve dok se ne ispuni određeni uslov. Uslov se može naći na početku ili kraju petlje:

**Do Until Uslov**  
Naredbe  
Loop

**Do**  
Naredbe  
Loop Until **Uslov**

- U prvom obliku se može desiti da se u petlju uopšte ne uđe. U drugom obliku se petlja izvršava minimum jedanput.
- Forsirani izlazak iz **Do Until** petlje se vrši naredbom **Exit Do**.
- Sabrati prirodne brojeve manje od 100 pomoću obe **Do Until** petlje.

```
S = 0 : K = 1
Do Until K = 100
  S = S + K
  K = K + 1
Loop
```

```
S = 0 : K = 0
Do
  K = K + 1
  S = S + K
Loop Until K = 99
```

# Do petlja

- Kod **Do While** i **Do Until** petlji, **While** i **Until** delovi su opcioni.
- **Do While** i **Do Until** petlje su specijalni slučajevi **Do** petlje.

Do

    Naredbe

Loop

- Iz **Do** petlje se najčešće izlazi sa **Exit Do**.
- Sabrati prirodne brojeve manje od 100 koristeći **Do** petlju.

```
S = 0 : K = 1
Do
  S = S + K
  K = K + 1
  If K = 100 Then Exit Do
Loop
```

# Nizovi

- Niz predstavlja grupu elemenata koji imaju isti tip i ime.
- Elementu niza se pristupa koristeći ime niza i indeks (redni broj elementa) koji se navodi u malim zagradama.
- Indeks prvog elementa niza je podrazumevano 0. Tako bi elementima niza  $X$ , dužine 10, pristupali na sledeći način:

$X(0), X(1), \dots, X(9)$

- Navođenjem

**Option Base 1**

na početku modula, indeks prvog elementa niza će biti 1 u svim procedurama tog modula. **Option Base** može biti 0 ili 1.

# Deklaracija nizova

- Naredba

`Dim X(10) As Integer`

deklariše celobrojni niz `X` od 11 elemenata, jer je prvi element `X(0)`, a poslednji `X(10)`.

- U deklaraciji se može navesti indeks prvog i poslednjeg elementa niza, pri čemu su dozvoljeni i negativni indeksi:

`Dim X(1 To 10) As Integer`

`Dim X(-5 To 5) As Integer`

- Indeks prvog elementa niza se može dobiti pomoću funkcije `LBound(ImeNiza)`, dok se indeks poslednjeg elementa može dobiti pomoću funkcije `UBound(ImeNiza)`.

# Primer sa nizovima

- Napisati proceduru koja formira niz od 20 Fibonacci-evih brojeva i elemente tog niza štampa u Immediate prozoru. Prva dva broja su 0 i 1, a svaki sledeći Fibonacci-ev broj jednak je zbiru prethodna dva.

```
Sub Fibonacci()  
  Dim X(1 To 20) As Integer, I As Integer  
  X(1) = 0 : X(2) = 1  
  Debug.Print X(1)  
  Debug.Print X(2)  
  For I = 3 To 20  
    X(I) = X(I - 1) + X(I - 2)  
    Debug.Print X(I)  
  Next  
End Sub
```

Korišćena **Sub** procedura.  
Poziva se sa **Fibonacci**.

Za štampu u Immediate prozoru tokom izvršavanja koristi se metoda **Debug.Print**.



# Višedimenzioni nizovi

- VBA podržava do 60 dimenzija nizova.
- Pri deklaraciji se može navesti samo gornja granica (donja je podrazumevano 0), ili se mogu navesti obe granice.

```
Dim X(10, 10) As Integer
```

```
Dim X(1 To 10, -5 To 5) As Integer
```

- Elementu višedimenzionog niza se pristupa navođenjem indeksa svake dimenzije

```
X(2, 5) = 14
```

- Granice pojedinih indeksa se dobijaju pomoću funkcija **LBound** i **UBound**, pri čemu se kao drugi argument navodi redni broj dimenzije. Na primer, indeks poslednje kolone dvodimenzionog niza **X** se dobija sa

```
UBound(X, 2)
```

# Dinamički nizovi

- VBA podržava i dinamičko alociranje nizova.
- Pri deklaraciji dinamičkog niza, ne navodi se indeks poslednjeg elementa:

`Dim X() As Integer`

- Pre prve upotrebe niza u programu, mora se definisati broj elemenata, što se radi naredbom `ReDim` na sledeći način:

`ReDim X(10)`

`ReDim X(1 To 10)`

- Pomoću `ReDim` se može vršiti i promena broja elemenata niza (redimenzionisanje) i to proizvoljan broj puta.
- Prilikom redimenzionisanja se gubi vrednost elemenata. Za očuvanje vrednosti elemenata niza koristiti naredbu `Preserve`, na primer:

`ReDim Preserve X(10)`

# Primer sa dinamičkim nizovima

- Napisati proceduru koja za ulazni argument ima ceo broj **N**, formira niz od N Fibonacci-evih brojeva i elemente tog niza štampa u Immediate prozoru. Ukoliko je  $N \leq 0$  izaći iz procedure.

```
Sub Fibonacci(N As Integer)
  Dim X() As Integer, I As Integer
  If N <= 0 Then Exit Sub
  ReDim X(1 to N)
  X(1) = 0
  If N >= 2 Then
    X(2) = 1
    For I = 3 To N
      X(I) = X(I - 1) + X(I - 2)
    Next
  End If
  For I = 1 To N
    Debug.Print X(I)
  Next
End Sub
```

Izlazak iz procedure  
sa **Exit Sub**.

Procedura se poziva sa:  
**Fibonacci 12** ili  
**Call Fibonacci(12)**

# Niz kao argument funkcije

- Niz može biti argument funkcije.
- Niz kao argument se navodi sa  
`Niz() As Tip`
- Granice indeksa se ne moraju prosleđivati kao argumenti jer možemo koristiti funkcije `LBound` i `UBound`.
- Napisati funkciju koja za argument ima celobrojni niz `X` i vraća maksimum tog niza.

```
Function MaksNiza(X() As Integer) As Integer
    Dim I As Integer
    MaksNiza = X(LBound(X))
    For I = LBound(X) + 1 To UBound(X)
        If MaksNiza < X(I) Then MaksNiza = X(I)
    Next
End Function
```

Funkcija se poziva kao:  
`M = MaksNiza(X)`

# Primer 1

- Napisati funkciju **Maks** koja za argumente ima tri cela broja **A**, **B** i **C**, i vraća najveći od njih.

```
Function Maks(A As Integer, B As Integer, C As Integer) As Integer
    Maks = A
    If Maks < B Then Maks = B
    If Maks < C Then Maks = C
End Function
```

# Primer 2

- Napisati funkciju **BrCif** koja ima jedan celobrojni argument **N**, a vraća broj cifara tog broja. Prilagoditi funkciju tako da radi i sa pozitivnim i negativnim argumentom. Uzeti da je maksimalan broj cifara 5.

```
Function BrCif(N As Integer) As Integer
  If N < 0 Then N = -N
  If N < 10 Then
    BrCif = 1
  ElseIf N < 100 Then
    BrCif = 2
  ElseIf N < 1000 Then
    BrCif = 3
  ElseIf N < 10000 Then
    BrCif = 4
  Else
    BrCif = 5
  End If
End Function
```

**Vežba:** Uraditi primer koristeći **Select Case** naredbu.

# Primer 3

- Napisati funkciju **ZbirCifara** koja određuje i vraća zbir cifara celog broja **N**, koji je argument funkcije. Prilagoditi funkciju da radi i u slučaju negativnog broja **N**.

```
Function ZbirCifara(N As Integer) As Integer
    Dim Cifra As Integer
    If N < 0 Then N = -N
    ZbirCifara = 0
    Do While N <> 0
        Cifra = N Mod 10
        ZbirCifara = ZbirCifara + Cifra
        N = N \ 10
    Loop
End Function
```

Dobijanje poslednje cifre (najmanja težina).

Uklanjanje poslednje cifre.

# Primer 4

- Napisati funkciju **JeLiSortiran** koja za argument ima niz **X** i vraća **True** ako je niz sortiran u rastući poredak i **False** u suprotnom. Napisati i sub-procedure koja testira funkciju.

```
Function JeLiSortiran(X() As Integer) As Boolean
    Dim I as Integer
    JeLiSortiran = False
    For I = LBound(X) To UBound(X) - 1
        If X(I) >= X(I + 1) Then Exit Function
    Next
    JeLiSortiran = True
End Function
```

## Testiranje funkcije

```
Sub Test()
    Dim X(1 To 5) As Integer
    Dim sortiran As Boolean
    X(1) = 2
    X(2) = 3
    X(3) = 16
    X(4) = 11
    X(5) = 23
    sortiran = JeLiSortiran(X)
    If sortiran = True Then
        MsgBox "Jeste sortiran"
    Else
        MsgBox "Nije sortiran"
    End If
End Sub
```