

Excel VBA alapjai

Az Excel VBA (Visual Basic for Applications) egy eseményvezérelt programozási nyelv, amely lehetővé teszi az Excel alkalmazások automatizálását és testreszabását. Ebben a részben az Excel VBA alapvető nyelvi elemeit mutatjuk be, beleértve a változók deklarálását, az alapvető típusokat és a vezérlési elemeket.

Változók Deklarálása

A VBA-ban a változók deklarálása a `Dim` kulcsszóval történik. A változó típusát is megadhatjuk a deklaráció során, ami segít a kód olvashatóságában és a típushibák elkerülésében.

- **Példa:** Egyszerű változó deklaráció

```
Dim szam As Integer  
Dim szoveg As String
```

Alapvető Típusok

A VBA számos alapvető adattípust támogat, többek között:

- **Integer:** Egész számok
- **Long:** Nagyobb egész számok
- **Single:** Lebegőpontos számok (egyszeres pontossággal)
- **Double:** Lebegőpontos számok (kétszeres pontossággal)
- **String:** Szöveges típus
- **Boolean:** Logikai típus (Igaz vagy Hamis)

Vezérlési Elemek

A VBA vezérlési szerkezetei lehetővé teszik a program ágának irányítását a különböző feltételek alapján.

- **If...Then...Else**

```
If szam > 10 Then  
    MsgBox "A szám nagyobb, mint 10."  
Else  
    MsgBox "A szám 10 vagy annál kisebb."  
End If
```

- **For Next Ciklus**

```
For i = 1 To 5  
    MsgBox "Szám: " & i
```

Next i

Itt fontos megjegyezni hogy a szövegeket & jellel adjuk össze, nem a plusz operátorral!

- **Do While Loop**

```
i = 1
Do While i <= 5
    MsgBox "Szám: " & i
    i = i + 1
Loop
```

Sztringek kezelése

A VBA-ban a stringek kezelésének alapjai közé tartozik a változók deklarálása, értékadás, és az összehasonlítás.

```
// Deklarálás
Dim exampleString As String
// Értékadás
exampleString = "Hello World"
```

Műveletek Sztringekkel

Összefűzés:

A & operátort használva fűzhetünk össze sztringeket.

```
Dim firstName As String
Dim lastName As String
Dim fullName As String
firstName = "John"
lastName = "Doe"
fullName = firstName & " " & lastName // "John Doe"
```

A **Len** függvény visszaadja a string hosszát.

```
Dim textLength As Integer
textLength = Len("Hello World") // 11
```

Részsstring Kinyerése

A **Mid** függvényt használva kinyerhetünk egy részsstringet egy adott pozíciótól kezdődően, adott hosszúságban.

```
Dim subString As String  
subString = Mid("Hello World", 7, 5) // "World"
```

Keresés a Sztringben

A **InStr** függvény visszaadja egy substring első előfordulásának pozícióját egy másik stringben.

```
Dim position As Integer  
position = InStr("Hello World", "World") // 7
```

Sztring Nagy- és Kisbetűssé Alakítás

A **UCase** és **LCase** függvényeket használva nagybetűssé, illetve kisbetűssé alakíthatunk egy sztringet.

```
Dim upperCaseString As String  
Dim lowerCaseString As String  
upperCaseString = UCase("Hello World") // "HELLO WORLD"  
lowerCaseString = LCase("Hello World") // "hello world"
```

Sztring Összehasonlítás

A **StrComp** függvényt használva összehasonlíthatunk két stringet, figyelembe véve a nagy- és kisbetűk különbségét is.

```
Dim result As Integer  
// 0, ha a stringek egyenlők  
result = StrComp("Hello", "hello", vbBinaryCompare) // Nem egyenlő  
result = StrComp("Hello", "Hello", vbBinaryCompare) // Egyenlő
```

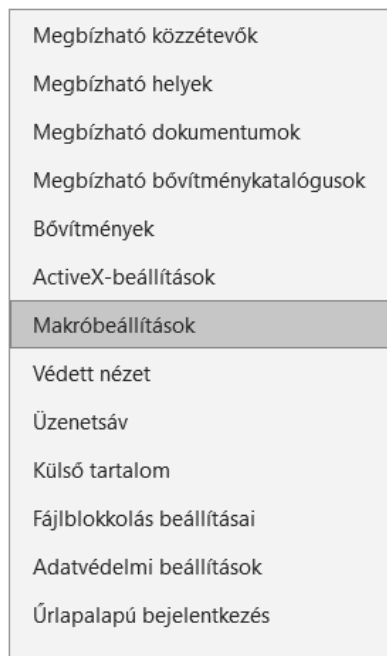
Összefoglalás

Ez a fejezet bevezetést nyújtott az Excel VBA alapjaihoz, beleértve a változók deklarálását, az alapvető típusokat és a vezérlési elemeket. A bemutatott kód példákkal könnyebbé válik az alapvető programozási koncepciók megértése és alkalmazása.

Makrók engedélyezése

A Fájl/Beállítások/Adatvédelmi központ-nál állítsuk be a makrók engedélyezését és utána zárjuk be a munkafüzetet és indítsuk újra.

Adatvédelmi központ



Makróbeállítások

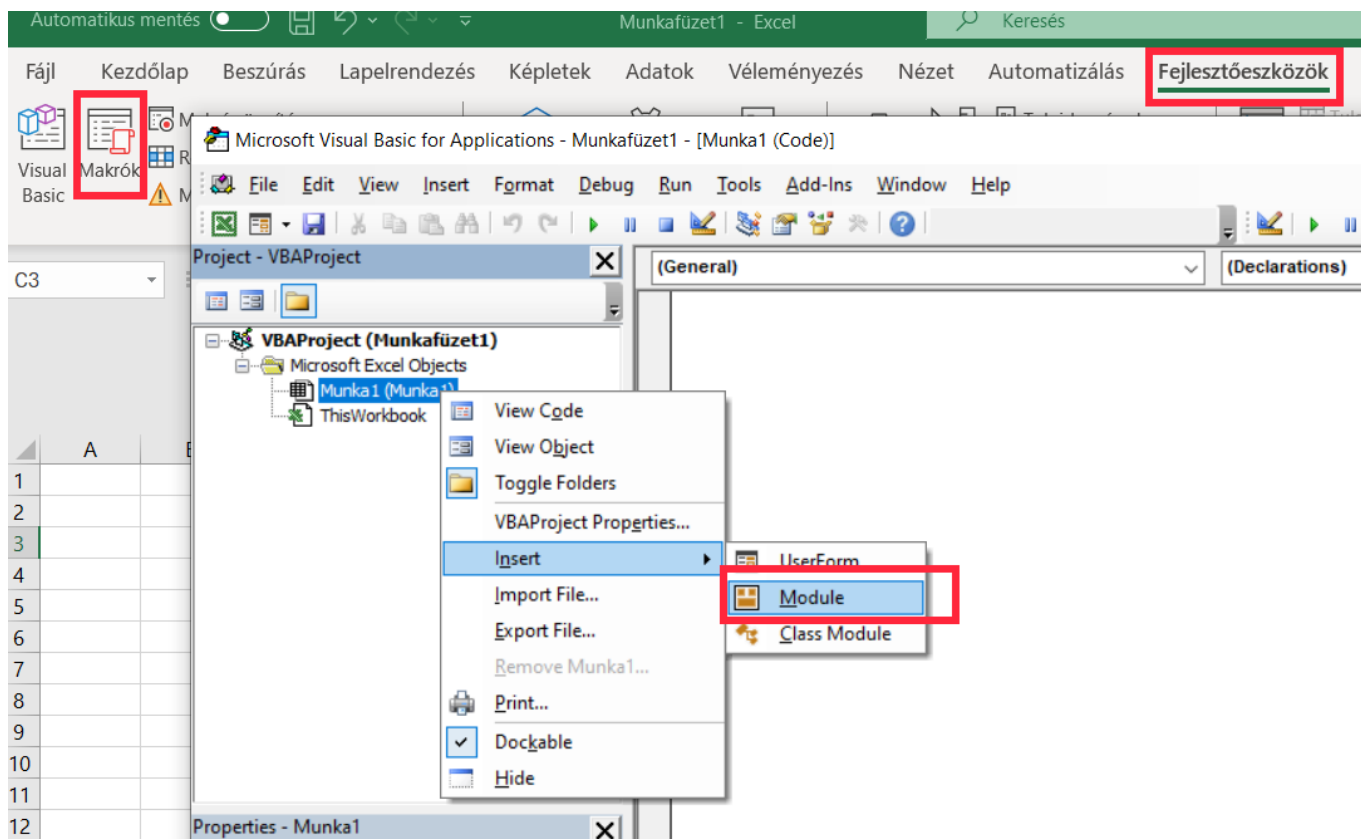
- ☐ A VBA-makrók letiltása értesítés nélkül
- ☐ A VBA-makrók letiltása értesítéssel
- ☐ A VBA-makrók letiltása a digitálisan aláírt makrók kivételével
- ☒ VBA-makrók engedélyezése (nem javasolt, mert veszélyes kód futtatását is lehetővé teszi)

☒ Excel 4.0-makrók engedélyezése, ha a VBA-makrók engedélyezve vannak

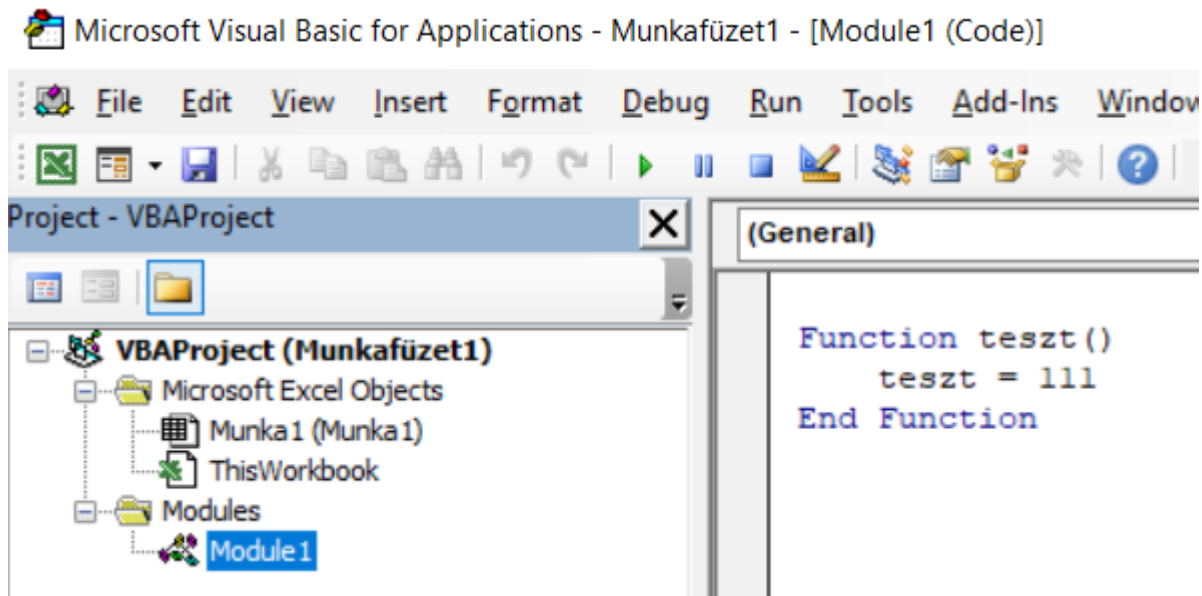
Fejlesztői makróbeállítások

☒ A VBA-projekt objektummodelljéhez való hozzáférés megbízható

Válasszuk ki a "Fejlesztőeszközök" fület, majd a "Makrók" gomb megnyomása után felugró ablakban a jobb egérgombbal megjelenő menüben illesszünk be egy új modult:



Definiáljunk egy teszt függvényt ami visszaad egy konstans értéket.



```
Function teszt()  
    teszt = 1  
End Function
```

használata: írjuk be egy cellába **=teszt()**

Adjunk össze két számot.

```
Function Osszeadas(szam1 As Double, szam2 As Double) As Double  
    Osszeadas = szam1 + szam2  
End Function
```

használata: **=Osszeadas(A1;B1)**

Írjunk egy függvényt, amely egy sztringet vesz bemenetként, és visszaadja annak fordított változatát.

```
Function SzovegForditas(szoveg As String) As String  
    Dim i As Integer  
    For i = Len(szoveg) To 1 Step -1  
        SzovegForditas = SzovegForditas & Mid(szoveg, i, 1)  
    Next i  
End Function
```

Ez a függvény egy tartomány elemeinek átlagát számítja ki.

```
Function TartomanyAtlag(tartomany As Range) As Double
    Dim osszeg As Double
    Dim db As Long
    osszeg = 0
    db = 0
    For Each cella In tartomany
        osszeg = osszeg + cella.Value
        db = db + 1
    Next cella
    TartomanyAtlag = osszeg / db
End Function
```

	A	B
1	2	3
2	3	10

Töltsük fel tetszőleges értékekkel a A1 és B2 téglalapba eső cellákat:

Egy másik cellába írjukbe: **=TartomanyAtlag(A1:B2)**

Viszont, ha az értékek között van olyan ami nem szám, akkor nem működik, ezért alakítsuk át, hogy kezelje a kivételeket is.

```
Function TartomanyAtlag(tartomany As Range) As Double
    Dim cella As Range
    Dim osszeg As Double
    Dim db As Long
    osszeg = 0
    db = 0
    For Each cella In tartomany
        If IsNumeric(cella.Value) Then
            osszeg = osszeg + cella.Value
            db = db + 1
        End If
    Next cella
    If db > 0 Then
        TartomanyAtlag = osszeg / db
    Else
        TartomanyAtlag = 0
    End If
End Function
```

Töltsük fel tetszőleges értékekkel a A1 és B2 téglalapba eső cellákat, de az egyiket akarattal elrontjuk:

	A	B
1	2	3
2	hello	1

Egy másik cellába írjuk be: **=TartomanyAtlag(A1:B2)**

Szorozzunk össze egy mátrix-ot egy vektorral:

```
Function MatrixVektorSzorzas(matrix As Range, vektor As Range) As Variant
    Dim sorok As Long, oszlopok As Long, i As Long, j As Long
    Dim eredmeny() As Double
    sorok = matrix.Rows.Count
    oszlopok = matrix.Columns.Count
    ' Ellenőrizzük, hogy a vektor mérete megfelel-e a mátrix oszlopainak
    számával
    If vektor.Rows.Count <> oszlopok Or vektor.Columns.Count > 1 Then
        MatrixVektorSzorzas = CVErr(xlErrValue)
        Exit Function
    End If
    ' oszlopvektor legyen az eredmény
    ReDim eredmeny(1 To sorok, 1 To 1) As Double
    For i = 1 To sorok
        For j = 1 To oszlopok
            eredmeny(i, 1) = eredmeny(i, 1) + matrix.Cells(i, j).Value *
vektor.Cells(j, 1).Value
        Next j
    Next i
    MatrixVektorSzorzas = eredmeny
End Function
```

Használata például ez lehet:

G1

⋮

✖

✓

f_x

=MatrixVektorSzorzas(A1:C3;E1:E3)

	A	B	C	D	E	F	G
1	1	2	3		1		8
2	4	5	6	x	2	=	20
3	7	8	9		1		32
4							

From:

<https://edu.iit.uni-miskolc.hu/> - Institute of Information Science - University of Miskolc

Permanent link:

https://edu.iit.uni-miskolc.hu/tanszek:oktatas:muszaki_informatika:vba_tutorial?rev=1712693718

Last update: **2024/04/09 20:15**

