Visual Basic és Excel makrók

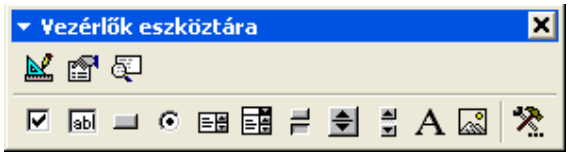
1. Bevezetés

Az Excel táblázatkezelővel sok probléma megoldható a beépített függvények és eljárások segítségével. Vannak más problémák, amelyekre az Excelnek nincsenek közvetlenül függvényei ill. eljárásai, de rendelkezik olyan lehetőségekkel, amelyeket felhasználva a megoldást magunk készíthetjük el. Az Excel Visual Basic for Applications (VBA) háttere használható fel e célokra. A továbbiakban bepillantást kapunk a VBA programozásba.

2. Űrlapok készítése Excelben

Elmélet

A „Nézet” menüben az „Eszköztárak” közül kapcsoljuk be az „Vezérlők” eszköztárat! Az eszköztáron megtaláljuk a Windows párbeszédpanelekről már ismert vezérlőket, melyeket elhelyezhetünk a munkalapon, valamint néhány ikont, ami a vezérlők szerkesztésében segít.



(Az űrlap eszköztáron található vezérlők ugyanazon funkciókat valósítják meg, mint a vezérlő eszköztáron lévők, csak a felhasználónak sokkal kevesebb beállítási lehetősége van. Az űrlap vezérlői például programkódból nem elérhetőek. Ezért ezt az eszköztárat nem fogjuk használni.)

A felhelyezhető vezérlők: *(A vezérlőknek sok neve van a köztudatban, mi magyarul és angolul is az Excel által használt neveket tüntetjük fel.)*

- Jelölőnégyzet (CheckBox): Egy kis négyzet felirattal. Kijelölt és nem kijelölt állapota lehet. Kijelölt állapotban a négyzetben egy pipa van. A vezérlőkről egyesével dönthető el, hogy ki legyenek jelölve, vagy sem, vagyis egyidejűleg több is kiválasztható.
- Beviteli mező (TextBox): Olyan mező, amibe a tervező mód kikapcsolása után a felhasználó írni tud.
- Parancsgomb (CommandButton): Olyan vezérlő, amelyet virtuálisan meg lehet nyomni. Hatására egy eljárás indul el. Csak a későbbiekben fogjuk használni.
- Választókapcsoló (OptionButton): Egy kis karika felirattal. Kijelölt és nem kijelölt állapota lehet. Kijelölt állapotot a karikában egy pont jelzi. A vezérlők együtt működnek. Több választókapcsolóból egyszerre csak egy lehet kijelölve. Ha másikat választunk ki, az előző kijelölésünk megszűnik.
- Listapanel (ListBox): Olyan mező, amiben egy felsorolás található, amiből a tervező nézet kikapcsolása után a felhasználó választhat.
- Kombi panel (ComboBox): Beszélőbb nevén legördülő lista. Mint a listánál, itt is több lehetőség közül választhatunk. Alaphelyzetben egy üres mező látszik mellette egy lefelé

mutató nyíllal. A nyílra kattintva láthatjuk a lista elemeit és választhatunk közülük. A választás után a lista eltűnik és csak a kiválasztott elemet látjuk. A listapanellel ellentétben itt a felhasználó maga is beírhat a mezőbe, ezáltal olyan értéket választva, amit a lista nem ajánlott fel.

-  Váltógomb (ToggleButton): A váltógomb a jelölőnégyzethez hasonlóan viselkedik, csak a kijelölt állapotot nem egy négyzetben lévő pipa mutatja, hanem az, hogy a gomb be van nyomva, vagy nincs.

-  Léptetőgomb (SpinButton): Két egymástól elfelé mutató nyíl. Segítségükkel számlálni tudunk a megadott határok között a megadott lépésközzel.

-  Görgetősáv (ScrollBar): A léptetőgomb rokona, azzal a különbséggel, hogy a két nyíl között egy sáv is található, aminek a segítségével nagyobb lépéseket is tehetünk. A sávon csúszka is van, amit a felhasználó a két szélső érték között szabadon tud csúsztatni.

-  Felirat (Label): E vezérlő segítségével helyezhetünk el feliratokat a munkalapon, amelyeket a tervező mód kikapcsolása után a felhasználó nem tud változtatni.

-  Kép (Picture): E vezérlő segítségével tudunk képet felhelyezni a munkalapunkra.

A vezérlők eszköztáron található, a szerkesztést és programozást segítő ikonok:

 ikon segítségével tudjuk ki-be kapcsolni, hogy az adott vezérlőt szerkeszteni, vagy használni szeretnénk. (Tervező mód - Kilépés a tervezésből)

 ikon: Miután egy vezérlőt elhelyeztünk a munkalapon ezzel az ikonnal tudjuk a tulajdonságait beállítani. A tulajdonságok ablak nincs magyarra fordítva. (Tulajdonságok)

 ikon: Az adott vezérlőhöz kapcsolódó programkód megtekintése. Csak a későbbiekben fogjuk használni. (Kód megjelenítése)

Néhány hasznos tulajdonság, melyeket a Properties ablakban abc sorrendben (Alphabetic), vagy témák szerint (Categorized) láthatunk és állíthatunk be:

- (Name): A vezérlő neve, ami áll a vezérlő típusnevéből, például Label, és egy sorszámból. A vezérlők átnevezhetőek, de ez kezdő felhasználók esetén nem javasolt.
- LinkedCell: Ez a tulajdonság kitűntetett jelentőséggel bír. A vezérlőt ennek segítségével tudjuk cellához kapcsolni. Ha a vezérlő értéke módosul, akkor a kapcsolt cella értéke is illetve fordítva. A vezérlők értéke mindig szövegként kerül a cellába, még akkor is, ha számot tartalmaz. Ha képletekben szeretnénk az adott cellára, mint számra hivatkozni, akkor az ÉRTÉK függvénnyel számmá kell konvertálnunk. A kapcsolt cellát az Excelben megszokott módon, tehát az oszlop betűjelével és a sor számával tudjuk megadni.
- Caption: A felirattal rendelkező vezérlők feliratát adhatjuk meg itt.
- Text: A beviteli mezőnek nem felirata van, hanem szövege. Ezt a Text tulajdonság megadásával tudjuk beállítani.
- Font: A felirattal, vagy szöveggel rendelkező vezérlők feliratának betűtípusát adhatjuk meg.
- ForeColor: A felirattal, vagy szöveggel rendelkező vezérlők feliratának színét adhatjuk meg.
- BackColor: Háttérszínt adhatunk a vezérlőnek.

- Enabled: A vezérlő elérhetőségét lehet állítani. Két értékű tulajdonság: ha az értéke True, akkor a vezérlő használható (például a gomb megnyomható, vagy a beviteli mezőbe lehet írni), ha az értéke False, akkor a vezérlő nem használható.
- Visible: A vezérlő láthatóságát lehet vele állítani. Szintén két értékű tulajdonság: ha az értéke True, akkor a vezérlő látható, ha False, akkor nem látható. Természetesen amíg tervező módban vagyunk, addig minden látható, de ha a fent bemutatott ikonnal kilépünk a tervezésből, akkor a láthatatlanra állított vezérlők eltűnnek.
- Value: A Jelölőnégyzetnek, a Választókapcsolónak és a Váltógombnak a tulajdonsága. Két értéke van: ha a vezérlő ki van választva, akkor True, ha nincs kiválasztva akkor False.
- Picture: A Kép vezérlő tulajdonsága. Itt tudjuk megadni a beillesztendő kép elérési útját.
- ListFillRange: Itt adhatjuk meg azt a cellatartományt, amely a kiválasztható értékeket tartalmazza listát tartalmazó vezérlők esetén. (Listapanel, Kombi panel) A cellatartományt az Excelben megszokott módon a két sarok cellát kettősponttal elválasztva tudjuk megadni.
- A vezérlők méretét és elhelyezését a tervező nézet bekapcsolása után szabadon változtathatjuk az egér segítségével a vezérlők oldalain és sarkain lévő méretező négyzetek segítségével, illetve a vezérlő közepére kattintva húzással. Ha pontos méretet vagy elhelyezést szeretnénk, azt elérhetjük a Height (magasság), Width (szélesség), Top (lap tetejétől való távolság), Left (lap bal szélétől való távolság) tulajdonságok megadásával képpontokban.

A vezérlők segítségével készíthetünk olyan űrlapokat, melyek hátterében az Excel végez számításokat, de a kiinduló értékeket a felhasználó az általunk meghatározott keretek között adhatja meg.

Feladatok

1. Feladat

Egy üres Excel munkalapra helyezzük fel a felsorolt vezérlőket és a LinkedCell tulajdonságokat állítsuk be különböző cellákra. Ezután lépjünk ki a tervező módból és próbáljuk ki a vezérlőket. Láthatjuk, hogy a választókapcsolónál, a jelölőnégyzetnél és a váltókapcsolónál szürkén jelzi az Excel, ha a kapcsolt cella tartalmát kitörljük.

2. Feladat

Nyissuk meg a valutavalto.xls munkafüzetet és készítsünk egy valutaváltó űrlapot!

	A	B	C	D	E	F	G
1							
2		(forrás: www.otp.hu - 2004. szeptember 21.)					
3			Egy-		Valuta	Valuta	
4			ség	Közép	Vételi	Eladási	
5		AUD	1	141,88	137,62	146,14	
6		BGN	1	126,27	121,22	131,32	
7		CAD	1	156,58	151,88	161,28	
8		CHF	1	159,33	154,55	164,11	
9		CZK	1	7,87	7,48	8,26	
10		CSD	1	3,34	3,17	3,51	
11		DKK	1	33,20	32,20	34,20	
12		EUR	1	246,93	240,76	253,10	
13		GBP	1	262,01	252,96	271,06	
14		HRK	1	33,35	31,68	35,02	
15		JPY	100	185,15	178,63	189,67	
16		NOK	1	29,42	28,54	30,30	
17		PLN	1	57,32	55,31	59,33	
18		SEK	1	27,25	26,43	28,07	
19		SKK	1	6,18	5,87	6,49	
20		USD	1	202,79	197,72	207,86	
21							

Az átváltani kívánt valutaösszeg beviteli mezőbe kerüljön, mellette kombi panelből lehessen kiválasztani a valutanemet! A fenti két vezérlőt csatoljuk az Excel munkafüzet egy-egy cellájához, melyek alapján egy harmadik cellában FKERES függvényt használva határozzuk meg az eredményt. Ezt szintén beviteli mezőben jelenítsük meg, melynek Enabled tulajdonságát False-ra állítva megtilthatjuk a módosítást. *(Vegyük figyelembe a számításnál, hogy nem mindegyik valutánál ugyanaz az egység!)*

Megoldás:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2		(forrás: www.otp.hu - 2004. szeptember 21.)											
3			Egy-		Valuta	Valuta	1		SKK	=	5,87	Ft.	
4			ség	Közép	Vételi	Eladási							
5		AUD	1	141,88	137,62	146,14			GBP				
6		BGN	1	126,27	121,22	131,32			HRK				
7		CAD	1	156,58	151,88	161,28			JPY				
8		CHF	1	159,33	154,55	164,11			NOK				
9		CZK	1	7,87	7,48	8,26			PLN				
10		CSD	1	3,34	3,17	3,51			SEK				
11		DKK	1	33,20	32,20	34,20			SKK				
12		EUR	1	246,93	240,76	253,10			USD				
13		GBP	1	262,01	252,96	271,06							
14		HRK	1	33,35	31,68	35,02							
15		JPY	100	185,15	178,63	189,67							
16		NOK	1	29,42	28,54	30,30							
17		PLN	1	57,32	55,31	59,33							
18		SEK	1	27,25	26,43	28,07							
19		SKK	1	6,18	5,87	6,49							
20		USD	1	202,79	197,72	207,86							
21													
22													
23		Valuta	1										
24		Valutanem	SKK										
25		Forint	5,87										
26													

3. Feladat

Készítsük el a Forintról idegen valutára váltó verziót is!

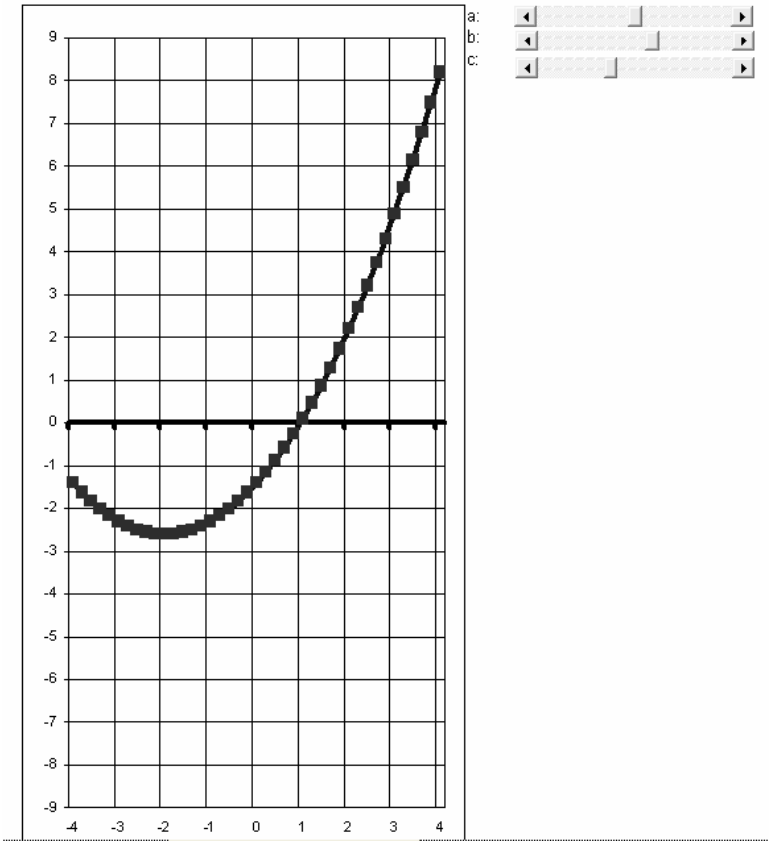
4. Feladat

Nyissuk meg az arajanlat.xls munkafüzetet és készítsünk árajánlatkérő űrlapot! A nyomdai munka ára: példányszám * oldalszám * darabár. erre jön példányonként a kötés ára és szintén példányonként a borító ára.

5. Feladat (+)

Készítsünk másodfokú függvényt ábrázoló diagrammot! Az x értékek változzanak –10 és +10 értékek között két tizedenként! Az f_x=a*(x+b)^2+c függvény a, b és c paramétereit gördítősávval lehessen megadni szintén –10 és +10 között egy tizedes változással! A diagrammot formázzuk úgy, hogy jól követhető legyen! *(Figyeljünk arra, hogy a gördítősáv alsó határértéke sem lehet negatív és a lépésköz is csak egész szám lehet, így nem tudjuk közvetlenül előállítani a feladat által kért értékeket csak külön cellákba transzformációval.)*

Megoldás: csuszka_diagram_mo.xls



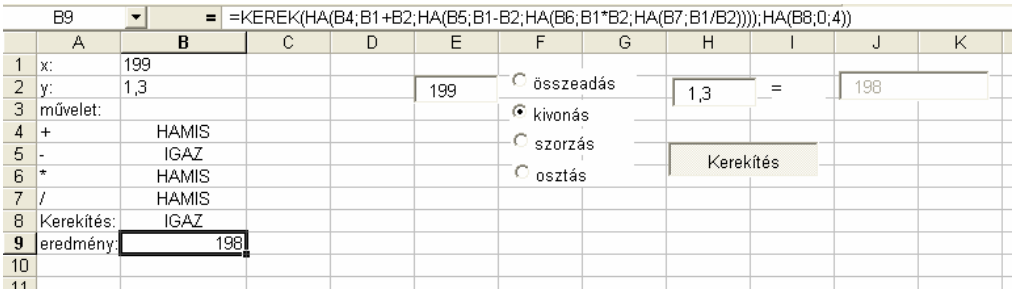
6. Feladat

Készítsünk űrlapot, amely segítséget nyújt betétlekötéshez! Nyissuk meg a betet.xls munkafüzetet és abba dolgozzunk! Hogy melyik bankba helyezzük el a betétet azt kombi panelből lehessen kiválasztani, az éves kamatot függvénnyel határozzuk meg a táblázatból, a havi kamat az éves kamat 1/12-e. A hónapokban mért futamidő megadásához használjunk gördítősávot (legfeljebb 3 évre lehessen számolni), a lekötött összeget pedig beviteli mezőbe lehessen beírni. A végösszeg kiszámításához a JBÉ pénzügyi függvényt használjuk (kamat, időszakok_száma, mai_érték)!

7. Feladat (+)

Készítsünk számológépet, ami a 4 alapműveletet tudja két szám között! A két számot beviteli mezővel lehessen megadni, a műveletet választókapcsolóval lehessen kiválasztani és legyen egy váltógomb, aminek benyomott állapota mellett egészre kerekíti az eredményt, egyébként pedig 4 tizedesre!

Megoldás:



Mit tanultunk meg

- A Vezérlők eszköztár ikonjai
- A vezérlők típusai és legfontosabb tulajdonságaik

3. Makrók Excelben

3.1. Előkészületek

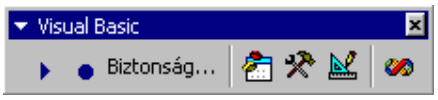
Elmélet

A makró egy rögzített műveletsorozat, melyet a felhasználó könnyedén újra és újra végre tud hajtatni a számítógéppel. Makrót létrehozhatunk rögzítéssel, vagy magunk is írhatjuk őket. A két módszert kombinálva is lehet használni.

A makrórögzítést általában arra használjuk, hogy az Excel által megírt programrészleteket később saját programjainkban felhasználjuk.

Makrókészítés előtt bizonyos előkészületeket kell tennünk:

Kapcsoljuk be a Visual Basic eszköztárat! Az eszköztár ikonjai megfelelnek az Eszközök menü Makró almenüjének menüpontjainak:



- Makró indítása (Eszközök/Makró/Makrók…): Ezzel a gombbal nyithatjuk meg azt a párbeszédpanelt, amelyben kiválaszthatjuk a már meglévő makróinkat és eldönthetjük, hogy mit akarunk velük csinálni:
  - Indítás: lefuttathatjuk az adott makrót
  - Mégse: bezárjuk a párbeszédpanelt
  - Lépésenként: az adott makrót soronként futtatjuk le
  - Szerkesztés: felnyílik a Visual Basic szerkesztő ablak, ahol a meglévő makrót módosíthatjuk

  - Létrehozás: ha nem a listából választunk, hanem egy új nevet írunk be, akkor megnyomhatóvá válik a Létrehozás gomb, ami az előző pontban is említett ablakot nyitja meg, ahol az általunk megadott néven létrejövő üres makrót szerkeszthetjük, vagyis írhatjuk meg
  - Törlés: a kiválasztott makrót törölhetjük vele
  - Egyebek: a kiválasztott makróhoz billentyűkombinációt rendelhetünk és megadhatunk egy leírást

- Makró rögzítése (Eszközök/Makró/Új makró rögzítése…): Ezzel a gombbal nyithatjuk meg azt a párbeszédpanelt, amelyben megadhatjuk a rögzítendő makró tulajdonságait:
  - nevét
  - billentyűparancsát
  - helyét
  - és leírását

- Biztonság... Biztonság (Eszközök/Makró/Biztonság…): Itt három biztonsági szint közül választhatunk. Számunkra a közepes szint a megfelelő.

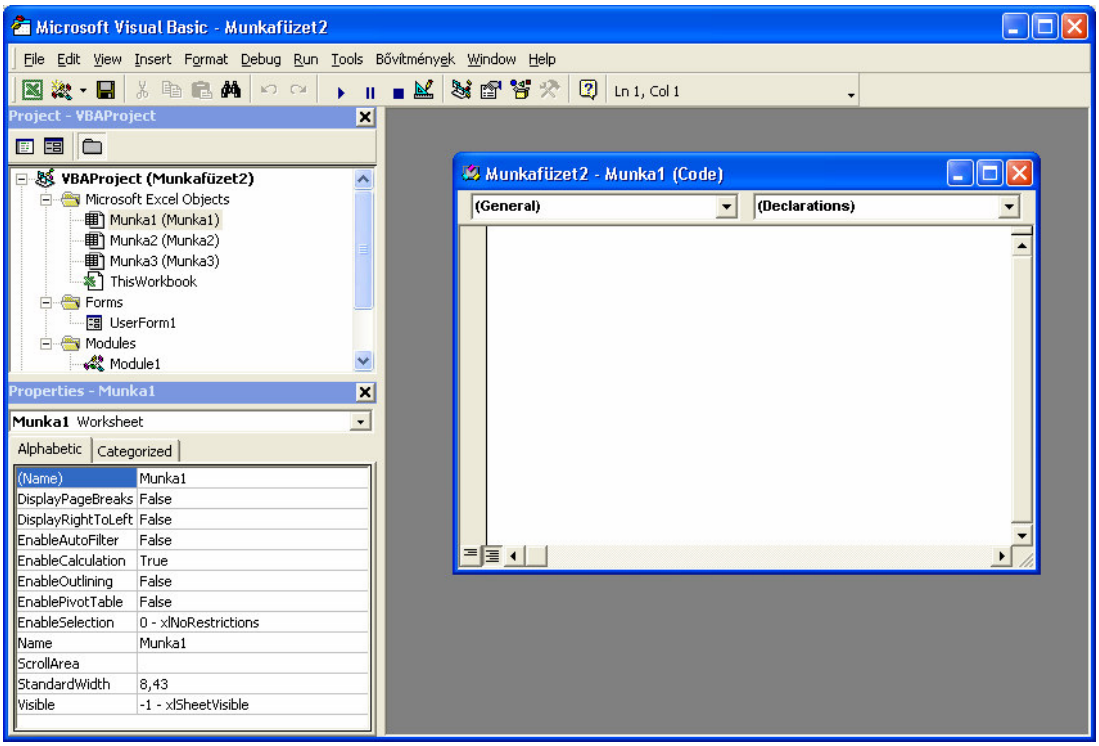
- Visual Basic (Eszközök/Makró/Visual Basic): Ezzel a gombbal nyithatjuk meg a Visual Basic szerkesztőfelületet.

- Vezérlők eszköztára: Ezzel a gombbal ki-be kapcsolhatjuk a Vezérlők eszköztárat.

- Tervező mód: Ezt a gombot már ismerjük a Vezérlők eszköztárról. Ezzel tudjuk ki-be kapcsolni, hogy a vezérlőinket szerkeszteni, vagy használni tudjuk.

- Microsoft Script Editor: HTML scriptek írásához nyújt segítséget. Ezt a gombot nem fogjuk használni.

Kattintsunk a már többször említett Visual Basic ikonra, és tekintsük meg részletesebben a megjelenő ablakot!



A Visual Basic szerkesztő egy új ablakban nyílik meg saját menüvel és eszköztárral. A View menüben bekapcsolhatjuk azokat az ablakokat, amelyek a fenti ábrán is láthatók:

- Project Explorer: Ebben az ablakban láthatjuk az összes megnyitott munkafüzetünk elemeit fa szerkezetben. Ha csak egy munkafüzetünk van nyitva és még nem írtunk makrót, akkor egy törzs van: VBAProject (Munkafüzet neve). Ebből a törzsből nyílik a Microsoft Excel Objects könyvtár, ami tartalmazza a munkafüzet munkalapjait egyesével zárójelben az általunk adott munkalapnevekkel, és egy ThisWorkbook nevezetű objektumot, ami az egész munkafüzetet együtt jelenti. Az Insert menüből, vagy az eszköztárról további elemeket szúrhatunk be a munkafüzetbe amelyek szintén könyvtárakba rendeződnek:
    - Moduls: ha beszúrunk egy modult, rögtön létrejön egy Moduls nevű könyvtár és ezentúl minden modulunk ide kerül.
    - Forms: ha beszúrunk egy UserFormot, rögtön létrejön egy Forms nevű könyvtár és ezentúl minden formunk ide kerül.
    - Az ablak tetején látható 3 ikon a következő:
        - View Code: a listából kiválasztott elem kódját mutatja a jobb oldali ablakban
        - View Object: a listából kiválasztott objektumot mutatja
        - Toggle Folders: a fent bemutatott mappaszerkezetet lehet vele elrejteni és akkor az objektumok „ömlesztve” jelennek meg
- Properties Window: Mindig a kiválasztott objektum tulajdonságait látjuk hasonlóan, mint az előző fejezetben megismert Tulajdonságok ablakban.

Makrót a Project Explorer ablaknál felsorolt minden elemhez írhatunk. Ha az ablakban a kiválasztott elemre duplán kattintunk, akkor jobb oldalt a szürke területen nyílik egy újabb ablak, ahol az adott elemhez tartozó makrók láthatók, illetve ide írhatók az új makrók.

Minden makrónak kell nevet adnunk. A makrónév egy szóból kell álljon, mely nem kezdődhet számmal, és nem tartalmazhatja a legtöbb írásjelet. Ékezetes betűk használata korábbi Windows verziókkal való kompatibilitás miatt nem ajánlott. Névnek már foglalt Visual Basic kulcsszavakat sem szabad adni, mert keveredéseket okozhat.

A makrók az Excel munkafüzettel együtt mentődnek.

Feladatok

8. Feladat

Nyissunk meg egy üres Excel munkafüzetet és a Visual Basic szerkesztőben hozzunk létre egy modult!

Mit tanultunk meg

- A Visual Basic eszköztár ikonjai
- A Visual Basic szerkesztő felépítése
- A makró neve

3.2. Makrók írása Excelben

Elmélet

Ha olyan makrót szeretnénk írni, ami nem használ vezérlőket és a munkafüzetünk minden munkalapján egyformán szeretnénk tudni futtatni, akkor egy modul elemhez kell írni. A makró kezdő- és zárósora mindig az alábbi:

```
Sub makrónév()
…
End Sub
```

A programkódban hivatkozhatunk a munkafüzetünk különböző részeire. A hivatkozást különbözőképpen használhatjuk. Mi az általunk programozás szempontjából legpraktikusabb módot foglaljuk össze a következő táblázatban:

	Az Excelbeli elnevezések	**Programkódbeli megfelelő**
1.	A B3 cella	Cells(3,2)
2.	A C4:G6 tartomány	Range(Cells(4,3),Cells(6,7))
3.	A B oszlop	Columns(2)
4.	A H, I, J oszlopokból álló tartomány	Range(Columns(8),Columns(10))
5.	A 2. sor	Rows(2)
6.	A 13.-tól a 16. sorig tartó tartomány	Range(Rows(13),Rows(16))
7.	A munkalap összes cellája	Cells

8.	Meghatározott munkalap cellái	Sheets(2).Cells
9.	Az éppen aktuális cella (ha tartomány van kijelölve, aktuális cella akkor is csak egy van, aminek a tartalma a szerkesztőlécben látszik)	ActiveCell
10.	Az éppen kijelölt objektum (cella, tartomány, rajz, diagramm)	Selection

A táblázat 8. sorában láthatunk egy példát arra (Sheets(2).cells), hogy az alárendelt objektumokat ponttal tudjuk elválasztani egymástól. A celláknak is vannak tulajdonságaik, mint a vezérlőknek. Programból úgy tudjuk ezeket a tulajdonságokat megadni, hogy a cella, vagy tartomány után írjuk ponttal elválasztva a tulajdonságot és = jel után adunk neki értéket. A leggyakrabban használt tulajdonságok:

- Value: a cella értéke, ha szöveg, akkor idézőjelek közé kell tenni. Ez az alapértelmezett tulajdonság, ezért ha nem írunk tulajdonságot, akkor ezt tudjuk megadni. A következő két sor eredménye ugyan az: az A1-es cellába az Alma felirat kerül:

```
Cells(1,1).Value = ”Alma”
Cells(1,1) = ”Alma”
```

- Interior.Color: a cella háttérszíne szövegesen megadva. Az angol színelnevezéseket lehet használni, csak elé kell írni egybe vele, hogy vb:

```
Cells(1,1).Interior.Color = vbRed
```

- Interior.ColorIndex: a cella háttérszíne számmal megadva. A Visual Basic 0-tól 56-ig számozza a színeket. Tehát az A1 cella háttérszínét a következőképpen is pirosra lehet állítani *(a 3-as a piros szín kódja)*:

```
Cells(1,1).Interior.ColorIndex = 3
```

- Font: a cella betűtípusa. Újabb ponttal elválasztva lehet megadni stílust, típust, színt a fent ismertetett módokon. A következő táblázatban példákat találunk az A1 cella betűformázásaira:

Betűtípus Arial-ra állítása	Cells(1,1).Font.Name = ”Arial”
Betűszín pirosra állítása	Cells(1,1).Font.Color = vbRed vagy Cells(1,1).Font.ColorIndex = 3
Betűméret 12-esre állítása	Cells(1,1).Font.Size = 12
Félkövér betűre	Cells(1,1).Font.Bold = True
Dőlt betűre	Cells(1,1).Font.Italic = True

Egy objektum több tulajdonságának megadásakor nem kell az objektum megnevezését minden sorban leírni, hanem a With, End With kulcsszavak között elég egyszer megadni, hogy melyik objektumról van szó és aztán csak a tulajdonságokat és a tulajdonságértékeket kell egymás alatt sorolnunk minden sort ponttal kezdve. Például ha egy cella betűtípusát a fenti táblázat alapján akarjuk beállítani, akkor azt megtehetjük a következőképpen is:

```
Sub formaz()
With Cells(1, 1).Font
    .Name = "Arial"
    .Color = vbRed
    .Size = 12
    .Bold = True
    .Italic = True
End With
End Sub
```

A programkódban használhatunk megjegyzéseket, hogy megírt eljárásainkat később is könnyen értelmezni tudjuk. A megjegyzéseket ’ jel után írjuk. Ezeket a program futásakor a számítógép nem veszi figyelembe és a kódban zöld színnel jelenik meg.

Feladatok

9. Feladat

Készítsünk makrót, amely az A1:D2 tartomány háttérszínét pirosra, betűszínét fehérre, betűtípusát 12-es Times New Roman dőltbetűsre állítja!

10. Feladat

Készítsünk makrót, amely a C oszlopot sárgára, a 4.-9. sorokat kékre, a metszetet pedig zöldre színezi!

11. Feladat

Készítsünk makrót, amely az egész munkalap formázását visszaállítja automatikusra (0-s háttérszín, 0-s betűszín, Arial betűtípus 10-es betűméret, nem dőlt, nem félkövér)

Megoldás:

```
Sub visszaallitas()
With Cells
    .Interior.ColorIndex = 0 'automatikus háttérszín
    .Font.ColorIndex = 0 'automatikus betűszín
    .Font.Name = "Arial" 'betűtípus
    .Font.Size = 10 'betűméret
    .Font.Bold = False 'nem félkövér
    .Font.Italic = False 'nem dőlt
End With
End Sub
```

12. Feladat

Készítsünk makrót, amely a kiválasztott objektum háttérszínét sárgára állítja. Próbáljuk ki különböző cellatartományok kijelölésével, illetve készítsünk egy diagrammot, és annak elemeit kijelölve is futtassuk le!

Mit tanultunk meg

- A makróírás nyitó és záró kulcsszava
- Az Excel celláira való hivatkozás módja
- A cellák tulajdonságai

3.2.1. Programozási struktúrák

Elmélet

Egy tevékenységsorozat (program) hívása során nem csak utasítások egymás utáni végrehajtását szeretnénk, hanem szükség lehet ismétlésekre, vagy valamely feltétel szerinti elágazásra (ilyennel már találkoztunk az Excel HA függvényénél.) Ilyenkor programjainkban az utasítások nem sorrendben egymás után hajtódnak végre (szekvencia), hanem használnunk kell az elágazásokat és a ciklusokat.

Elágazás: feltételhez kötjük, hogy melyik utasítás hajtódjon végre. Ennek szintaxisa Visual Basic-ben:

```
If feltétel Then
      utasítások, amik akkor futnak le, ha a feltétel igaz
Else
      utasítások, amik akkor futnak le, ha a feltétel hamis
End If
```

Ebből az If és az End If kötelező, az Else ág elhagyható, ilyenkor, ha a feltétel nem teljesül, akkor az End If utáni sorra ugrik a program.

Ciklus: ugyanazt az utasítássort szeretnénk többször megismételtetni a programmal. (A megfelelő kulcsszavak közé írt, ismétlendő utasítássort ciklusmagnak nevezzük.) Erre több lehetőségünk van. A legegyszerűbb, ha előre tudjuk pontosan, hogy hányszor szeretnénk futtatni az utasítássort. Ekkor használjuk a leszámláló (iterációs) ciklust. Ennek szintaxisa Visual Basic-ben:

```
For ciklusváltozó = kezdőérték To végérték Step lépésköz
      ciklusmag
Next
```

A ciklusváltozó bármilyen számváltozó lehet (leggyakrabban az i, j, k betűk valamelyikét használjuk). Első lefutáskor a változó a kezdőértéket veszi fel, aztán minden újabb lefutáskor lépésközzel változik az értéke, amíg meg nem haladja a végértéket. Amikor a változó értéke a végértéknél nagyobb, akkor a ciklusmag már nem fut le. (Ha nem adtunk meg lépésközt, akkor annak értéke: +1.

Előfordul, hogy nem tudjuk előre, hogy hányszor kell lefutnia az utasítássornak, hanem egy feltételhez akarjuk kötni a ciklus végét. Ilyenkor használjuk a feltételes ciklusok valamelyikét. Visual Basicben összesen 4 féle feltételes ciklus létezik, ezek közül kettőt fogunk megismerni. A ciklus elejét a Do, végét a Loop kulcsszó jelzi. Elöltesztelős ciklusnál a feltételt a ciklus elején a Do While kulcsszavak után adjuk meg. Ebben az esetben a feltétel teljesülése estén történik az ismétlés. (Előfordulhat, hogy már az első lefutáskor sem teljesül a feltétel, ilyenkor egyszer sem fut le a ciklus belsejében lévő utasítássorozat.) Hátultesztelős ciklusnál a feltételt a ciklus végén a Loop Until kulcsszavak után adjuk meg. Ebben az esetben mindaddig történik az ismétlés, amíg a feltétel hamis. A hátultesztelős ciklus mindig lefut legalább egyszer. A lehetséges szintaxisok:

```
Do While feltétel
      …
Loop
```

```
Do
      …
Loop Until feltétel
```

A leszámláló ciklus átírható feltételes ciklussá. Ez esetben létre kell hoznunk egy változót a ciklus előtt, és kezdőértéket kell neki adnunk, majd a ciklusmagban a változó értékének megfelelő növekedését, vagy csökkenését nekünk kell biztosítani. A feltételben azt kell megfogalmazni, hogy ha a változó meghaladta a végértéket, akkor legyen vége az ismétlésnek.

Az alábbi táblázatban egy példát látunk egy számlálós ciklus átírására a 2 feltételes ciklusba. Az utasítás az ossz változóba összegzi a számokat 1-től 10-ig.

```
ossz = 0
For i = 1 To 10
  ossz = ossz + i
Next
```

```
ossz = 0
i = 1
Do While i <= 10
  ossz = ossz + i
  i = i + 1
Loop
```

```
ossz = 0
i = 1
Do
  ossz = ossz + i
  i = i + 1
Loop Until i > 10
```

Általában úgy érdemes a ciklusokat használni, hogy csak akkor használunk feltételes ciklust, ha nem tudjuk előre, hogy hányszor kell lefutnia az utasítássornak.

Feladatok

13. Feladat

Készítsünk makrót, amely az A oszlopot feltölti az egész számokkal 0-tól 56-ig! A B oszlop celláinak háttérszíne legyen az A oszlopban lévő számoknak megfelelő!

14. Feladat

Készítsünk makrót, amely feltölti az A1:J10 tartományt a 10*10-es szorzótáblával! A 3-mal osztható számok hátterét színezzük sárgára, az egy maradékot adókét kékre, a 2 maradékot adókét zöldre!

15. Feladat

Nyissuk meg a tozsde.xls munkafüzetet! A tábla részvények alakulását mutatja. Színezzük a növekedők sorának hátterét zöldre, a csökkenőkét pirosra, a stagnálókét kékre!

16. Feladat

Nyissuk meg a valahol.xls munkafüzetet! Készítsünk makrót, ami megkeresi az „oszlop” munkalap A oszlopában található adatsort és beszínezi a hátterét sárgára. Az oszlopban az adatsor előtt szereplő cellák háttere legyen fekete! A makró indításakor mindig tűnjön el minden háttérszínezés! Úgy írjuk meg a programot, hogy bármilyen munkalapon lefuttatható lehessen, függetlenül attól, hogy hol van az adatsor és mekkora!

17. Feladat

Nyissuk meg a valahol.xls munkafüzetet! Készítsünk makrót, ami megkeresi a „táblázat” munkalapon található A1 cellától induló táblázat jobb alsó sarkát és az egész táblázat hátterét beszínezi sárgára. A makró indításakor mindig tűnjön el minden háttérszínezés! Úgy írjuk meg a programot, hogy bármilyen munkalapon lefuttatható lehessen, függetlenül attól, hogy mekkora a táblázat!

18. Feladat (+)

Nyissuk meg a valahol.xls munkafüzetet! Készítsünk makrót, ami megkeresi a „valahol” munkalapon található táblázatot és beszínezi a hátterét sárgára. A sorban és oszlopban a táblázat előtt szereplő cellák háttere legyen fekete! A makró indításakor mindig tűnjön el minden háttérszínezés! Úgy írjuk meg a programot, hogy bármilyen munkalapon lefuttatható lehessen, függetlenül attól, hogy hol van a táblázat és mekkora!

Megoldás:

```
Sub keres_tablazat()
Cells.Interior.ColorIndex = 0
i = 1             '(1,1) ponttól átlósan keressük az első nem üres cellát
j = 1
Do While Cells(i, j) = Empty
    i = i + 1      ' sorokon megyünk végig
    j = 1          ' sor első oszlopától indulunk
    Do While Cells(i, j) = Empty And j < i
        j = j + 1  'sor értékéig és a nem üres celláig lépkedünk
    Loop
Loop
tsork = i          'táblázat kezdő sora
toszlk = j         'táblázat kezdő oszlopa
'keressük a táblázat jobb alsó sarkát
'toszlk tsork-tól keressük, hogy melyik sorig van a táblázat
Do While Cells(i, toszlk) <> Empty
    i = i + 1
Loop
tsorv = i - 1      'megtaláltuk a tábla alatti első üres sort
Do While Cells(tsork, j) <> Empty
    j = j + 1
Loop
toszlv = j - 1     'oszlop, ameddig tart a táblázat
'színezzük számlálós ciklussal a táblázatot
For i = tsork To tsorv
    For j = toszlk To toszlv
        Cells(i, j).Interior.Color = vbYellow
    Next
Next
For i = 1 To tsork - 1        'tábla feletti sorok
    For j = 1 To toszlv       'tábla feletti oszlopok
        Cells(i, j).Interior.Color = vbBlack
    Next
Next
For i = tsork To tsorv      'tábla melletti sorok
    For j = 1 To toszlk - 1 'tábla elleti oszlopok
        Cells(i, j).Interior.Color = vbBlack
    Next
Next
End Sub
```

Mit tanultunk meg

- Az elágazás szintaktikája
- A számlálós ciklus szintaktikája
- A feltételes ciklusok szintaktikája

3.2.2. Néhány hasznos VB függvény

Elmélet

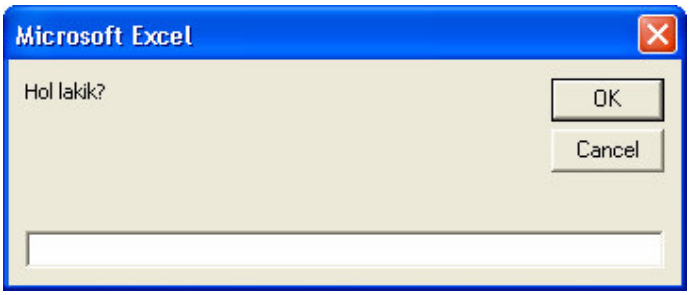
A program futása során kapcsolatot tarthatunk a felhasználóval. Kérhet tőle adatokat az InputBox függvény segítéségével és üzenhet neki a MsgBox függvény segítségével.

Az InputBox szintaxisa:

```
valtozo = InputBox(szöveg, cím, alapértelmezés)
```

Az InputBox egy kis ablak, ami megjelenik a képernyőn. Az ablakban lévő beviteli mezőbe írhat a felhasználó. A program a változóban tárolja szövegként, amit a felhasználó beírt. A

zárójelben felsorolt paraméterek közül a szöveget kötelező megadni. Ez fog megjelenni az ablakban a beviteli mező fölött. A cím lesz az ablak fejlécében, az alapértelmezés pedig a beviteli mezőben, de át lehet írni. A két gomb Ok és Cancel. Az Ok gomb megnyomásával bekerül a beviteli mező tartalma a változóba, a Cancel gomb megnyomásával a változó tartalma üres lesz.



A MsgBox szintaxisa:

```
MsgBox szöveg, gombok, cím
```

A MsgBox egy kis ablak, ami egy üzenettel jelenik meg a képernyőn. A zárójelben felsorolt paraméterek közül a szöveget kötelező megadni. Ez fog megjelenni az ablakban. A cím lesz az ablak fejlécében. A gomboknál azt lehet megadni, hogy milyen gombok jelenjenek meg az ablakban. Ha nem adunk meg semmit, vagy 0-t írunk, akkor csak Ok gomb lesz. *(Vigyázzunk arra, hogy az InputBoxnál kell zárójel, míg a MsgBox itt bemutatott használatánál nem!)*

Szövegek kiírásakor szükségünk lehet arra, hogy szó szerinti szövegeket és változótartalmakat együtt jelenítsünk meg. A szó szerint megjelenítendő szöveget idézőjelbe kell tenni, a változó nevét idézőjel nélkül kell leírni. Ha össze akarjuk fűzni őket, akkor az & jelet kell használnunk. Pl. ha egy varos nevű változóban tároljuk a város nevét, és szeretnénk egy üdvözletet küldeni, hogy „Isten hozta … városban!”, akkor azt a következőképpen tehetjük meg:

```
MsgBox „Isten hozta „ & varos & „ városban!”
```

Azért tettünk az idézőjelen belülre szóközt, hogy ne írjuk egybe a hozta, ill. a városban szót a város nevével.

A véletlen számok generálására szolgáló függvény az Rnd. Ez a függvény így argumentum nélkül használható és egy véletlen számot ad eredményül 0 és 1 félig zárt intervallumban, 0 lehet, 1 nem. Ennek a függvénynek a transzformálásával lehet megadni, hogy milyen számokat akarunk generálni. Például –10 és +10 közötti egész számokat szeretnénk, úgy hogy a –10 és a +10 is közte legyen:

```
(int(Rnd*11)-5)*2
```

Ahhoz, hogy az Rnd függvény valóban véletlen számokat generáljon a függvény használata előtt be kell írnunk a Randomize kulcsszót a programunkba. *(Mielőtt a Randomize kulcsszót beírnánk, teszteljük le a programunkat! A Randomize nélkül az Rnd függvény minden programfutáskor ugyanazt a véletlen számsort fogja generálni.)*

Feladatok

19. Feladat

Készítsünk makrót, amely InputBoxban bekéri a felhasználó nevét, aztán MsgBoxban köszönti a felhasználót a saját nevén!

20. Feladat

Egészítsük ki az előző feladatot úgy, hogy ismételje meg a név bekérését, ha a felhasználó a Cancel gombot nyomta meg!

Megoldás

```
Sub udvozlet()
Do
    nev = InputBox("Hogy hívnak?", "Kérdés")
Loop Until nev <> Empty
MsgBox "Szerbusz kedves " & nev & "!", 0, "Üdvözlet"
End Sub
```

21. Feladat

Készítsünk makrót, amely InputBoxban számokat kér be és a számokat egymás alá beírja az A oszlopba! Mindaddig folytassa a bekérést, amíg a felhasználó a Cancel gombot meg nem nyomja!

22. Feladat

Készítsünk makrót, ami –6 és +7 közötti véletlen egész számokkal feltölti az A oszlopot a 2.-tól a 18. sorig, majd a páros számok betűszínét kékre, a páratlanok háttérszínét pedig sárgára változtatja! Vigyázzunk arra, hogy ha többször egymás után le akarjuk futtatni a programot, akkor az előző formázásokat meg kell szűntetnünk a program elején!

Megoldás:

```
Sub veletlen()
Cells.Interior.ColorIndex = 0
Cells.Font.ColorIndex = 0
Randomize
For i = 2 To 18
    Cells(i, 1) = Int(Rnd * 14 - 6)
    If Cells(i, 1) Mod 2 = 0 Then
        Cells(i, 1).Font.Color = vbBlue
    Else
        Cells(i, 1).Interior.Color = vbYellow
    End If
Next
End Sub
```

23. Feladat

Készítsünk makrót, ami 10 és 30 közötti véletlen egész számokkal feltölti az A1:G10 cellatartományt, majd a páros számok hátterét sárgára színezi, a páratlanoknak pedig a betűtípusát megváltoztatja kék félkövérre, majd jelenjen meg egy MsgBox, ami közli, hogy hány darab páros számunk van! Vigyázzunk arra, hogy ha többször egymás után le akarjuk futtatni a programot, akkor az előző formázásokat meg kell szűntetnünk a program elején!

24. Feladat

Készítsünk makrót, ami –20 és +20 közötti véletlen páros számokkal feltölti az E5:K20 cellatartományt, majd a negatív számok hátterét sárgára, a nem negatívakét kékre színezi!

25. Feladat

Egészítsük ki az előző feladatot úgy, hogy a nullák háttere legyen zöld!

26. Feladat

Készítsünk makrót, ami InputBoxban bekér egy alsó, egy felső határt, majd egy lépésközt! Generáljon az A1 cellába egy a megadott feltételeknek megfelelő véletlen egész számot!

27. Feladat

Nyissuk meg az urlap.xls munkafüzetet! A munkafüzethez készítsünk új makrót! A makró ellenőrizze az űrlapot a következő szempontok alapján:

- Az irányítószám 1000 és 9999 közé essen
- A születés dátuma 1900 és 2004 közé essen
- Az összegek és különbségek értelemszerűen egyezzenek

Hiba esetén a kitöltő üzenet ablakban kapjon figyelmeztetést hibájáról!

28. Feladat (+)

Készítsünk makrót, ami sakktáblát készít a következő feltételekkel!

- A sakktábla első kockája az A1-es cella legyen!
- Az A1-es cella sötét legyen!
- A sakktábla szabályainak megfelelően váltakoznak a sötét és világos cellák!
- A cellák alakját ne változtassuk!
- A sakktábla méretét InputBoxban adhassa meg a felhasználó (ugyanannyi sor és oszlop legyen)!
- Próbáljuk meg a lehető legrövidebben megírni a makrót!

Mit tanultunk meg

- Az InputBox függvény
- A MsgBox függvény
- Szövegösszefűzés
- Az Rnd függvény

3.2.3. Változók

Elmélet

Programjaink során gyakran használunk változókat.

A következő táblázatban felsoroljuk a leggyakrabban használt változótípusokat Visual Basicben:

típusnév	mit tárol
string	szöveg
integer	egész szám (32768-ig)
single, double	valós szám
long	hosszú egész
date	dátum
boolean	igaz/hamis
variant	általános

Visual Basicben nem kötelező a változók deklarálása (definiálása). A változók variant típussal jönnek létre első használatkor. Ha a változót definiáltuk valamilyen típusra, akkor a megadott típusnak megfelelő műveleteket hajthatunk végre vele. Ez gondokat is okozhat, ha nem megfelelő tartalmat adunk meg.

Nézzünk a mi lesz a következő feladat eredménye különböző bemenő adatok hatására!

Egész számot várok a kért cellába

```
Sub beker()
Dim szam As Integer
szam = InputBox("Kérem a születési évedet!")
Cells(3, 4) = szam
End Sub
```

Beírt érték	Eredmény
1983	1983
1975,5	1975
alma	’Run-time error 13’ nem megadott típust kapott a rendszer

A problémán javíthatunk. Általános típust definiálunk (vagy elhagyjuk a definíciót), ez a Variant. A változó tartalmát ellenőrizhetjük, így csak a kívánt típust fogadjuk el. Először csak hibaüzenetként megjelenítjük, a következő programrészletben megmutatjuk, hogy hogyan javíthatjuk. A rosszul megadott típus így ellenőrizhető.

Az IsNumeric(Változó) függvény eredménye igaz, vagy hamis. Ha a változó tartalma numerikus érték eredménynek True-t (igazat) kapok. Ezt használhatjuk ellenőrzésre.

```
Sub bekervariant()
Dim szam As Variant
szam = InputBox("Kérem a születési évedet!")
If Not IsNumeric(szam) Then   ’szam változó tartalmát vizsgáljuk
    MsgBox ("Ismételje meg, nem számot adott")
End If
Cells(3, 4) = szam
End Sub
```

A fenti feladatot ciklusba szervezve, mindaddig bekéri az adatot, míg a megfelelő típusú értéket nem adjuk meg. Utána lehet akár nagyságrendet is vizsgálni.

```
Sub bekerciklus()
Dim szam As Variant
szam = InputBox("Kerem a születési évedet!")
Do While IsNumeric(szam) = False
    MsgBox ("Ismételje meg, nem számot adott")
    szam = InputBox("Kerem a születési évedet!")
Loop
Cells(3, 4) = szam
End Sub
```

Feladatok

29. Feladat

Készítsünk makrót, ami induláskor egy InputBoxban bekér egy számot, a számnak megfelelő méretű szorzótáblát készít az A1-es cellától kezdődően, majd egy újabb InputBoxban bekér egy újabb számot! Az újonnan beadott számnál nagyobb számok cellájának háttérszínét színezze pirosra!

30. Feladat

Egészítsük ki az előző feladatot úgy, hogy a kisebb számok háttere legyen sárga, az egyenlőkké pedig kék!

Mit tanultunk meg

- A változók használata
- A változók deklarálása
- A változók típusai

3.2.4. Algoritmusok

Elmélet

Az algoritmus egy feladat megoldására szolgáló egyértelmű szabályokkal követhető lépések (utasítások) sorozata.

Minimumkeresés: feltételezzük a halmaz első tagjáról, hogy az a legkisebb, és megjegyezzük az értékét. Végigmegyünk a halmazon és megvizsgáljuk, hogy találunk-e az eddig feltételezettnél kisebbet. Ha találtunk, akkor feltételezzük, hogy az a legkisebb és megjegyezzük az értékét. A halmaz végére biztosak lehetünk abban, hogy a feltételezett legkisebb valóban a halmazban előforduló legkisebb.

Természetesen a maximumkeresés is működik az előbb leírt módon.

Előfordulhat, hogy nem a legkisebb szám értékére vagyunk kíváncsiak, hanem arra, hogy a halmaz melyik eleme a legkisebb. Ilyenkor a feltételezett legkisebbnek nem az értékét, hanem a helyét jegyezzük meg.

Sorba rendezés buborék módszerrel: végigmegyünk a halmazon és páronként összehasonlítjuk az elemeket. Ha a két elem sorrendje megfelelő, akkor úgy hagyjuk őket, ha nem megcseréljük őket. Az összehasonlítgatást többször meg kell tennünk, hogy a végleges sorrend kialakuljon. Legrosszabb esetben elemszám-1-szer kell végigmennünk a halmaz elemein páronként összehasonlítva őket. Két elemet egy segédváltozó segítségével tudunk kicserélni egymással: először az egyik elemet kitesszük a segédváltozóba, aztán a másik elemet átírjuk az egyik helyére, majd a segédváltozóból visszaírjuk az egyik elemet a másik helyére.

A Visual Basic nem csak számokat tud összehasonlítani egymással, hanem karaktereket is az ASCII kódjuk alapján, így névsorba rendezés is lehetséges az angol ABC keretein belül. *(A magyar ABC különleges karakterei az ASCII kódjuk alapján nincsenek a „helyükön")*

Feladatok

31. Feladat

Nyissuk meg az adatbazis.xls munkafüzetet! Készítsünk makrót amely a G1 cellától kezdődően kiírja a legfiatalabb diák(ok) nevét és átlagát!

32. Feladat

Nyissuk meg az adatbazis.xls munkafüzetet! Készítsünk makrót amely a G1 cellától kezdődően kiírja a legrosszabbul tanuló diák(ok) nevét és születési dátumát!

33. Feladat

Nyissa meg az adatbazis.xls munkafüzetet! Másolja le az adatbázist a másolat munkalapra! Az adatbázis munkalapon rendezze az adatbázist makróval átlag alapján növekvő sorrendbe!

Mit tanultunk meg

- Minimum és maximumkeresés
- Sorba rendezés

3.2.5. Objektumokhoz kapcsolódó makrók

Elmélet

A 3.1. fejezet végén említettük, hogy nem csak modulba lehet makrót írni, hanem az egyes munkalapokhoz és a munkafüzethez is. Ha a makrókat az eddig megtanult módon, de nem modulba, hanem az egyik munkalaphoz írjuk, akkor akármelyik munkalapról indítjuk el a makrót, a cellahivatkozások mindig annak a munkalapnak a celláira fognak vonatkozni, amelyikről indítottuk őket (ha nincs bennük a munkalapot is megjelölő direkt hivatkozás).

Az 1. fejezetben vezérlőket helyeztünk el a munkalapokon, de nem írtunk hozzájuk programot. A munkalapokhoz írt makrókkal ezt is megtehetjük.

Nyissunk meg egy üres Excel munkafüzetet! A Munka1 munkalapra helyezzünk fel egy parancsgombot (CommandButton) a Vezérlők eszköztárról. Menjünk át a Visual Basic szerkesztőfelületre és válasszuk a Project Explorerben a Microsoft Excel Objects közül a Munka1 munkalapot, kattintsunk rá kétszer! A jobb oldalon megjelenő ablak tetején két legördülő listát látunk. A baloldaliban vannak a munkalaphoz kapcsolódó objektumaink, a jobboldaliban pedig a kiválasztott objektumhoz tartozó események. Ha legördítjük a baloldali listát, abban 3 elemet látunk:

- (General): zárójelben van, mert ez igazából nem objektum, ide lehet írni a globális változódeklarálásokat.
- CommandButton1: ez az a parancsgomb, amit az előbb felhelyeztünk. Ha lenne több vezérlőnk a munkalapon, azok mind megjelennének itt.
- Worksheet: maga a munkalap is egy objektum, amihez tartoznak események.

Eddigi makróinkat mindig menüből indítottuk. A makrókat azonban indíthatja egy-egy esemény is. Ezeket választhatjuk ki a jobb oldali listából. A programunk indulhat pl. egy

parancsgomb megnyomásakor, vagy egy cella tartalmának megváltozásakor automatikusan. Tekintsünk át néhány eseményt a parancsgombhoz ill. a munkalaphoz!

A parancsgomb néhány eseménye:

Click	kattintás
DblClick	duplakattintás
MouseMove	amikor az egérkurzor fölé kerül

A munkalap néhány eseménye:

SelectionChange	ha a munkalapon mást jelölünk ki
Change	ha bármelyik cella tartalma, vagy értéke megváltozik a munkalapon
BeforeDoubleClick	dupla kattintáskor a szerkesztő üzemmódba menetel előtt (természetesen nem a dupla kattintás előtt)
BeforeRightClick	jobb-egérgomb kattintáskor mielőtt a gyorsmenü megjelenik (természetesen nem a kattintás előtt)
Calculate	ha a munkalapon számítás történt

A Change és a Calculate eseményekkel nagyon kell vigyázni, mert ha olyan utasítássort kapcsolunk hozzájuk, ami egy újabb változást, újabb számolást okoz, akkor végtelen ciklushoz juthatunk. *(A végtelen ciklus leállítása a ctrl+Break billentyűkombinációval lehetséges)*

Feladatok

34. Feladat

Helyezzünk el a Munka1 munkalapon egy parancsgombot, a feliratát változtassuk meg arra, hogy „Szia”! A parancsgomb megnyomásakor jelenjen meg egy MsgBox a következő felirattal: „Üdvözöllek a VBA világában!”!

35. Feladat

Írjunk makrót a Munka1 munkafüzethez, ami egy cellára való dupla kattintáskor beleírja a nevünket az aktuális cellába!

Megoldás:

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
ActiveCell = "Neumann"
End Sub
```

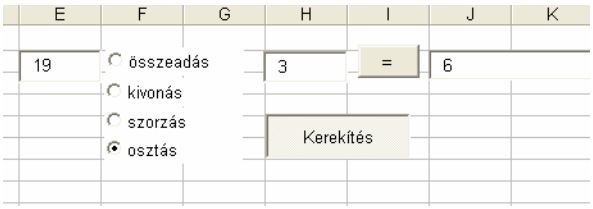
36. Feladat

Írjunk makrót, aminek a hatására, ha rákattintunk egy cellára, annak véletlenszerűen változtassa meg a háttérszínét!

37. Feladat (+)

Készítsük el a 7. feladatban látott számológépet makró írással! Az egyenlőségjel helyén legyen parancsgomb, melynek megnyomása indítja el a programot!

Megoldás:



A programkód:

```
Dim eredm As Double    'Azért itt deklaráljuk, hogy mindkét eljárásból
                       ' lássuk majd
```

```
Private Sub CommandButton1_Click()
Dim elso As Double     'Valósként deklaráljuk a változókat
Dim masodik As Double
If IsNumeric(TextBox1.Text) Then    'Vizsgáljuk, hogy a megadott
    elso = TextBox1.Text            'érték szám-e
Else
    MsgBox ("Nem számot írtál be az első mezőbe!")
End If
If IsNumeric(TextBox2.Text) Then
    masodik = TextBox2.Text
Else
    MsgBox ("Nem számot írtál be a második mezőbe!")
End If
If OptionButton1.Value = True Then  'Vizsgáljuk, hogy melyik
    eredm = elso + masodik          'opció van kiválasztva
End If
If OptionButton2.Value = True Then
    eredm = elso - masodik
End If
If OptionButton3.Value = True Then
    eredm = elso * masodik
End If
If OptionButton4.Value = True Then
    If masodik <> 0 Then      'Vizsgáljuk, hogy nem 0-val akarunk-e osztani
        eredm = elso / masodik
    Else
        MsgBox ("Nullával nem tudok osztani!")
    End If
End If
If ToggleButton1.Value = True Then  'Vizsgáljuk, hogy kerekített
    TextBox3.Text = Int(eredm)      'értéket várunk-e
Else
    TextBox3.Text = Round(eredm, 4)'A round függvény paraméterei
                                    'a kerekítendő szám és az,
                                    'hogy hány tizedesjegyet
                                    'jelenítsen meg
End If
End Sub
```

```
Private Sub ToggleButton1_Click()   'Ha már megvan az eredmény, akkor
                                    'is tudjuk változtatni, hogy
                                    'kerekítünk-e
If ToggleButton1.Value = True Then
    TextBox3.Text = Int(eredm)
Else
    TextBox3.Text = Round(eredm, 4)
End If
End Sub
```

38. Feladat

Nyissuk meg a jatek.xls munkafüzetet! A táblázatban rejtvényeket találunk, mindegyik megfejtése egy szám. Helyezzünk el a táblázat mellé egy parancsgombot „Ellenőrzés" felirattal! Ha a játékos megnyomja a gombot, a gép ellenőrizze, hogy a megoldásai jók-e! A helyes megoldások mellé írja ki hogy „Helyes"! Ha minden megfejtés megvan, MsgBox-ban gratuláljon is!

Ennyi játékos van egy focicsapatban.	11
Ennyi rabló volt Alibabához.	40
Ennyi byte egy Kilobyte	1024
Ennyi képviselői hely van a parlamentben	386
Az ötszög belső szögeinek összege	540
Ennyi másodperc egy óra	3600
Ebben az évben hat meg Karl Marx.	1883
Ennyi naponta van telihold	21
Ennyi ezer km a Föld sugara	6

39. Feladat

Javítsuk ki az előző feladatot úgy, hogy a játékot tetszőleges számú feladattal ki lehessen egészíteni, azaz az ellenőrzést az első üres sorig hajtsuk végre!

40. Feladat (+)

Módosítsuk a 36. feladatot úgy, hogy ha a játékos új megoldást ír a táblába, a gép rögtön ellenőrizze, hogy a megoldás jó-e! Ha igen írja a megoldás mellé, hogy „Helyes"! *(Vigyázzunk arra, hogy nehogy végtelen ciklusba bonyolódjunk! Ezt úgy tudjuk elkerülni, ha mindig csak annak a cellának az értékét változtatjuk, aminek tényleg változik az értéke, tehát ha egy cellában már szerepel, hogy „helyes", akkor nem írjuk ezt felül, vagy ha egy cella üres, akkor nem tesszük újra üressé.)*

41. Feladat

Az előző feladatot tovább szépíthetjük, ha nem azt írjuk a helyes megoldás mellé, hogy „Helyes", hanem egy pipát teszünk, *(a pipa a Wingdings betűtípus ü betűje)* valamint ha a Megoldás munkalapot elrejtjük. *(Munkalapot úgy tudunk elrejteni, hogy a Visual Basic szerkesztőben a Properties ablakban a munkalap Visible tulajdonságánál a 0xlSheetHidden értéket választjuk)*

Mit tanultunk meg

- A munkalap eseményei
- A parancsgomb eseményei

4. Függvények Excelben

Elmélet

Excelben saját függvényeket is készíthetünk. Ezek megjelennek a függvényvarázslóban is a Felhasználói függvénykategória alatt. A függvényeket külön modulba kell írni nem abba, ahol a saját makróink vannak, de egy modulba több függvény is írható. A függvény

névadására a makróknál tanultak érvényesek. A függvény kezdő és záró sora mindig az alábbi:

```
Function függvénynév()
…
End Function
```

Első lépésként hozzunk létre egy modult függvényeinknek. Az új modulunkba írjuk be a fenti két sort és nézzük meg Excelben, hogy mi történik! Excelben hívjuk meg a függvényvarázslót és válasszuk ki a függvénycsoportok közül az utolsót, a felhasználóit. A függvények között egyetlen függvény lesz felsorolva, az amit az előbb írtunk meg. Válasszuk ki! Megjelenik a szokásos ablak, ahol meg lehetne adni az argumentumokat, de helyette csak egy felirat van: „Ennek a függvénynek nincs argumentuma”. Persze, hogy nincs, hiszen nem adtuk meg, hogy mi legyen az. Azt is látjuk, hogy az előre számolt érték 0. Ez is azért van mert nem írtunk semmit a függvényünkbe. Nyomjuk meg a Kész gombot, majd módosítsuk a függvényünket!

Azt, hogy mi legyen a függvény eredménye úgy adhatjuk meg, hogy a függvény neve után írjuk egy = jel után. Tehát írjuk a következőt:

```
Function masodik()
masodik = ”Ez az eredmény”
End Function
```

Most már ez a függvény is benne lesz a függvényvarázslóban. Argumentuma ennek sincs, de az eredmény most már nem 0, hanem az a felirat, amit idézőjelben beírtunk. Így készíthetünk konstansfüggvényt.

Következő lépésként legyen a függvényünknek argumentuma is. Lehet egy vagy több, a függvény neve után a zárójelbe adhatjuk meg őket vesszővel elválasztva. Első példánkban egyetlen argumentumunk lesz és a függvény annyit fog tenni, hogy az argumentumot adja vissza eredményül.

```
Function ugyanaz(arg)
ugyanaz = arg
End Function
```

Készíthetünk olyan függvényt is, amelyiknek nem csak egy érték az eredménye, hanem egy tömb. Ezeket tömbfüggvénynek hívjuk. Ilyenkor a függvény neve és az egyenlőségjel után az Array kulcsszót írjuk és zárójelben vesszővel elválasztva soroljuk fel a tömb elemeit. A függvény kipróbálásakor a Gyakoriság függvénynél megismert módszert kell használni: F2, Ctrl+Shift+Enter. Oda kell azonban figyelnünk, mert a Gyakoriság függvény függőlegesen adta az eredményeket, míg saját tömbfüggvényünk vízszintesen fogja.

```
Function negyzet_kob (szam)
negyzet_kob = Array(szam^2, szam^3)
End Function
```

A fenti függvény a megadott szám négyzetét és köbét adja eredményül tömbként.

Feladatok

42. Feladat

Készítsünk konstansfüggvényt, ami az e-t adja eredményül! (e értéke: 2,718282)

43. Feladat

Készítsünk függvényt, ami a sugár megadása után kiszámítja a gömb térfogatát!

44. Feladat

Készítsünk tömbfüggvényt, ami az oldalak megadása után kiszámítja a téglalap kerületét és területét!

45. Feladat

Készítsünk függvényt, ami kiszámítja a megadott szám faktoriálisát!

46. Feladat

Készítsünk függvényt, ami egy vezetéknév és egy keresztnév megadása után előállítja a teljes nevet! Vigyázzunk, hogy a név két része ne legyen egybeírva!

47. Feladat

Készítsünk tömbfüggvényt, ami az első elem, a hányados és az elemszám megadása után tömbként visszaadja a mértani sorozat első és utolsó elemét!

48. Feladat

Készítsünk tömbfüggvényt, ami az $a*x^2+b*x+c=0$ másodfokú egyenletet oldja meg az a, b és c paraméterek megadása után! A megoldásra nem vezető paraméterek esetén értelmes, a hiba típusára utaló hibaüzenetet kapjunk!

49. Feladat

Készítsünk függvényt ami kiszámítja a Fibonachi sor n-edik elemét. (Fibonachi sor első két eleme 1, 1, aztán minden új elem az előző két szám összege.)

Mit tanultunk meg

- A függvények nyitó és záró kulcsszava
- A függvény bemenő paramétereinek megadása
- A függvény eredményének megadása
- A tömbfüggvény készítése