

Változók használata

A paraméter egy adott eljáráshoz tartozó adat. Szükségünk lehet azonban az eljárásokhoz szorosan nem kötődő adatok tárolására is. Ezt valósíthatjuk meg a változók segítségével.

Példa: Lufik

A következő példában a **lufi** eljárás egy tetszőleges színű, 60 pont méretű lufit rajzol, amely egy 100 pont hosszú madzag végén van. A lufi megrajzolása után a teknőc visszatér a kiindulási pontba. A **lufik** eljárás pedig ezt hívja meg 20-szor, előtte azonban a teknőcöt tetszőleges szöggel elfordítja. Így egy maréknyi egyforma méretű lufit kapunk.



```
főablak (ablak) változtatása
Alapok  Megjelenés  Események
lufi    eljárás lufi
lufik   tsz? tetsz
        e 100
        pontméret 60
        h 100
        vége
```

```
főablak (ablak) változtatása
Alapok  Megjelenés  Események
lufi    eljárás lufik
lufik   törölkép
        ism 20
        [
          b tetsz
            lufi
        ]
        vége
```

Készítsd el a „lufirajzó” programot! *Hogyan lehetne úgy átírni, hogy a lufik mérete és a madzag hossza mindig tetszőleges legyen?*

Hogyan tudnánk elérni, hogy a lufik mérete változzon? Ha a lufi méretének megadására a **pontméret 20 + tetsz** utasítást használjuk, akkor a lufik mérete tetszőleges, de legalább 20 pont nagyságú lesz. *Próbáld ki!*

Vajon el tudjuk-e ugyanígy érni azt is, hogy a madzag hossza tetszőleges legyen? Írjuk az **e 100** helyett **e 100 + tetsz**, a **h 100** helyett pedig **h 100 + tetsz** utasítást! Az eredményt az ábrán látjuk, az bizony nem egy „luficsokor”.

A hiba oka nyilván az, hogy a **tetsz** értéke más lesz, amikor a teknőc előre megy, és más, amikor hátra. Vajon hogyan tudnánk megoldani, hogy a **100 + tetsz** értéket a teknőc megjegyezze, és mindkét esetben ezt használja?



```
főablak (ablak) változtatása
Alapok  Megjelenés  Események  Változók
lufi    eljárás lufi
lufik   tsz? tetsz
        globvál "m 100 + tetsz
        e :m
        pontméret 20 + tetsz
        h :m
        vége
```



A **100 + tetsz** értéket az **m** változó tartalmazza. Előre és hátra így ugyanígy megy a teknőc.

A programozási nyelvekben az adatokat *változók* segítségével tárolhatjuk a memóriában. A változókat a nevük azonosítja.

A Logo nyelvben például a `globvál "m 100` létrehoz egy *m* nevű változót, melynek értéke 100. Az `e :m` utasítás hatására pedig a teknőc annyit lép előre, amennyi az *m* változó értéke.

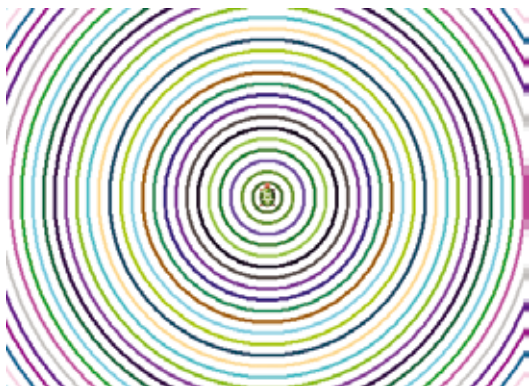
Figyeljünk arra, hogy amikor a változó értékét módosítjuk, akkor a neve elé tegyünk egy macskakörmöt, amikor pedig lekérdezzük, akkor egy kettőspontot!

Ezek után már érthető az ábrán látható eljárás. A `globvál "m 100 + tetsz` sorban létrehozuk az *m* nevű változót, melynek értéke `100+tetsz`. Az *m* változó értéke mindaddig nem változik, amíg új értéket nem adunk neki. Így a teknőc a 4. és a 6. sorban ugyanannyi lépést tesz meg előre, illetve hátra.

Körök

A változó értékét az eljárásban akár módosíthatjuk is. A következő példában koncentrikus köröket rajzolunk, egyre nagyobb sugárral.

A kör sugarát az *r* változó tartalmazza. Az *r* értéke kezdetben 10, majd minden kör megrajzolása előtt 20-szal nő.



```
főablak (ablak) változtatása
Alapok  Megjelenés  Események  Változók
körök   eljárás körök
globvál "r 10
ism 100
[
globvál "r :r+20
tss! tetsz
kör :r
]
vége
```

A kör sugarát az *r* változó tartalmazza, melynek értékét minden kör megrajzolása előtt növeljük. Mit jelent a macskaköröm, illetve a kettőspont a `globvál "r :r+20` sorban?

Kérdések, feladatok

1. Készíts programot, amely kirajzol egy cseresznyét a képernyőre! Módosítsd úgy a programot, hogy a cseresznyék szára véletlen hosszú, de legalább 50 pont legyen! Egészítsd ki egy újabb eljárással a programot úgy, hogy az szórja tele a képernyőt cseresznyékkel!
2. Készíts körsort úgy, hogy a körök az ábrán látható módon, egy pontban találkozzanak!
3. Miben tud „többet” a változó, mint a paraméter?

