Programozási nyelvek
5. előadás

Logo – funkcionális programozás

Elemi függvények:

```
eljárás kotangens :x
 eredmény (cos :x)/sin :x
vége
```

```
eljárás pi
 eredmény 3,14159
vége
```

```
eljárás abs :x
 hak :x<0 [eredmény -:x] [eredmény :x]
vége
```

```
eljárás fakt :n
 hak :n=0 [eredmény 1] [eredmény :n*fakt :n-1]
vége
```

Függvény összetételi módok:

kompozíció, alternatíva, rekurzió.

Logo – funkcionális programozás

Adatok Logo-ban:

- Szám: számjegyek
- Szó: "karakterek
- Mondat: [szavak és számok]
- Bekezdés: [[első mondat] ... [utolsó mondat]]
- ... a zárójelezés akármeddig folytatható

Az egyes adatok kezelésére függvényeket írhatunk.

Logo – funkcionális programozás

Első szó:

Mivel a mondat szavak sorozata, ezért van értelme a mondat első szaváról beszélni.

```
eljárás elsőszó :mondat
  eredmény első :mondat
vége
```

Példa:

```
elsőszó [Imagine Logo]   →   Imagine
```

Logo – funkcionális programozás

Első betű:

Az első betű ugyanúgy első eleme a szónak, ahogyan az első szó az első eleme volt a mondatnak.

```
eljárás elsőbetű :szó
  eredmény első :szó
vége
```

Példa:

```
elsőbetű "Imagine   →   I
```

Ahogyan van `első` függvény, van `utolsó` függvény is.

Logo – funkcionális programozás

Mondat ciklikus léptetése szavanként eggyel balra:

A ciklikus balra léptetés alapelve: vegyük le a mondat első szavát, majd az így kapott, egy szóval rövidebb mondat végére (azaz utolsó szavának) tegyük vissza!

```
eljárás balralép :mondat
  eredmény utolsónak első :mondat
                    elsőnélküli :mondat
vége
```

Példa:

```
balralép [Logo programozási nyelv] →
         [programozási nyelv Logo]
```

Logo – funkcionális programozás

Ciklikus léptetés szavanként eggyel jobbra:

A megoldása hasonló az előző feladathoz, az ott alkalmazott függvények megfelelő párját kell használni. (`elsőnélküli` → `utolsónélküli`, `utolsónak` → `elsőnek`)

```
eljárás jobbralép :mondat
  eredmény elsőnek utolsó :mondat
                   utolsónélküli :mondat
vége
```

Példa:

```
jobbralép [Logo programozási nyelv]      →
          [nyelv Logo programozási]
```

Logo – funkcionális programozás

Fordítsuk meg egy mondat szavai sorrendjét:

Ha a mondat egyetlen szót sem tartalmaz, azaz üres, akkor a megfordítás önmaga. Ha legalább egy szó van benne, akkor vegyük ki az első szót, fordítsuk meg a maradék mondatot, majd tegyük az eredmény végére a kivett első szót.

```
eljárás mondatfordít :mondat
  ha üres? :mondat [eredmény :mondat]
  eredmény utolsónak első :mondat
           mondatfordít elsőnélküli :mondat
vége
```

```
mondatfordít [ez egy mondat] →
    [mondat egy ez]
```

Logo – funkcionális programozás

Egy szó összes magánhangzóját cseréljük e-betűre:

Menjünk végig a szó betűin! Ha magánhangzót találunk, akkor az eredménybe e betűt tegyünk, mássalhangzó esetén pedig magát a betűt!

```
eljárás csere :szó
  ha üres? :szó [eredmény :szó]
  ha eleme? első :szó magánhangzók
    [eredmény elsőnek "e csere elsőnélküli :szó]
  eredmény elsőnek első :szó
                    csere elsőnélküli :szó
vége
```

Egyetlen probléma: `magánhangzók` nevű művelet/konstans nincs a Logo-ban.

Logo – funkcionális programozás

Konstans:

Funkcionális nyelvek konstansa a konstans függvény:

```
eljárás magánhangzók
  eredmény [a á e é i í o ó ö ő u ú ü ű]
vége
```

vagy

```
eljárás magánhangzók
  eredmény "aáeéiíoóöőuúüű
vége
```

Logo – funkcionális programozás

Egy szó hangrendjének megadása szótagonként:

A szótagok hangrendjéhez a szó magánhangzóit kell megvizsgálni. Az a, á, o, ó, u, ú mély magánhangzók, a többiek pedig magasak.

```
eljárás mélyek
  eredmény [a á o ó u ú]
vége
eljárás magasak
  eredmény [e é i í ö ő ü ű]
vége
```

Logo – funkcionális programozás

Egy szó hangrendjének megadása szótagonként:

```
eljárás hangrendek :szó
  ha üres? :szó [eredmény :szó]
  ha eleme? első :szó mélyek
     [eredmény elejére "mély
                hangrendek elsőnélküli :szó]
  ha eleme? első :szó magasak
     [eredmény elejére "magas
                hangrendek elsőnélküli :szó]
  eredmény hangrendek elsőnélküli :szó
vége
```

Hangrendek ”alma → [mély mély]

Logo – funkcionális programozás

Utasítás összefoglaló:

`eredmény`	a Logo függvények eredménye következik mögötte
`üres? :sorozat`	igaz, ha a paramétere üres sorozat, hamis, ha nem
`eleme? :elem :sorozat`	igaz, ha az elem eleme a sorozatnak
`első :sorozat`	a sorozat első elemét adja
`utolsó :sorozat`	a sorozat utolsó elemét adja

Logo – funkcionális programozás

Utasítás összefoglaló:

`elsőnélküli :sorozat`	a sorozat első elem nélküli részsorozatát adja
`utolsónélküli :sorozat`	a sorozat utolsó elem nélküli részsorozatát adja
`elsőnek :elem :sorozat`	a sorozat első eleme elé szúrja be az új elemet
`utolsónak :elem :sorozat`	a sorozat utolsó eleme mögé szúrja be az elemet

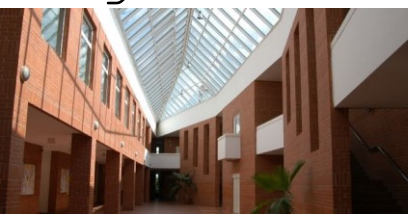
Logo – programozási tételek

Összegzés tétel:

```
eljárás összeg :x
 ha üres? :x [eredmény 0]
 eredmény (első :x)+összeg elsőnélküli :x
vége
```

Megszámolás tétel:

```
eljárás számol :x
 ha üres? :x [eredmény 0]
 ha t első :x [eredmény 1+számol elsőnélküli :x]
 eredmény számol elsőnélküli :x
vége
```



Logo – programozási tételek

Maximumkiválasztás tétel:

```
eljárás maximum :x
 ha üres? elsőnélküli :x [eredmény első :x]
 ha (első :x)>maximum elsőnélküli :x
    [eredmény első :x]
 eredmény maximum elsőnélküli :x
vége
```

Probléma:

Ez a függvény rossz esetben minden hívásakor kétszer hívja saját magát!

Logo – programozási tételek

Maximumkiválasztás tétel:

```
eljárás maximum :x
 ha üres? elsőnélküli :x [eredmény első :x]
 eredmény nagyobb első :x maximum elsőnélküli :x
vége
eljárás nagyobb :a :b
 ha :a>:b [eredmény :a]
 eredmény :b
vége
```

Logo – programozási tételek

Eldöntés tétel:

```
eljárás eldönt :x
 ha üres? :x [eredmény "hamis]
 ha t első :x [eredmény "igaz]
 eredmény eldönt elsőnélküli :x
vége
```

Logo – programozási tételek

Kiválasztás tétel:

```
eljárás kiv :x
 ha t első :x [eredmény első :x]
 eredmény kiv elsőnélküli :x
vége
eljárás kiv :x
 ha t első :x [eredmény lista első :x 1]
 eredmény növel kiv elsőnélküli :x
vége
eljárás növel :pár
  eredmény lista első :pár 1+utolsó :pár
vége
```

Logo – programozási tételek

Keresés tétel:

```
eljárás keres :x
 ha üres? :x [eredmény [hamis]]
 ha t első :x [eredmény lista "igaz első :x]
 eredmény keres elsőnélküli :x
vége
```

Logo – programozási tételek

Kiválogatás tétel:

```
eljárás kiválogat :x
 ha üres? :x [eredmény []]
 ha t első :x [eredmény elsőnek első :x
                          kiválogat elsőnélküli :x]
 eredmény kiválogat elsőnélküli :x
vége
```

Logo – szövegvezérelt rajzolás

Ovilogó

Az első osztályosok számára készült Logo nyelv egyszerűbb az általunk használtnál. Összesen 4 utasítást, valamint egyjegyű egész számokat ismer:

E: előre lép 10*X egységet

H: hátra lép 10*X egységet

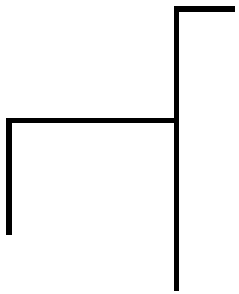
B: balra fordul 90 fokot

J: jobbra fordul 90 fokot

0..9: a lépés egységét (X-et) az adott számjegyre változtatja

Példa:

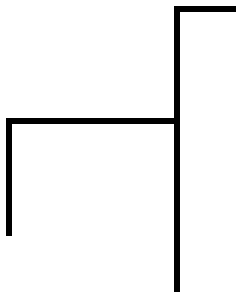
```
ovi "EEJEEEJ3E5HB1E 1
```



Logo – szövegvezérelt rajzolás

Ovilogó

```
eljárás ovi :szó :x
 ha nem üres? :szó
   [hakülönben szám? első :szó
       [ovi elsőnélküli :szó első :szó]
       [elvégez első :szó :x
        ovi elsőnélküli :szó :x]]
vége
```



```
eljárás elvégez :betű :x
 hakülönben :betű="E [előre :x*10]
  [hakülönben :betű="H [hátra :x*10]
    [hakülönben :betű="B [balra 90]
      [ha :betű="J [jobbra 90]]]]
vége
```

Logo – szövegvezérelt rajzolás

Morze

A Morze ábécében az egyes betűket hosszú és rövid vonalakkal jelöljük. A következő feladatban az alábbi betűket használjuk:

a: • – o: – – – p: • – – • r: • – • t: –

Példa:

```
szórajzol "por      • – – •  – – –  • – •
szórajzol "tar      –  • –  • – •
```

Logo – szövegvezérelt rajzolás

Morze

```
eljárás szórajzol :szó
 ha nem üres? :szó [betű első :szó szünet
                    szórajzol elsőnélküli :szó]
vége
eljárás betű :b
 ha :b = "a [rövid hosszú]
 ha :b = "o [hosszú hosszú hosszú]
 ha :b = "p [rövid hosszú hosszú rövid]
 ha :b = "r [rövid hosszú rövid]
 ha :b = "t [hosszú]
vége
```

Logo – szövegvezérelt rajzolás

Morze

```
eljárás rövid
 előre 1 szünet
vége
eljárás hosszú
  előre 5 szünet
vége
eljárás szünet
 tollatfel előre 5 tollatle
vége
```

Logo – szövegvezérelt rajzolás

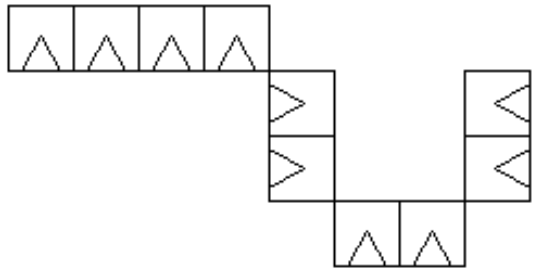
Meander

Meandernek nevezik az olyan sormintákat, amelyek valamilyen szabályszerűség szerint kanyarognak.

A meander az alapelemet egy szöveg karakterei szerinti sorrendben ismétli. A meander először jobbra indul. Ha az X betű következik, akkor a haladási irányt megtartja; a J betű hatására az irány jobbra változik 90 fokkal, a B hatására pedig balra 90 fokkal.

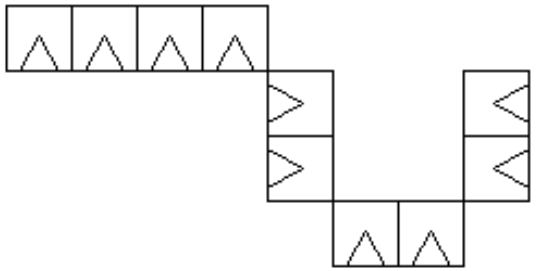
Példa:

```
meander 15 "XXXJXBXBX
```



Logo – szövegvezérelt rajzolás

Meander



```
eljárás meander :h :sz
 alap :h
 ha nem üres? :sz [mozdul első :sz
                   meander :h elsőnélküli :sz]
vége
eljárás mozdul :b
 ha :b="X [jobbra 90 előre :h balra 90]
 ha :b="J [jobbra 90 előre :h]
 ha :b="B [jobbra 90 előre :h balra 90 előre :h
           balra 90 hátra :h]
vége
```

Logo – tanulás

Tanulás

A teknőcöt a JBEH parancsokkal vezéreljük. J hatására jobbra, B hatására balra fordul 90 fokkal, E hatására előre lép egyet, H hatására pedig hátra. A beírt parancsokat rajzolás közben megjegyezzük mindaddig, amíg V betűt nem nyomunk. A T betűre töröljük a képernyőt! Ezután az R betű lenyomására a képernyő közepéről indulva rajzoljuk ki a megjegyzett rajzot, az A betű hatására az y-tengelyre vett tükörképét, a B betű hatására az x-tengelyre vett tükörképét, a C betűre pedig az origóra vett tükörképét! A következő V betűre a program fejeződjön be!

Logo – tanulás

```
eljárás tanulás
 rajzol feldolgoz olvasjel olvasjel
vége
eljárás rajzol :szó :kar
 ha :kar="T [törölkép]
 ha :kar="R [kirajzol :szó]
 ha :kar="A [kirajzol tükrözy :szó]
 ha :kar="B [kirajzol tükrözx :szó]
 ha :kar="C [kirajzol tükrözo :szó]
 ha :kar<>"V [rajzol :szó olvasjel]
vége
```

Logo – tanulás

```
eljárás kirajzol :szó
 ha nem üres? :szó [ha első :szó="J [jobbra 90]
                    ha első :szó="B [balra 90]
                    ha első :szó="E [előre 10]
                    ha első :szó="H [hátra 10]
                    kirajzol elsőnélküli :szó]
vége
```

Logo – tanulás

```
eljárás feldolgoz :kar
 ha :kar="V [eredmény []]
 ha :kar="J [jobbra 90]
 ha :kar="B [balra 90]
 ha :kar="E [előre 10]
 ha :kar="H [hátra 10]
 eredmény elsőnek :kar feldolgoz olvasjel
vége
```

Logo – tanulás

```
eljárás tükrözo :szó
 ha üres? :szó [eredmény :szó]
 ha "E=első :szó [eredmény elsőnek "H
                          tükrözo elsőnélküli :szó]
 ha "H=első :szó [eredmény elsőnek "E
                          tükrözo elsőnélküli :szó]
 eredmény elsőnek első :szó
                    tükrözo elsőnélküli :szó
vége
```


Programozás nyelvek
5. előadás vége