Bevezetés a programozásba

2. Előadás
Programkonstrukciók:
Elágazások, ciklusok

ISMÉTLÉS

Kifejezések: a helyes kifejezés

- Olyan műveleti jeleket és értékeket tartalmaz, amiknek van együtt jelentése
- ***Típus***nak fogjuk nevezni az értékhalmaz és a művelethalmaz együttesét
- Magának a kifejezésnek is van típusa, például az „1+1” egy szám típusú kifejezés
- Összetett kifejezéseket is lehet fogalmazni, például „**8+3*(3+7)**”, ilyenkor fontos, hogy helyes részkifejezésekből álljon.

ISMÉTLÉS

Változók

- **Névvel jelölt, adott típushoz tartozó elem**
- Tehát a változónak van
  - Neve
  - Típusa
- Az „1+x” csak akkor értelmes kifejezés, ha „x” olyan változó, amihez van olyan művelet, hogy „számmal összeadás”. Például x szám.
- Ahhoz, hogy egy programban/kifejezésben változót használhassunk, először jeleznünk kell
- Ezt ***deklaráció***nak nevezik

ISMÉTLÉS

Kimenet, bemenet

- Változókat értékadáson kívül a külvilággal való kapcsolattartásra használhatunk
- A bemenet (input) jelentése (egyelőre) az, hogy a program felhasználója ad értéket a változónak
- A kimenet (output) jelentése (egyelőre) az, hogy egy kifejezés eredményét megörökítjük a külvilágnak, például képernyőre írással
- **`VÁLTOZÓK: a :EGÉSZ`**
  `BE: a`
  `KI: a+1`

ISMÉTLÉS

Változók, értékadás

- **`VÁLTOZÓK: a: EGÉSZ`**
  `a := 1`
- Az értékadás a változó tartalmát megváltoztatja egy kifejezés eredményére:
  `a := 8+3*(3+7)`
- **`a := a + 1`**
- **`a := b + c + d`**
- Az alábbi kifejezés csak akkor értelmes, ha „b”, „c” és „d” már deklarált változók, amiknek megfelelő a típusuk

Egyszerű program

```
PROGRAM hellóvilág
  KI: "Helló világ"
PROGRAM_VÉGE
```

Szekvencia

```
PROGRAM szekvencia
  VÁLTOZÓK:
    a: EGÉSZ

  BE: a
  a := a + 1
  a := a * 2
  KI: a
PROGRAM_VÉGE
```

Elágazás

```
PROGRAM elágazás
  VÁLTOZÓK:
    a: EGÉSZ

  BE: a
  HA a > 0 AKKOR
    KI: "pozitív"
  HA_VÉGE
PROGRAM_VÉGE
```

Elágazás

```
PROGRAM elágazás
  VÁLTOZÓK:
    a: EGÉSZ

  BE: a
  HA a > 0 AKKOR
    KI: "pozitív"
  KÜLÖNBEN
    KI: "nem pozitív"
  HA_VÉGE
PROGRAM_VÉGE
```

Elágazás

```
PROGRAM elágazás
  VÁLTOZÓK:
    a: EGÉSZ

  BE: a
  HA a > 0 AKKOR
    KI: "pozitív"
  KÜLÖNBEN
    HA a < 0 AKKOR
      KI: "negatív"
    KÜLÖNBEN
      KI: "nulla"
    HA_VÉGE
  HA_VÉGE
PROGRAM_VÉGE
```

Elágazás: fogalmak

```
PROGRAM
. . .

   HA  feltétel
       feltétel ága
  (KÜLÖNBEN
       különben ág, „else ág”)
   HA_VÉGE
. . .
PROGRAM_VÉGE
```

Specifikáció

- A specifikáció lényege, hogy a feladatot a lehető legprecízebben megfogalmazzuk
- Az a feladat, hogy „adjuk meg egy szám gyökét”, pontosításra szorul: mi van, ha a szám negatív?
  - a program futásidejű hibával leáll
  - a program nem ad semmilyen eredményt
  - a program kiírja, hogy érvénytelen a bemenő adat

Specifikáció

- Előfeltétel: milyen körülmények között követelünk helyes működést
- Utófeltétel: mit várunk a kimenettől, mi az összefüggés a kimenet és a bemenet között
- Ezek **feltételek**, tehát vagy teljesülnek, vagy nem. Ha teljesülnek, akkor a program megoldja a feladatot.
- A specifikáció feltételekből áll, nem utasításokból, mert a feladatot írja le, és nem a programot.

Specifikáció

- BE: a:nemnegatív valós
  KI: b:nemnegatív valós, b*b=a
- BE: a: valós
  KI: ha a>=0: b nemnegatív valós, b*b=a
- BE: a:valós
  KI: ha a>=0: b nemnegatív valós, b*b=a
  ha a < 0: „érvénytelen bemenet”

Specifikáció

- BE: a:nemnegatív valós
  KI: b:nemnegatív valós, b*b=a
- BE: a: valós
  KI: ha a>=0: b
- BE: a:valós
  KI: ha a>=0: b
  ha a < 0: „

```
PROGRAM szekvencia
  VÁLTOZÓK:
    a, b: VALÓS

  BE: a
  b := a ^ 0.5
  KI: b
PROGRAM_VÉGE
```

Specifikáció

- BE: a:nemnegatív valós
  KI: b:nemnegatí
- BE: a: valós
  KI: ha a>=0: b n
- BE: a:valós
  KI: ha a>=0: b n
  ha a < 0: „ér

```
PROGRAM szekvencia
  VÁLTOZÓK:
    a, b: VALÓS

  BE: a
  HA a >= 0 AKKOR
    b := a ^ 0.5
    KI: b
  HA_VÉGE
PROGRAM_VÉGE
```

Specifikáció

- BE: a:nemnegat
  KI: b:nemnegat
- BE: a: valós
  KI: ha a>=0: b r
- BE: a:valós
  KI: ha a>=0: b r
  ha a < 0: „ér

```
PROGRAM szekvencia
  VÁLTOZÓK:
    a, b: VALÓS

  BE: a
  HA a >= 0 AKKOR
    b := a ^ 0.5
    KI: b
  KÜLÖNBEN
    KI: "érvénytelen
adat"
  HA_VÉGE
PROGRAM_VÉGE
```

Összeadó program v1.0

```
PROGRAM összeadó1
  VÁLTOZÓK:
    a, b: EGÉSZ

  BE: a, b
  KI: a + b
PROGRAM_VÉGE
```

Összeadó program v1.1

```
PROGRAM összeadó2
  VÁLTOZÓK:
    a, b, c: EGÉSZ

  BE: a, b, c
  KI: a + b + c
PROGRAM_VÉGE
```

Sok számot hogyan adunk össze?

- Alapvető, hogy a programszöveg nem függhet az adatoktól, tehát azt nem lehet csinálni, hogy legyen annyi a+b+c+d... amennyi kell. Más megoldás után kell nézni
- Megpróbálhatnánk azt, hogy megismételtetünk műveletet, ahányszor kell
- Ehhez ismétlődő szakaszokat kell keresni/csinálni a programban

Sok számot hogyan adunk össze?

- Tételezzük fel, hogy a sok számot úgy tudjuk beolvasni, hogy az első szám a sorozat hossza
- A módszer tehát olyasmi lesz, hogy
  BE: számokszáma
  Ismételd meg az összeg növelését a következő sorozatelemmel
  számokszáma **alkalommal**
- Lássuk, hogyan lehet ezt megoldani

Ciklus

…

CIKLUS AMÍG *logikai kifejezés*

programrészlet

CIKLUS_VÉGE

…

Összeadó program v2.0

```
PROGRAM összeadó3
  VÁLTOZÓK:
    a, b, összeg: EGÉSZ

  BE: a, b
  összeg := a + b
  KI: összeg
PROGRAM_VÉGE
```

Összeadó program v2.1

```
PROGRAM összeadó4
  VÁLTOZÓK:
    a, b, összeg: EGÉSZ

  BE: a
  összeg := a
  BE: b
  összeg := összeg + b
  KI: összeg
PROGRAM_VÉGE
```

Összeadó program v2.1

```
PROGRAM összeadó5
  VÁLTOZÓK:
    a, összeg: EGÉSZ

  BE: a
  összeg := a
  BE: a
  összeg := összeg + a
  KI: összeg
PROGRAM_VÉGE
```

Összeadó program v2.1

```
PROGRAM összeadó6
  VÁLTOZÓK:
    a, összeg: EGÉSZ

  összeg := 0
  BE: a
  összeg := összeg + a
  BE: a
  összeg := összeg + a
  KI: összeg
PROGRAM_VÉGE
```

Összeadó program v2.1

```
PROGRAM összeadó6
  VÁLTOZÓK:
    a, összeg: EGÉSZ

  összeg := 0
  BE: a
  összeg := összeg + a
  BE: a
  összeg := összeg + a
  KI: összeg
PROGRAM_VÉGE
```

Ismétlés amit annyiszor kell csinálni ahány adat van

Összeadó program v3.0

```
PROGRAM sorozatösszeadó
  VÁLTOZÓK:
    n, a, összeg, i: EGÉSZ

  BE: n
  i := 0
  összeg := 0
  CIKLUS AMÍG i < n
    BE: a
    összeg := összeg + a
    i := i + 1
  CIKLUS_VÉGE
  KI: összeg
PROGRAM_VÉGE
```

Összeadó programok összevetése

```
PROGRAM összeadó6
  VÁLTOZÓK:
    a, összeg: EG

  összeg := 0
  BE: a
  összeg := össze
  BE: a
  összeg := össze
  KI: összeg
PROGRAM_VÉGE
```

```
PROGRAM sorozatösszeadó
  VÁLTOZÓK:
    n, a, összeg, i: EGÉSZ

  BE: n
  i := 0
  összeg := 0
  CIKLUS AMÍG i < n
    BE: a
    összeg := összeg + a
    i := i + 1
  CIKLUS_VÉGE
  KI: összeg
PROGRAM_VÉGE
```

Ciklus

- A ciklus az ismétlés lehetősége, egy programrészletet addig ismételünk meg, amíg egy adott feltétel (ciklusfeltétel) teljesül, és befejezzük, ha hamissá válik
  - Egyes nyelvek az „amíg nem” feltételt (angolul „until”) használják
- Fontos, hogy a feltétel előbb vagy utóbb igazzá váljon, különben a ciklus végtelen sokáig ismételné az adott programrészt, és ez ritkán feladat („végtelenciklus”)

Ciklus: fogalmak

```
PROGRAM ...
  ...
   CIKLUS AMÍG ciklusfeltétel

      ciklusmag

   CIKLUS_VÉGE
  ...
PROGRAM_VÉGE
```

Ciklus

- A ciklus magja egy tetszőleges önmagában teljes programrészlet
- A ciklusfeltétel egy logikai típusú kifejezés
- A ciklusfeltétel kiértékelése újra és újra megtörténik, lefutásonként
- Ha az eredmény hamis, a ciklus végetér, és a „ciklus_vége” sor után folytatódik
- Előfordulhat, hogy a kifejezés azonnal hamis, a program ilyenkor kihagyja a ciklusmag lépéseit

Ciklus

- Elvi tudnivalók a ciklusokról:
  - ha nem lenne ciklus (vagy annak megfelelő más eszköz) a program csak annyi lépésből állhatna, ahány sora van, ez nyilvánvalóan szűkös lenne.
  - ha a ciklusfeltétel kifejezésében nincs változó, vagy egyik változó sem szerepel a ciklusmagban értékadás baloldalán, az gyanús.
  - Érdemes mindig végiggondolni a változók jelentését a ciklusokban

Összeadó program v3.0

```
PROGRAM sorozatösszeadó
  VÁLTOZÓK:
    n, a, összeg, i: EGÉSZ

  BE: n
  i := 0
  összeg := 0
  CIKLUS AMÍG i < n
    BE: a
    összeg := összeg + a
    i := i + 1
  CIKLUS_VÉGE
  KI: összeg
PROGRAM_VÉGE
```

n: ennyi elem van a sorozatban

a: ennél az elemnél tartunk

i: az ennyiedik elemnél tartunk

összeg: az eddig látott elemek összege

Ciklus összefoglalás

- Akkor használjuk, ha valamit többször kell végrehajtani
- Ki kell mindig találni, hogy milyen kezdeti értékekkel, milyen ciklusfeltétellel, és milyen ciklusmaggal oldható meg a feladat
  - Egy nagyon gyakori eset, hogy ciklusváltozót használunk, és abban számoljuk, hogy hányszor futott le a ciklusmag
- Fontos, hogy mindig tudjuk, melyik változónk mit jelent

Programkonstrukciók

- Elemi programok: értékadás, KI, BE, …
- Minden elemi program helyes program
- Minden helyes program kombinálható:
  - szekvencia
  - elágazás
  - ciklus
- Az eredmény: a program egy hierarchikus szerkezete a fenti programkonstrukcióknak, és végső soron elemi programok kombinációja

Programkonstrukciók összefoglalás

- Elágazás kell, ha más kódra van szükség egyes esetekben
- Ciklus kell, ha ismételni kell lépéseket
- Okos megbízható matematikusok eredménye: Minden algoritmikusan megoldható probléma megoldható szekvencia, elágazás és ciklus segítségével.
  - Tehát vége is a féléves anyagnak...