
# Bevezetés a programozásba

3. Előadás
Algoritmusok, tételek

ISMÉTLÉS

# 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.

ISMÉTLÉS

# Elágazás

```
PROGRAM elágazás
  VÁLTOZÓK:
    a: EGÉSZ

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

ISMÉTLÉS

# Ciklus

```
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
```

ISMÉTLÉS

# 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...~~
  - Érdemes lesz azért bejárni..

# Algoritmus

- Sokféle definíció van, praktikusan elágazásból, ciklusból, értékadásból és I/O-ból álló helyes program elve, ami egy adott feladatot megold
  - A fogalom ennél absztraktabb
- Néhány algoritmus általános iskolai anyag, mint a „papíron szorzás”
- Néhány száz algoritmusnak neve is van
- Mire jó ismerni algoritmusokat: ha valaki már megoldott egy problémát, nekünk már ne kelljen

# Program - algoritmus

- A program egy konkrét feladatot, konkrét formátumban adott kimenettel és bemenettel old meg
- Az algoritmus a megoldás elve, formátumtól, típusoktól amennyire lehet, függetlenül
- A továbbiakban programokat és az azokból absztrakcióval kapott algoritmusokat fogtok látni

# Sorozat

- A mai előadáson minden sorozat formátuma a következő:
  - Az első bemenet egy szám, ami a következő sorozat hosszát jelenti
  - Utána jönnek a sorozatelemek

# Sorozat

- A mai előadás
  következő:
  - Az első beme
    sorozat hoss
  - Utána jönnek

A „minden elemet lemásoló” program

```
PROGRAM elemenként
  VÁLTOZÓK:
    n, i: EGÉSZ,
    a: VALÓS

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

# Összegzés

```
PROGRAM sorozatösszeadó
  VÁLTOZÓK:
    n, i: EGÉSZ,
    a, összeg: VALÓS
  BE: n
  i := 0
  összeg := 0
  CIKLUS AMÍG i < n
    BE: a
    összeg := összeg + a
    i := i + 1
  CIKLUS_VÉGE
  KI: összeg
PROGRAM_VÉGE
```

**összeg**: az eddig látott adatok összege

Minden sorozat-elem pontosan egy alkalommal szerepel **a**-ban

# Összegzés tétele

```
Változók: összeg, a : T

  összeg := 0
  CIKLUS AMÍG nincs vége a sorozatnak
    a := következő elem
    összeg := összeg ⊕ f(a)
  CIKLUS_VÉGE
```

# Összegzés tétele

- Mire jó
  - Kumulatív (halmozó) feladatok sorozatokon
- Mi cserélhető
  - Kezdőérték
  - Művelet, függvény
- Példák
  - Átlag
  - Faktoriális
  - Négyzetösszeg

# Számlálás

```
PROGRAM számlálás
  VÁLTOZÓK:
    i, n, sz, a: EGÉSZ

  BE: n
  sz := 0
  i := 0
  CIKLUS AMÍG i < n
    BE: a
    HA a MOD 2 = 0 AKKOR
      sz := sz + 1
    HA_VÉGE
    i := i + 1
  CIKLUS_VÉGE
  KI: sz
PROGRAM_VÉGE
```

sz: számláló, az eddig látott megfelelőek száma

# Számlálás tétele

```
VÁLTOZÓK: sz: EGÉSZ, a:T

  sz := 0
  CIKLUS AMÍG nincs vége a sorozatnak
    a:= következő elem
    HA feltétel(a) AKKOR
      sz := sz + 1
    HA_VÉGE
  CIKLUS_VÉGE
```

# Számlálás tétele

- Mire jó
  - „mennyi?” „hány?” feladatok sorozatokon
- Mi cserélhető
  - Feltétel
  - Növelő függvény
- Példák
  - Osztók száma
  - Szavak száma egy szövegben

# Lineáris keresés

```
PROGRAM lineáris_keresés
  VÁLTOZÓK:
    i, n, a, hol: EGÉSZ,
    van: LOGIKAI

  BE: n
  van := HAMIS
  hol := 0
  i := 0
  CIKLUS AMÍG i < n ÉS NEM van
    BE: a
    i := i + 1
    HA a MOD 2 = 0 AKKOR
      van := IGAZ
      hol := i
    HA_VÉGE
  CIKLUS_VÉGE
  KI: van
  HA van AKKOR
    KI: hol
  HA_VÉGE
PROGRAM_VÉGE
```

Két eredmény van: van-e megfelelő, és (ha igen) hol

van: van-e az eddig látott elemek között megfelelő?

hol: a megfelelő elem indexe

(ha nincs megfelelő, nem érdekes, mi az értéke)

# Lineáris keresés tétele

```
VÁLTOZÓK:
  hol,i: EGÉSZ, van: LOGIKAI, a: T

van := HAMIS
hol := 0
i := 0
CIKLUS AMÍG nincs vége a sorozatnak ÉS NEM van
  a := következő elem
  i := i + 1
  HA feltétel(a) AKKOR
    van := IGAZ
    hol := i
  HA_VÉGE
CIKLUS_VÉGE
```

# Lineáris keresés tétele

- Mire jó
  - „van-e?”, „hányadik?” feladatok sorozatokon
- Mi cserélhető
  - Feltétel
- Példák
  - Prím-e egy szám (van-e osztója?)
  - Szerepel-e egy bizonyos érték egy sorozatban

# Maximum keresés

```
PROGRAM maximumkeresés
  VÁLTOZÓK:
    i, n, hol: EGÉSZ,
    a, max: VALÓS

  BE: n
  i := 1
  BE: a
  max := a
  hol := 1
  CIKLUS AMÍG i < n
    BE: a
    i := i + 1
    HA max < a AKKOR
      max := a
      hol := i
    HA_VÉGE
  CIKLUS_VÉGE
  KI: max, SV, hol
PROGRAM_VÉGE
```

Két eredmény: mennyi a maximum, és melyik elem volt az. Mindig létezik, ha legalább egy elemű a sorozat

A max és hol az eddig látott maximális elemre vonatkoznak, ez az elején az első elem → i:=1

Ha találunk az eddigi maximumnál nagyobbat akkor lecseréljük

# Maximum keresés tétele

```
VÁLTOZÓK:
  i, hol: EGÉSZ,
  a, max: T

i := 1
a := első elem
max := f(a)
hol := 1
CIKLUS AMÍG nincs vége a sorozatnak
  a := következő elem
  i := i + 1
  HA max < f(a) AKKOR
    max := f(a)
    hol := i
  HA_VÉGE
CIKLUS_VÉGE
```

# Maximum keresés tétele

- Mire jó
  - „Mi / Mennyi a leg..?” feladatok sorozatokon
- Mi cserélhető
  - Feltétel, reláció
  - Függvény (másik változó akár, aminek az értéke a bemenettől függ)
- Példák
  - Minimumkeresés
  - Zárójelek mélysége szövegben

# Elemenként feldolgozás

- Sok adat, kevés változó
- Egy feladat elemenként feldolgozható, ha
  - meg lehet úgy oldani, hogy a hosszú adatsorból egyszerre csak kevésre van szükség
  - csak egyszer kell végignézni mindegyiket
- Sok hasznos algoritmus tartozik ide
  - Az előbb látottak
  - Válogatás (pl. sorozatból a párosakat írjuk ki)
  - Összefésülés

# Elemenként feldolgozás

- Példa egyéb elemenként feldolgozható feladatra:
  - Sorozat értékkészletét befoglaló intervalluma
  - Sorozat második legnagyobb értéke
  - Van-e még egy olyan ami megegyezik az első elemmel
- Példa feladatra, ami **nem** elemenként feldolgozható:
  - Sorozat növekvő/csökkenő sorrendbe rendezése
  - Sorozat mediánja
  - Van-e két egyforma

# Tételek kombinációi

- Feladat: egy intervallumban számoljuk meg a prímszámokat
- Számlálás tétel kell hozzá, de még nem elég: nincs „prím-e” műveletünk
- Ezért a prímdöntést lineáris kereséssel (van-e valódi osztója?) dönthetjük el a számlálás belsejében
- → „állapottér transzformáció”, úgy teszünk a számlálásban, mintha logikai értékeket olvasnánk, amiket egy másik tétellel állítunk elő

# prímek száma intervallumon

```
VÁLTOZÓK:
  a,sz,ieleje, ivége: EGÉSZ

  BE: ieleje, ivége
  a:=ieleje
  sz := 0
  CIKLUS AMÍG a<=ivége
     HA prím(a) AKKOR
        sz := sz + 1
     HA_VÉGE
     a:=a+1
  CIKLUS_VÉGE
  KI:sz
```

# prímek száma intervallumon

```
VÁLTOZÓK:
  a,sz,iel

  BE: iele
  a:=ielej
  sz := 0
  CIKLUS AM
    HA prím
      sz :
    HA_VÉG
    a:=a+1
  CIKLUS_V
  KI:sz
```

```
PROGRAM van_e_valódi_osztó
  VÁLTOZÓK:
    i, n: EGÉSZ, van: LOGIKAI

  BE: n
  van := HAMIS
  i := 2
  CIKLUS AMÍG i < n ÉS NEM van
    HA n MOD i = 0 AKKOR
      van := IGAZ
    HA_VÉGE
    i := i + 1
  CIKLUS_VÉGE
  KI: van
PROGRAM_VÉGE
```

```
PROGRAM tétel_kombináció
  VÁLTOZÓK:
    a, sz, ieleje, ivége, i: EGÉSZ,
    van: LOGIKAI

  BE: ieleje, ivége
  a := ieleje
  sz := 0
  CIKLUS AMÍG a <= ivége
    van := HAMIS
    i := 2
    CIKLUS AMÍG i < a ÉS NEM van
      HA a MOD i = 0 AKKOR
        van := IGAZ
      HA_VÉGE
      i := i + 1
    CIKLUS_VÉGE
    HA NEM van AKKOR
      sz := sz + 1
    HA_VÉGE
    a := a + 1
  CIKLUS_VÉGE
  KI: sz
PROGRAM_VÉGE
```

```
VÁLTOZÓK:
    a,sz,ieleje, ivége: EGÉSZ


    BE: ieleje, ivége
    a:=ieleje
    sz := 0
    CIKLUS AMÍG a<=ivége
        HA prím(a) AKKOR
            sz := sz + 1
        HA_VÉGE
        a:=a+1
    CIKLUS_VÉGE
    KI:sz
```

```
OGRAM van_e_valódi_osztó
VÁLTOZÓK:
  i, n: EGÉSZ, van: LOGIKAI

BE: n
van := HAMIS
i := 2
CIKLUS AMÍG i < n ÉS NEM van
  HA n MOD i = 0 AKKOR
    van := IGAZ
  HA_VÉGE
  i := i + 1
CIKLUS_VÉGE
KI: van
OGRAM_VÉGE
```