Programozási tételek

A tételekről általánosságban

> *Előre is elnézést kérek a probléma szó ismételgetéséért!* ☺

A programozási tételek olyan algoritmusminták, melyek egy adott típusprobléma megoldását adják meg. Sokszor más megoldás is lehetséges, de ezek a tételek bizonyítottak, és legtöbbször a lehető legegyszerűbb megoldását adják az adott kérdéskörnek.

A tételeket általában nem „egy az egyben”, hanem módosításokkal, sokszor jelentős módosításokkal, használjuk fel egy programozási feladat megvalósításakor, tehát nem a szószerinti bemagolásuk a lényeg, hanem a logikájuk és az, hogy milyen problémákhoz használhatók.

Egy programozási feladat szövegének elemzése során ismerjük fel, hogy milyen típusú problémákkal állunk szemben. Ha képesek vagyunk felismerni ezeket a problémaköröket, akkor már ki tudjuk választani a megfelelő algoritmusmintát, azaz programozási tételt a megoldáshoz, és annak logikája mentén specializálni az adott tételt a konkrét feladathoz. Így nagyon nagy szerepe van a szövegértésnek! (Mint minden érettségi feladatban, és általában a nagybetűs életben!)

Természetesen ezek a tételek az egészen egyszerűtől az igazán bonyolultig terjednek, és sok olyan probléma is lehet, amikor nem ismerünk semmilyen algoritmusmintát, amit „bevethetnénk”, de ha kombináljuk, összerakjuk a továbbiakban tárgyalandó legalapvetőbb tételeket, akkor nagy valószínűséggel képesek leszünk megoldani az érettségi feladatokat. Remélem!

Programozási tételek csoportosítása

A programozási tételeket olyan szempontból is elemezhetjük, hogy milyen típusú, szerkezetű adatot ad vissza az adott algoritmus minta. Ilyen szempontból:

- **sorozat → érték** (pl. összegzés, kiválasztás, lineáris keresés)
- **sorozat → sorozatok** (pl. kiválogatás, szétválogatás)
- **sorozatok → sorozat** (pl. metszet)

típusú tételekről beszélhetünk. (De lehet beszélni elemi és összetett tételekről is.)

Elemi tételek (sorozat → érték)	Összetett tételek (sorozat(ok) → sorozat(ok))
sorozatszámítás	másolás
eldöntés	kiválogatás
kiválasztás	szétválogatás
lineáris keresés	metszet
megszámolás	únió
maximumkiválasztás	összefuttatás

Programozási tételek leírása

A tételek adatsorozatokat (tömböket, vektorokat – ez nyelvfüggő) dolgoznak fel, ezekkel kapcsolatos típusfeladatokra adnak megoldást.

ELEMI PROGRAMOZÁSI TÉTELEK

SOROZATSZÁMÍTÁS (ÖSSZEGZÉS, PRODUKTUM…) TÉTELE

***sorozat → érték** típusú, azaz **elemi** tétel*

Adott egy **N** elemű **X** sorozat. Az elemeken értelmezhető az **f** művelet, és $\mathbf{f_0}$, a művelethez kapcsolódó *nullelem* (vagy neutrális elem). Adjuk meg az elemek alapján **f eredményét** (pl. az elemek összegét)!

(**f** művelet lehet pl.: **+**, ***…, ezekhez kapcsolódóan $\mathbf{f_0}$: 0, 1…)

Algoritmus mondatszerű leírása:

```
Sorozatszámítás(N, X, S):
  S:=F0
  Ciklus i=1-től N-ig
    S:=f(S,X[i])
  Ciklus vége
Eljárás vége.
```

ELDÖNTÉS TÉTELE

***sorozat → érték** típusú, azaz **elemi** tétel*

1. Adott egy **N** elemű **X** sorozat és egy a sorozat elemein értelmezhető **T** tulajdonság. **T(X[i])** ***igaz***, ha **X[i] T** tulajdonságú, és ***hamis*** ellenkező esetben. Döntsük el, **van-e** a sorozatban **T** tulajdonságú elem!

Algoritmus mondatszerű leírása:

```
Eldöntés(N, X, VAN):
  i:=1
  Ciklus amíg i≤N és nem T(X[i])
    i:=i+1
  Ciklus vége
  VAN:=(i≤N)
Eljárás vége.
```

2. Adott egy **N** elemű **X** sorozat és egy a sorozat elemein értelmezhető **T** tulajdonság. **T(X[i])** ***igaz***, ha **X[i] T** tulajdonságú, és ***hamis*** ellenkező esetben. Döntsük el, hogy **minden elem T** tulajdonságú-e!

Algoritmus mondatszerű leírása:

```
Eldöntés(N, X, MIND):
  i:=1
  Ciklus amíg i≤N és T(X[i])
    i:=i+1
  Ciklus vége
  MIND:=(i>N)
Eljárás vége.
```

KIVÁLASZTÁS TÉTELE

***sorozat → érték** típusú, azaz **elemi** tétel*

Adott egy **N** elemű **X** sorozat és egy a sorozat elemein értelmezhető **T** tulajdonság. **T(X[i])** ***igaz***, ha **X[i] T** tulajdonságú, és ***hamis*** ellenkező esetben.

Biztosan van (legalább egy) **T** tulajdonságú elem a sorozatban. **Adjuk meg** az egyik **T** tulajdonságú **elemet**! (Jobb, ha a sorszámát adjuk meg…)

Algoritmus mondatszerű leírása:

```
Kiválasztás(N, X, SORSZAM):
  i:=1
  Ciklus amíg nem T(X[i])
    i:=i+1
  Ciklus vége
  SORSZAM:=i
Eljárás vége.
```

LINEÁRIS KERESÉS TÉTELE

***sorozat → érték** típusú, azaz **elemi** tétel*

Adott egy **N** elemű **X** sorozat és egy a sorozat elemein értelmezhető **T** tulajdonság. **T(X[i])** ***igaz***, ha **X[i] T** tulajdonságú, és ***hamis*** ellenkező esetben. **Döntsük el**, **van-e** a sorozatban **T** tulajdonságú elem és **ha van, adjuk meg** (sorszámát vagy értékét)!

Ez a tétel az előző két tétel „összeépítése” (Eldöntés + Kiválasztás → Lineáris keresés)

Algoritmus mondatszerű leírása:

```
Keresés(N, X, VAN, SORSZAM):
  i:=1
  Ciklus amíg i≤N és nem T(X[i])
    i:=i+1
  Ciklus vége
  VAN:=(i≤N)
  Ha VAN akkor SORSZAM=i
Eljárás vége.
```

MEGSZÁMOLÁS TÉTELE

***sorozat → érték** típusú, azaz **elemi** tétel*

Adott egy **N** elemű **X** sorozat és egy a sorozat elemein értelmezhető **T** tulajdonság. **T(X[i])** ***igaz***, ha **X[i] T** tulajdonságú, és ***hamis*** ellenkező esetben. Határozzuk meg a **T** tulajdonságú **elemek számát**!

Algoritmus mondatszerű leírása:

```
Megszámolás(N,X,db):
  db:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor db:=db+1
  Ciklus vége
Eljárás vége.
```

MAXIMUMKIVÁLASZTÁS TÉTELE

***sorozat → érték** típusú, azaz **elemi** tétel*

Adott egy **N** elemű **X** sorozat és *a sorozat elemein értelmezhető a <, >, = relációk*. Határozzuk meg a sorozat **maximális értékű** elemét!

Algoritmus mondatszerű leírása:

```
Maximumkiválasztás(N,X,max):
  max:=1
  Ciklus i=2-től N-ig
    Ha X[max]<X[i] akkor max:=i
  Ciklus vége
Eljárás vége
```

Ha a *relációt megfordítjuk*, akkor a minimális értékű elem kiválasztása lesz az algoritmus eredménye.

ÖSSZETETT PROGRAMOZÁSI TÉTELEK

MÁSOLÁS TÉTELE

***sorozat → sorozat** típusú, azaz **összetett** tétel*

Adott egy **N** elemű **X** sorozat és egy a sorozat elemein értelmezhető **f** művelet, függvény. Állítsunk elő egy **új sorozatot**, melybe „bemásoljuk” az **eredeti sorozat f függvénnyel átalakított elemeit**!

X[i] → f(X[i]) X i. eleméből Y i. eleme áll elő…

(***f** akármi lehet, akár semmi is…*)

Algoritmus mondatszerű leírása:

```
Másolás(N,X,Y):
  Ciklus i=1-től N-ig
    Y[i]:=f(X[i])
  Ciklus vége
Eljárás vége.
```

KIVÁLOGATÁS TÉTELE

***sorozat → sorozat** típusú, azaz **összetett** tétel*

Adott egy **N** elemű **X** sorozat és egy a sorozat elemein értelmezhető **T** tulajdonság. **T(X[i])** ***igaz***, ha **X[i] T** tulajdonságú, és ***hamis*** ellenkező esetben. Válogassuk ki a **T** tulajdonságú **elemeket**

a. egy **új sorozatba**!
b. **„helyben”**, azaz az eredeti sorozatban!

Algoritmus mondatszerű leírása:

a.

```
Kiválogatás(N,X,Y):
  j:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor j:=j+1; Y[j]:=X[i]
  Ciklus vége
Eljárás vége.
```

b.

```
Kiválogatás(N,X,j):
  j:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor j:=j+1; X[j]:=X[i]
  Ciklus vége
Eljárás vége.
```

SZÉTVÁLOGATÁS TÉTELE

***sorozat → sorozatok** típusú, azaz **összetett** tétel*

Adott egy **N** elemű **X** sorozat és egy a sorozat elemein értelmezhető **T** tulajdonság. **T(X[i])** ***igaz***, ha **X[i] T** tulajdonságú, és ***hamis*** ellenkező esetben. Válogassuk szét a **T** tulajdonságú és nem **T** tulajdonágú **elemeket**

a. **két új sorozatba**,
b. **egy új sorozatba**,
c. **„helyben”**, azaz az eredeti sorozatban!

Algoritmus mondatszerű leírása:

a.

```
Szétválogatás(N,X,Y,Z,dbY,dbZ):
  dbY:=0
  dbZ:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor dbY:=dbY+1;Y[dbY]:=X[i]
               különben dbZ:=dbZ+1;Z[dbZ]:=X[i]
  Ciklus vége
Eljárás vége.
```

b. (T tulajdonságú elemek az *új sorozat* elejére, nem T tulajdonságú elemek a végére kerülnek)

```
Szétválogatás(N,X,Y,dbY,indexZ):
  dbY:=0
  indexZ:=N+1
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor dbY:=dbY+1;Y[dbY]:=X[i]
               különben indexZ:=indexZ-1;Y[indexZ]:=X[i]
  Ciklus vége
Eljárás vége.
```

c. (T tulajdonságú elemek az *eredeti* sorozat elejére, nem T tulajdonságú elemek a végére kerülnek)

```
Szétválogatás(N,X,dbT):
  e:=1; v:=N; temp:=X[e]
  Ciklus amíg e<v
    // T tulajdonságú elem keresése
    Ciklus amíg e<v és nem T(X[v])
      v:=v-1
    Ciklus vége
    Ha e<v akkor
      X[e]:=X[v]; e:=e+1
      // nem T tulajdonságú elem keresése
      Ciklus amíg e<v és T(X[e])
        e:=e+1
      Ciklus vége
      Ha E<V akkor X[v]:=X[e]; v:=v-1
    Elágazás vége
  Ciklus vége
  X[e]:=temp
  Ha T(X[e]) akkor dbT:=e különben dbT:=e-1
Eljárás vége.
```

METSZET TÉTELE

***sorozatok → sorozat** típusú, azaz **összetett** tétel*

Adott egy **N** elemű **X** sorozat és egy **M** elemű **Y** sorozat. Határozzuk meg **X és Y közös elemeit** egy új sorozatban! (Természetesen kettőnél több sorozatra is kiterjeszthető a minta…)

Algoritmus mondatszerű leírása:

```
Metszet(N,X,M,Y,db,Z):
  db:=0
  Ciklus i=1-től N-ig
    j:=1
    Ciklus amíg j≤M és X[i]≠Y[j]
      j:=j+1
    Ciklus vége
    Ha j≤M akkor db:=db+1; Z[db]:=X[i]
  Ciklus vége
Eljárás vége.
```

Itt észrevehető, hogy **tételek egybeépítése** történt, azaz olyan *kiválogatást* hajtunk végre, amely *eldöntésen* alapul:

Kiválogatjuk **X** sorozat elemeit a **Z**-be, eldöntve, hogy *benne vannak-e* **Y**-*ban is*.

Ha nincs közös, metszetbeli elem, akkor **Z** üres marad.

ÚNIÓ (EGYESÍTÉS) TÉTELE

***sorozatok → sorozat** típusú, azaz **összetett** tétel*

Adott egy **N** elemű **X** sorozat és egy **M** elemű **Y** sorozat. Határozzuk meg **X és Y únió jának elemeit** egy új sorozatban! (Természetesen kettőnél több sorozatra is kiterjeszthető a minta…)

A *közös elemeket*, természetesen, *csak egyszer* vegyük figyelembe.

Algoritmus mondatszerű leírása:

```
Únió(N,X,M,Y,db,U):
  U:=X; db:=N
  Ciklus j=1-től M-ig
    i:=1
    Ciklus amíg i≤N és X[i]≠Y[j]
      i:=i+1
    Ciklus vége
    Ha i>N akkor db:=db+1; U[db]:=Y[j]
  Ciklus vége
Eljárás vége.
```

Itt is észrevehető, hogy **tételek egybeépítése** történt, azaz egyszer *másolást* hajtunk végre, majd olyan *kiválogatást*, amely *eldöntésen* alapul:

Átmásoljuk **X** elemeit **U**-ba, majd kiválogatjuk **Y** sorozat elemeit a **U**-ba, eldöntve, hogy ne legyenek *benne* **X**-*ben is*.

ÖSSZEFUTTATÁS TÉTELE

***sorozatok → sorozat** típusú, azaz **összetett** tétel*

Adott egy **N** elemű **rendezett X** sorozat és egy **M** elemű **rendezett Y** sorozat. Állítsuk elő a sorozatok rendezett unióját! (Természetesen kettőnél több sorozatra is kiterjeszthető a minta…)

A *közös elemeket*, természetesen, *csak egyszer* vegyük figyelembe.

Az *únió tételéhez* hasonló a típusfeladat, de *kihasználhatjuk a sorozatok rendezettségét*, valamint *az eredménysorozatnak is rendezettnek kell le*nnie!

Algoritmus mondatszerű leírása:

```
Összefuttatás(N,X,M,Y,Z):
  i:=1; j:=1; indZ:=0
  Ciklus amíg i≤N és j≤M
    indZ:=indZ+1
    Elágazás
      X[i]<Y[j] esetén Z[indZ]:= X[i]; i:=i+1
      X[i]=Y[j] esetén Z[indZ]:= X[i]; i:=i+1; j:=j+1
      X[i]>Y[j] esetén Z[indZ]:=Y[j]; j:=j+1
    Elágazás vége
  Ciklus vége
  Ciklus amíg i≤N
    indZ:=indZ+1
    Z[indZ]:=X[i]
    i:=i+1
  Ciklus vége
  Ciklus amíg j≤M
    indZ:=indZ+1
    Z[indZ]:=Y[j]
    j:=j+1
  Ciklus vége
Eljárás vége.
```