Elemi algoritmusok

Olyan algoritmusokat tárgyalunk az alábbiakban, amelyek sok probléma megoldási algoritmusában mint részprobléma megoldási algoritmusok fordulnak elő. Ezek az algoritmusok feladatosztályt oldanak meg, így részint absztrakt formában kerülnek megfogalmazásra. Jellemzőjük lesz, hogy inputjuk egy vagy több sorozat, outputjuk pedig egy érték, egy sorozat vagy több sorozat. Ezt egy táblázattal szemléltethetjük:

Input		Output
Egy sorozat	—	Egy érték
Egy sorozat	—	Egy sorozat
Egy sorozat	—	Több sorozat
Több sorozat	—	Egy sorozat

Az alábbi algoritmusok kerülnek ismertetésre:

Sorszám	Algoritmus neve
I.	Sorozatszámítás
II.	Eldöntés
III.	Kiválasztás
IV.	Lineáris keresés
V.	Megszámlálás
VI.	Maximum kiválasztás

Sorszám	Algoritmus neve
VII.	Másolás
VIII.	Kiválogatás
IX.	Szétválogatás
X.	Metszet
XI.	Egyesítés
XII.	Összefuttatás

Az algoritmusok inputja egy vagy több sorozat lesz, outputja érték vagy egy sorozat vagy több sorozat. A sorozat elemei valamely absztrakt adattípushoz tartoznak. Formálisan, ha *X* sorozat, akkor az elemeit indexelt *x* betűvel fogjuk jelölni. Az x_i elemek valamely ***A*** absztrakt adattípusnak felelnek meg (például valós számok esetén $\boldsymbol{A}=\boldsymbol{R}$ és $x_i \in \boldsymbol{A}$ azt jelenti, hogy $x_i \in \boldsymbol{R}$) Az *X* sorozathoz mindenképpen tartozik egy Hossz[*X*] attribútum, amely a sorozat éppen aktuális elemszámát adja meg, valamint tartozik egy Méret[*X*] attribútum, amely a sorozat részére lefoglalt memória terület nagyságát adja meg elemméret egységgel számolva.

Többször előfordul majd egy szimbolikusan □ jellel jelölt bináris művelet (binér művelet) valamely ***A*** halmazon. (Például ha $\boldsymbol{A}=\boldsymbol{R}$ és □=+, akkor az $x \,\square\, y = x + y$, azaz a bináris művelet a valós számok halmazán az összeadás, amely valós számpárhoz egy valós számot rendel.) Gyakran van a bináris műveletnek úgynevezett *zérus eleme(nullelem)*. A zérus elem olyan elem, amely esetén a művelet a hozzá párba vett másik operandust adja. A nullelem jelölése Null, Null $\in \boldsymbol{A}$ és Null$\square a = a$, $\forall\ a \in \boldsymbol{A}$ (*baloldali nullelem*). Egyes algoritmusokban megköveteljük, hogy az ***A*** absztrakt adattípusban a halmaz teljesen rendezett halmazt alkosson. Ez azt jelenti, hogy tetszőleges két eleme ennek a halmaznak összehasonlítható legyen egymással és az összehasonlítás *rendezési reláció* legyen. Lesznek olyan algoritmusok, amelyekben az elemeknek valamely T tulajdonságát kell vizsgálni. A T tulajdonság egy függvényt jelent, amely az elemek absztrakt adattípusa halmazán van értelmezve és ezen halmaz minden eleméhez hozzárendel egy logikai értéket, azaz a halmazt a logikai értékek $\boldsymbol{L}=\{hamis, igaz\}$ halmazába képezi bele:
T: $\boldsymbol{A} \rightarrow \boldsymbol{L}$.

$$T(a) = \begin{cases} hamis, & ha\ \ a \in A\ \ nem\ T\ \ tulajdonságú \\ igaz, & ha\ \ a \in A\ \ T\ \ tulajdonságú \end{cases}$$

Az algoritmusokban az **INC** utasítás eggyel növeli, a **DEC** utasítás eggyel csökkenti a paramétere, operandusa számértékét. Feltételezzük továbbá, hogy a logikai kifejezések kiértékelésekor *optimalizált kiértékelés*t végzünk, azaz, ha egy ÉS művelet egyik argumentuma *hamis*, akkor a másikat már nem kell vizsgálni, mert akkor a művelet eredménye egyértelműen *hamis*. Ugyanez vonatkozik a VAGY műveletre, ahol, ha az egyik argumentum *igaz*, akkor a másik ismerete nélkül is az eredmény *igaz* lesz.

I.	**Sorozatszámítás**
	Sorozathoz rendel egy értéket, amelyet egy a sorozaton végrehajtott bináris műveleten alapuló függvény ad.

Sorozatszámítás(@X, @s)
Input paraméter: X – sorozat, $x_i \in \boldsymbol{A}$
Output paraméter: s – a kapott egyetlen érték $s \in \boldsymbol{A}$
// $\square$ egy bináris művelet (kétváltozós művelet)
// Null a $\square$ nulleleme, Null $\in \boldsymbol{A}$, Null $\square\ a = a$, $\forall\ a \in \boldsymbol{A}$
$s \leftarrow$ Null // Inicializálás
FOR $i \leftarrow 1$ **TO** Hossz[X] **DO**
$s \leftarrow s \square x_i$
RETURN (s)

Két lehetséges konkretizálása a fenti algoritmusnak következik alább.

n szám összege	**n karakterből sztring készítése**
$\boldsymbol{A}=\boldsymbol{R}$	$\boldsymbol{A}=\text{Ascii}^*$, $(\text{Ascii} \subset \text{Ascii}^*)$
$x_i \in \boldsymbol{R}$	$x_i \in \text{Ascii}$
$s \in \boldsymbol{R}$	$s \in \text{Ascii}^*$
$\square$: $\boldsymbol{R} \times \boldsymbol{R} \rightarrow \boldsymbol{R}$,	$\square$: $\text{Ascii}^* \times \text{Ascii}^* \rightarrow \text{Ascii}^*$,
$a \square b = a + b$, $a,b \in \boldsymbol{R}$	$a \square b = a \bullet b$, $a,b \in \text{Ascii}^*$ a konkatenáció művelete
Null=0, mert $0+a=a\ \ \forall\ a \in \boldsymbol{R}$	Null=ε az üres string, mert $\varepsilon \bullet a = a\ \ \forall\ a \in \text{Ascii}^*$

Összeg(@X, @s)
$s \leftarrow 0$
FOR $i \leftarrow 1$ **TO** Hossz[X] **DO**
$s \leftarrow s + x_i$
RETURN (s)

Sztringkészítés(@X, @s)
$s \leftarrow \varepsilon$
FOR $i \leftarrow 1$ **TO** Hossz[X] **DO**
$s \leftarrow s \bullet x_i$
RETURN (s)

II. Eldöntés

A bemeneti sorozathoz egy logikai értéket kell rendelni. Két alapproblémát tekintünk.
1.Van-e adott T tulajdonságú elem a sorozatban?
2. Minden eleme a sorozatnak T tulajdonságú-e?

1. alapprobléma
Eldöntés(@*X*, @*Van*)
Input paraméter: *X* – sorozat, $x_i \in \boldsymbol{A}$
Output paraméter: *Van* $\in \boldsymbol{L}$={hamis, igaz}
// T: $\boldsymbol{A} \rightarrow \boldsymbol{L}$
$i \leftarrow 1$
WHILE ($i \le$Hossz[X]) ÉS $\overline{T(x_i)}$ **DO**
INC(*i*)
Van ← ($i \le$Hossz[X])
RETURN (*Van*)

2. alapprobléma
Eldöntés(@*X*, @*Mind*)
Input paraméter: *X* – sorozat, $x_i \in \boldsymbol{A}$
Output paraméter: *Mind* $\in \boldsymbol{L}$={hamis, igaz}
// T: $\boldsymbol{A} \rightarrow \boldsymbol{L}$
$i \leftarrow 1$
WHILE ($i \le$Hossz[X]) ÉS $T(x_i)$ **DO**
INC(*i*)
Mind ← ($i >$Hossz[X])
RETURN (*Mind*)

III. Kiválasztás

Meg kell adni a sorozat egy adott T tulajdonságú elemét a sorszáma révén. Feltesszük, hogy ilyen elem van a sorozatban. (Például a kiválasztás előtt már volt egy *eldöntés*.)

Kiválasztás (@*X*, @*Sorszám*)
Input paraméter: *X* – sorozat, $x_i \in \boldsymbol{A}$
Output paraméter: *Sorszám* $\in \boldsymbol{Z}$ – a T tulajdonságú elem indexe
// T: $\boldsymbol{A} \rightarrow \boldsymbol{L}$
$i \leftarrow 1$
WHILE $\overline{T(x_i)}$ **DO**
INC(*i*)
Sorszám ← *i*
RETURN (*Sorszám*)

IV. Lineáris keresés

Eldöntés és kiválasztás együtt. Adott T tulajdonságú elemet megtalálni az input sorozatban, ha van benne. Ha nincs, akkor az az eredmény, hogy nincs ilyen elem.

Keresés (@*X*, @*Van*, @*Sorszám*)

// Input paraméter: *X* – sorozat, $x_i \in \boldsymbol{A}$

// Output paraméter: *Sorszám* $\in \boldsymbol{Z}$ – a T tulajdonságú elem indexe

// *Van* = *igaz* esetén van T tulajdonságú elem és akkor

// *Sorszám* egy ilyen elemnek az indexe.

// *Van* = *hamis* esetén nincs T tulajdonságú elem és akkor

// *Sorszám* értelmezhetetlen

// T: $\boldsymbol{A} \rightarrow \boldsymbol{L}$

$i \leftarrow 1$

WHILE ($i \leq$Hossz[X]) ÉS $\overline{T(x_i)}$ **DO**

INC(i)

Van $\leftarrow$ ($i \leq$ Hossz[X])

IF *Van*

THEN *Sorszám* $\leftarrow i$

RETURN (*Van*, *Sorszám*)

ELSE **RETURN** (*Van*)

V. Megszámolás

Adott tulajdonságú elemek számát kell meghatározni.

```
Megszámolás (@X, @Darab )
Input paraméter: X – sorozat, x_i ∈ A
Output paraméter: Darab ∈ Z – a T tulajdonságú elemek száma
// T: A→ L
Darab ← 0
FOR   i ← 1 TO Hossz[X] DO
         IF    T(x_i)
                 THEN INC(Darab)
RETURN ( Darab )
```

VI. Maximum kiválasztás

A sorozat elemei egy teljesen rendezett halmazból jönnek. A sorozat maximális elemének indexét kell meghatározni.

```
Maximum_kiválasztás (@X, @Sorszám )
Input paraméter: X – sorozat, x_i ∈ A, teljesen rendezett halmaz
Output paraméter: Sorszám ∈ Z – a T tulajdonságú elem indexe
Sorszám ← 1
FOR   i ← 2 TO Hossz[X] DO
         IF    x_Sorszám < x_i
                 THEN Sorszám ← i
RETURN (Sorszám)
```

VII. Másolás

A bemeneti sorozat minden elemének elkészítjük egy transzformáltját. Ez a sorozat az eredmény. A transzformáció: $z=f(x)$.

```
Másolás (@X, @Y )
Input paraméter: X – sorozat, x_i ∈ A
Output paraméter: Y – sorozat, y_i ∈ B
// f : A→ B
Hossz[Y] ← Hossz[X]
FOR   i ← 1 TO Hossz[X] DO
         y_i ← f(x_i)
RETURN (Y)
```

VIII. Kiválogatás

A bemenő sorozat adott T tulajdonságú elemeinek kigyűjtése. A sorszámokat gyűjtjük ki.

Kiválogatás (@X, @Y)			
Input paraméter: X – sorozat, $x_i \in \boldsymbol{A}$			
Output paraméter: Y – sorozat, $y_i \in \boldsymbol{Z}$ – a T tulajdonságú elemek indexei			
// T : $\boldsymbol{A} \rightarrow \boldsymbol{L}$			
Hossz[Y] $\leftarrow 0$			
FOR $i \leftarrow 1$ **TO** Hossz[X] **DO**			
	IF $T(x_i)$		
		THEN	**INC**(Hossz[Y])
			$y_{\text{Hossz}[Y]} \leftarrow i$
RETURN (Y)			

IX. Szétválogatás

Kiválogatás a T tulajdonságú elemekre is és a nem T tulajdonságúakra is. Magukat az elemeket válogatjuk szét.

Szétválogatás (@X, @Y, @Z)			
Input paraméter: X – sorozat, $x_i \in \boldsymbol{A}$			
Output paraméter: Y – sorozat, $y_i \in \boldsymbol{A}$ – a T tulajdonságú elemek			
Z – sorozat, $z_i \in \boldsymbol{A}$ – a nem T tulajdonságú elemek			
// T : $\boldsymbol{A} \rightarrow \boldsymbol{L}$			
Hossz[Y] $\leftarrow 0$			
Hossz[Z] $\leftarrow 0$			
FOR $i \leftarrow 1$ **TO** Hossz[X] **DO**			
	IF $T(x_i)$		
		THEN	**INC**(Hossz[Y])
			$y_{\text{Hossz}[Y]} \leftarrow x_i$
		ELSE	**INC**(Hossz[Z])
			$z_{\text{Hossz}[Z]} \leftarrow x_i$
RETURN (Y, Z)			

X. Metszet

Kiválogatni azokat az elemeket, amelyek közösek a két bemenetben. Feltételezzük, hogy mindkét bemenet külön-külön vizsgálva csupa különböző elemet tartalmaz. Az eredmény is ilyen legyen.

```
Metszet (@X, @Y, @Z )
Input paraméter: X – sorozat, xi ∈ A, csupa különböző
                 Y – sorozat, yi ∈ A, csupa különböző
Output paraméter: Z – sorozat, zi ∈ A, csupa különböző
Hossz[Z] ← 0
FOR  i ← 1 TO Hossz[X] DO
        j ← 1
        WHILE  (j≤Hossz[Y]) ÉS (xi ≠ yj) DO
               INC(j)
        IF     j≤ Hossz[Y]
               THEN   INC(Hossz[Z])
                      zHossz[Z] ← xi
RETURN (Z )
```

XI. Egyesítés

Kiválogatni azokat az elemeket, amelyek a két bemeneti sorozat legalább egyikében benne vannak. Mindkét bemenet külön-külön vizsgálva csupa különböző elemet tartalmaz. Az eredmény is ilyen legyen.

```
Egyesítés (@X, @Y, @Z )
Input paraméter: X – sorozat, xi ∈ A, csupa különböző
                 Y – sorozat, yi ∈ A, csupa különböző
Output paraméter: Z – sorozat, zi ∈ A, csupa különböző
Hossz[Z] ← Hossz[X]
FOR  i ← 1 TO Hossz[X] DO
        zi ← xi
FOR  j ← 1 TO Hossz[Y] DO
        i ← 1
        WHILE  (i≤ Hossz[X]) ÉS (xi ≠ yj) DO
               INC(i)
        IF     i >Hossz[X]
               THEN   INC(Hossz[Z])
                      zHossz[Z] ← yj
RETURN (Z )
```

XII. **<u>Összefuttatás</u>** (Összefésülés)

Két szigorúan monoton növekvő sorozatból egy kimenő szigorúan monoton növekvő sorozatot készíteni. (Azonos elemekből csak egyet tenni az eredmény sorozatba.)

Összefuttatás (@*X*, @*Y*, @*Z*)

Input paraméter: *X* – sorozat, $x_i \in \boldsymbol{A}$, csupa különböző $x_i < x_{i+1}$
Y – sorozat, $y_i \in \boldsymbol{A}$, csupa különböző $y_i < y_{i+1}$

Output paraméter: *Z* – sorozat, $z_i \in \boldsymbol{A}$, csupa különböző $z_i < z_{i+1}$

Hossz[*Z*] ← 0
i ← 1
j ← 1
WHILE (*i*≤Hossz[*X*]) ÉS (*j*≤Hossz[*Y*]) **DO**
  INC(Hossz[*Z*])
  CASE
    $x_i < y_j$: $z_{Hossz[Z]} \leftarrow x_i$
      INC(*i*)
    $x_i = y_j$: $z_{Hossz[Z]} \leftarrow x_i$
      INC(*i*)
      INC(*j*)
    $x_i < y_j$: $z_{Hossz[Z]} \leftarrow y_j$
      INC(*j*)
WHILE *i*≤Hossz[*X*] **DO**
  INC(Hossz[*Z*])
  $z_{Hossz[Z]} \leftarrow x_i$
  INC(*i*)
WHILE *j*≤Hossz[*Y*] **DO**
  INC(Hossz[*Z*])
  $z_{Hossz[Z]} \leftarrow y_j$
  INC(*j*)
RETURN (*Z*)