



ELTE | IK

PROGRAMOZÁS

4. előadás

Horváth Győző, Horváth Gyula, Szlávi Péter



Ismétlés



Feladatmegoldás lépései

1. Specifikáció

- a) Példa
- b) Bemenet, kimenet
 - i. egyszerű adat?
 - ii. több különböző? – rekord
 - iii. több azonos? – **tömb**
- c) Előfeltétel
- d) Utófeltétel

2. Algoritmus

- a) Adat \rightarrow változók
- b) Új halmazok \rightarrow típusok
- c) Beolvasás
- d) Feldolgozás
 - i. támpontok az uf-ben
 - ii. végrehajtható spec.
 - iii. és, vagy, \rightarrow , \forall , \exists
 - iv. **nevezetes minták**
- e) Kiírás

3. Kód

Megfeleltetések

Példa adat	Specifikáció halmaz	Algoritmus típus	Kód type
3	N	Egész	int
-3	Z	Egész	int
3,3	R	Valós	double
igaz	L	Logikai	bool
"alma"	S	Szöveg	string
"a"	K	Karakter	char
(név:"Győző", jegy: 5)	Név x Jegy, S x N	Rekord	struct
[3, 5, -6, 2]	Z[1..n]	Tömb	int[]

Analóg programozás – visszavezetés

- Visszavezetés
 - Konkrét feladat felírása
 - Összevetés a minta sablonjával
 - Különbségek felírása egy táblázatba
 - Különbségek alkalmazása a sablon algoritmusában
 - → Konkrét feladat algoritmus



Programozási minták

1. Összegzés
2. Megszámolás
3. Maximumkiválasztás
4. Feltételes maximumkeresés
5. Keresés
6. Eldöntés
7. Kiválasztás

Most Common DUPLO Parts



Brick Architect BA

szummás, mindenes feladat



számlálós ciklus

létezikes feladat



feltételes ciklus



Másolás függvényyszámítás



Másolás

Feladatok:

- Egy **számsorozat tagjainak** adjuk meg az abszolút értékét!
- Egy szöveget alakítsunk át **csupa** kisbetűssé!
- **Számoljuk ki** két vektor összegét!
- **Készítsünk** függvény**táblázat**ot a $\sin(x)$ függvényről!
- **Ismerünk N** dátumot 'éé.hh.nn' alakban, adjuk meg **őket** 'éé. hónapnév nn.' alakban!

Mi bennük a közös?

n darab „valamihez” kell hozzárendelni másik n darab „valamit”, ami akár az előbbitől különböző típusú is lehet. A darabszám, a sorrend is marad.
Az elemeken operáló függvény ugyanaz.

Példa – abszolút értékek algorithmikus gondolkodással

x			y
1	3,3	→	3,3
2	-5,8	→	5,8
3	4,5	→	4,5
4	-2,2	→	2,2

Feladat:

Egy **számsorozat tagjainak** adjuk meg az abszolút értékét!

Specifikáció:

Be: $n \in \mathbb{N}$, $x \in \mathbb{R}[1..n]$

Ki: $y \in \mathbb{R}[1..n]$

Ef: -

Uf: $\forall i \in [1..n] : (y[i] = \text{abs}(x[i]))$

Algoritmus:

```
i=1..n
  y[i]:=abs(x[i])
```

Másolás sablon

i	y
e	$\rightarrow f(e)$
e+1	$\rightarrow f(e+1)$
...	$\rightarrow \dots$
u	$\rightarrow f(u)$

Feladat

Adott az egész számok egy $[e..u]$ intervalluma és egy $f:[e..u] \rightarrow H$ függvény. Rendeljük az $[e..u]$ intervallum minden értékéhez az f függvény hozzá tartozó értékét!

Specifikáció

Be: $e \in \mathbb{Z}, u \in \mathbb{Z}$

Ki: $y \in H[1..u-e+1]$

Ef: -

Uf: $\forall i \in [e..u]: (y[i-e+1] = f(i))$

Rövidítve:

Uf: $y = \text{MÁSOL}(i=e..u, f(i))$

Algoritmus

i = e..u

y[i-e+1] := f(i)

Példa – két vektor összege visszavezetés

Számoljuk ki két vektor összegét!

Feladatsablon

(mintafeladat)

Be: $e \in \mathbb{Z}$, $u \in \mathbb{Z}$

Ki: $y \in H[1..u-e+1]$

Ef: -

Uf: $y = \text{MÁSOL}(i=e..u, f(i))$



Két vektor összege

(konkrét feladat)

Be: $n \in \mathbb{N}$, $p \in R[1..n]$, $q \in R[1..n]$

Ki: $r \in R[1..n]$

Ef: -

Uf: $r = \text{MÁSOL}(i=1..n, p[i]+q[i])$

Visszavezetés:

y	\sim	r
$e..u$	\sim	$1..n$
$f(i)$	\sim	$p[i]+q[i]$

Algoritmus:

$i = e..u$

$y[i-e+1] := f(i)$

$i = 1..n$

$r[i] := p[i] + q[i]$

Kiválogatás



Kiválogatás

Feladatok:

- **Adjuk** meg egy osztály kitűnő tanulóit!
- **Adjuk** meg egy természetes szám **összes** osztóját!
- **Adjuk** meg egy mondat magas hangrendű szavait!
- **Adjuk** meg emberek egy halmazából a 180 cm felettieket!
- **Adjuk** meg egy év azon napjait, amikor délben nem fagyott!
- **Soroljuk** föl egy szó magánhangzóit!

Mi bennük a közös?

n darab „valami” közül kell megadni az összes, adott T tulajdonsággal rendelkezőt!

Példa – nem fagyos napok algorithmikus gondolkodással

hőm			poz	
1	-2,2	→ db=2	1	2
2	1,5		2	3
3	2,8			
n=4	-1,0			

Feladat:

Adjuk meg egy év azon napjait, amikor délben nem fagyott!

Specifikáció:

Be: $n \in \mathbb{N}$, $\text{hőm} \in \mathbb{R}[1..n]$

Ki: $db \in \mathbb{N}$, $\text{poz} \in \mathbb{N}[1..db]$

Ef: $\forall i \in [1..n] : (-100 \leq \text{hőm}[i] \leq 100)$

Uf: $db = \text{DARAB}(i=1..n, \text{hőm}[i] > 0)$ és

$\forall i \in [1..db] : (\text{hőm}[\text{poz}[i]] > 0)$ és

$\forall i \in [1..db] : (\forall j \in [1..db] : (i < j \rightarrow \text{poz}[i] < \text{poz}[j]))$

megszámolás

a kiválogatott indexekhez
tartozó értékek pozitívak

nincs két egyforma index a
poz tömbben, részsorozata
az indestartományak

Algoritmus:

db:=0

i=1..n

hőm[i]>0

true

false

db:=db+1

poz[db]:=i

Példa – nem fagyos napok algorithmikus gondolkodással

hőm			poz	
1	-2,2	→ db=2	1	2
2	1,5			3
3	2,8			
n=4	-1,0			

Feladat:

Adjuk meg egy év azon napjait, amikor délben nem fagyott!

Specifikáció:

Be: $n \in \mathbb{N}$, $\text{hőm} \in \mathbb{R}[1..n]$

Ki: $db \in \mathbb{N}$, $\text{poz} \in \mathbb{N}[1..db]$

Ef: $\forall i \in [1..n]: (-100 \leq \text{hőm}[i] \leq 100)$

Uf: $db = \text{DARAB}(i=1..n, \text{hőm}[i] > 0)$ és

$\forall i \in [1..db]: (\text{hőm}[\text{poz}[i]] > 0)$ és

$\text{poz} \subseteq [1..n]$

megszámolás

a kiválogatott indexekhez
tartozó értékek pozitívak

nincs két egyforma index a
poz tömbben

Algoritmus:

db:=0

i=1..n

hőm[i]>0

true

false

db:=db+1

poz[db]:=i

Példa – nem fagyos napok algorithmikus gondolkodással

hőm			poz	
1	-2,2	→ db=2	1	2
2	1,5			3
3	2,8			
n=4	-1,0			

Feladat:

Adjuk meg egy év azon napjait, amikor délben nem fagyott!

Specifikáció:

Be: $n \in \mathbb{N}$, $\text{hőm} \in \mathbb{R}[1..n]$

Ki: $\text{db} \in \mathbb{N}$, $\text{poz} \in \mathbb{N}[1..\text{db}]$

Ef: $\forall i \in [1..n]: (-100 \leq \text{hőm}[i] \leq 100)$

Uf: $\text{db} = \text{DARAB}(i=1..n, \text{hőm}[i] > 0)$ és
 $\forall i \in [1..\text{db}]: (\text{hőm}[\text{poz}[i]] > 0)$ és
 $\text{poz} \subseteq [1..n]$

Algoritmus:

db:=0

i=1..n

hőm[i]>0

true

false

db:=db+1

poz[db]:=i

Példa – kitűnő tanulók algoritmikus gondolkodással

Feladat:

Adjuk meg egy osztály kitűnő tanulóit!

Specifikáció és algoritmus:

Be: $n \in \mathbb{N}$, $\text{diákok} \in \text{Diák}[1..n]$, $\text{Diák} = \text{Név} \times \text{Jegy}$, $\text{Név} = S$, $\text{Jegy} = N$

Ki: $db \in \mathbb{N}$, $\text{jelesek} \in S[1..db]$

Ef: $\forall i \in [1..n]: (1 \leq \text{diákok}[i].\text{jegy} \leq 5)$

Uf: $db = \text{DARAB}(i=1..n, \text{diákok}[i].\text{jegy}=5)$ és

$\forall i \in [1..db]: (\exists j \in [1..n]:$
 $\text{diákok}[j].\text{jegy}=5 \text{ és } \text{jelesek}[i]=\text{diákok}[j].\text{név})$ és

$\forall i \in [1..db]: (\forall j \in [1..db]: (i < j \rightarrow$
 $\exists ii \in [1..n]: (\exists jj \in [1..n]:$
 $ii < jj \text{ és }$
 $\text{diákok}[ii].\text{név} = \text{jelesek}[i]$
 $\text{diákok}[jj].\text{név} = \text{jelesek}[j]$

))
))

a kiválogatott nevek
különböző indexekhez
tartoznak, sőt részsorozat!

diákok			jelesek	
	név	jegy		
1	P	4	1	F
2	F	5	→ db=2	G
3	E	2		
n=4	G	5		

megszámolás

a kiválogatott nevek olyan
indexhez tartoznak, ahol a
jegy 5-ös

db:=0

i=1..n

diákok[i].jegy=5

true

db:=db+1

jelesek[db]:=diákok[i].név

false

Példa – kitűnő tanulók algorithmikus gondolkodással

Variáció:

Részsorozat rövidítve

Specifikáció és algoritmus:

Be: $n \in \mathbb{N}$, $\text{diákok} \in \text{Diák}[1..n]$, $\text{Diák} = \text{Név} \times \text{Jegy}$, $\text{Név} = S$, $\text{Jegy} = N$

Ki: $db \in \mathbb{N}$, $\text{jelesek} \in S[1..db]$

Ef: $\forall i \in [1..n]: (1 \leq \text{diákok}[i].\text{jegy} \leq 5)$

Uf: $db = \text{DARAB}(i=1..n, \text{diákok}[i].\text{jegy}=5)$ és
 $\forall i \in [1..db]: (\exists j \in [1..n]:$
 $\text{diákok}[j].\text{jegy}=5 \text{ és } \text{jelesek}[i] = \text{diákok}[j].\text{név})$ és
 $\text{jelesek} \subseteq \text{diákok.név}$

a kiválogatott nevek
különböző indexekhez
tartoznak, sőt részsorozat!

diákok			jelesek	
	név	jegy		
1	P	4	1	F
2	F	5	→ db=2	G
3	E	2		
n=4	G	5		

megszámolás

a kiválogatott nevek olyan
indexhez tartoznak, ahol a
jegy 5-ös

db:=0

i=1..n

diákok[i].jegy=5

true

db:=db+1

jelesek[db]:=diákok[i].név

false

Példa – kitűnő tanulók algoritmikus gondolkodással

diákok			jelesek	
	név	jegy		
1	P	4	1	F
2	F	5	→ db=2	G
3	E	2		
n=4	G	5		

Variáció:

Részsorozat rövidítve

Specifikáció és algoritmus:

Be: $n \in \mathbb{N}$, $\text{diákok} \in \text{Diák}[1..n]$, $\text{Diák} = \text{Név} \times \text{Jegy}$, $\text{Név} = S$, $\text{Jegy} = N$

Ki: $db \in \mathbb{N}$, $\text{jelesek} \in S[1..db]$

Ef: $\forall i \in [1..n] : (1 \leq \text{diákok}[i].\text{jegy} \leq 5)$

Uf: $db = \text{DARAB}(i=1..n, \text{diákok}[i].\text{jegy}=5)$ és
 $\forall i \in [1..db] : (\exists j \in [1..n] : (\text{diákok}[j].\text{jegy}=5 \text{ és } \text{jelesek}[i] = \text{diákok}[j].\text{név}))$ és
 $\text{jelesek} \subseteq \text{diákok.név}$

megszámolás

a kiválogatott nevek olyan indexhez tartoznak, ahol a jegy 5-ös

a kiválogatott nevek
különböző indexekhez
tartoznak

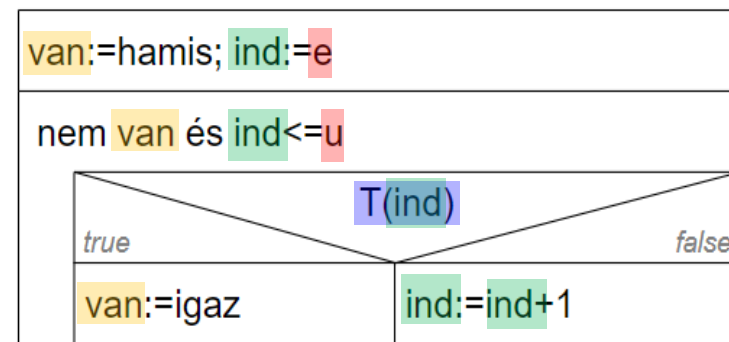
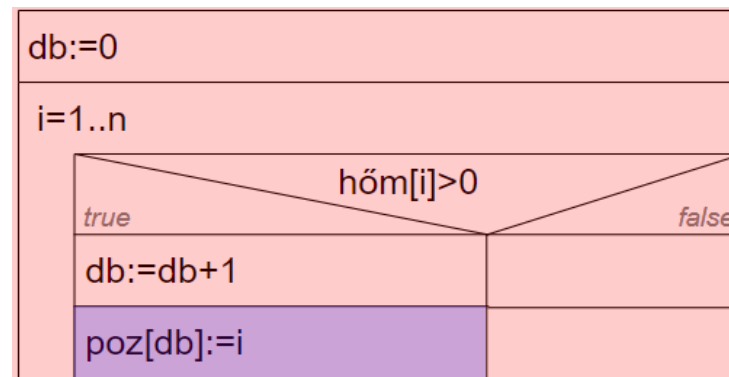
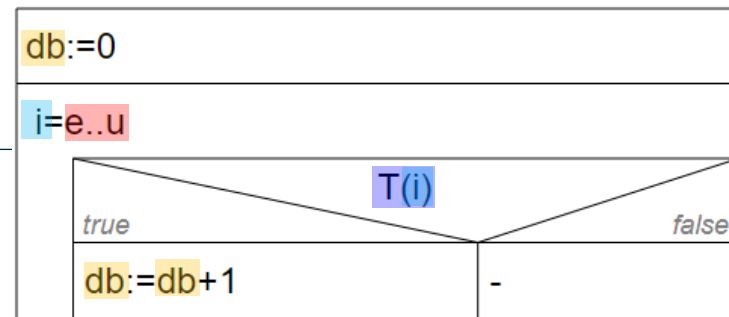
db:=0

i:=1..n

diákok[i].jegy=5	
true	false
db:=db+1	
jelesek[db]:=diákok[i].név	

Tanulságok

- A kiválogatás hasonlít:
 - Megszámolásra
 - Hányszor teljesült a T tulajdonság?
 - + mely esetekben?
 - Keresésre
 - Ha teljesült, akkor hol a T tulajdonság?
 - + mindenkre



Tanulságok

- Eredményképpen
 - intervallum elemei, vagy
 - ezekhez rendelt értékek

diákok			jeles			jelesek		
	név	jegy	→	indexek	→			
1	P	4		1	2	1	F	
2	F	5		db=2	3	db=2	G	
3	E	2						
n=4	G	5						

db:=0	
i=1..n	
hőm[i]>0	
true	false
db:=db+1	
poz[db]:=i	

db:=0	
i=1..n	
diákok[i].jegy=5	
true	false
db:=db+1	
jelesek[db]:=diákok[i].név	

Kiválogatás sablon

i	T(i)	f(i)
e	→ HAMIS	
e+1	→ IGAZ →	1 f(e+1)
e+2	→ IGAZ →	2 f(e+2)
u	→ HAMIS	

Feladat

Adott az egész számok egy $[e..u]$ intervalluma, egy ezen értelmezett $T:[e..u] \rightarrow \text{Logikai feltétel}$ és egy $f:[e..u] \rightarrow H$ függvény. Határozzuk meg az f függvény az $[e..u]$ intervallum azon értékeinél felvett értékeit, amelyekre a T feltétel teljesül!

Specifikáció

Be: $e \in \mathbb{Z}$, $u \in \mathbb{Z}$

Ki: $db \in \mathbb{N}$, $y \in H[1..db]$

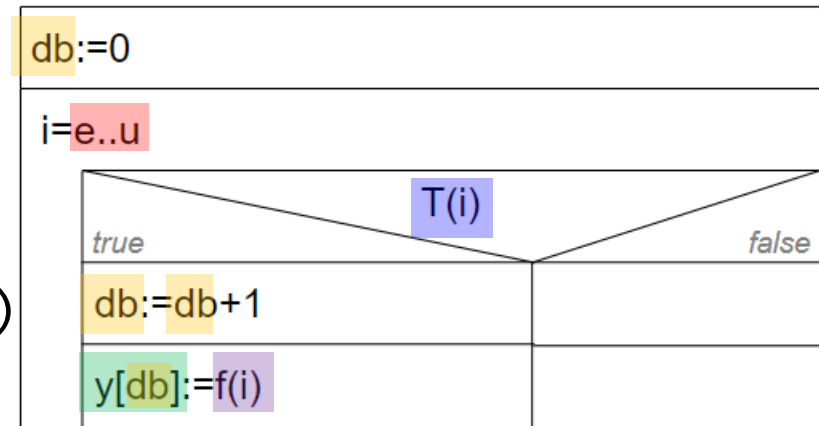
Ef: -

Uf: $db = \text{DARAB}(i=e..u, T(i))$ és
 $\forall i \in [1..db]:$
 $\exists j \in [e..u]: T(j) \text{ és } y[i] = f(j))$
és $y \subseteq (f(e), f(e+1), \dots, f(u))$

Rövidítve:

Uf: $(db, y) = \text{KIVÁLOGAT}(i=e..u, T(i), f(i))$

Algoritmus



Példa – összes osztó visszavezetés

Adjuk meg egy természetes szám összes osztóját!

Feladatsablon

(mintafeladat)

Be: $e \in \mathbb{Z}$, $u \in \mathbb{Z}$

Ki: $db \in \mathbb{N}$, $y \in H[1..db]$

Ef: -

Uf: $(db, y) =$

$KIVÁLOGAT(i = e..u, T(i), f(i))$

Visszavezetés:

Algoritmus:

$db := 0$

$i = e..u$

$T(i)$
<i>true</i>
<i>false</i>

$db := db + 1$

$y[db] := f(i)$

Összes osztó

(konkrét feladat)

Be: $n \in \mathbb{N}$

Ki: $db \in \mathbb{N}$, $osztók \in [1..db]$

Ef: -

Uf: $(db, osztók) =$

$KIVÁLOGAT(i = 1..n, i|n, i)$



$y \sim osztók$
 $e..u \sim 1..n$
 $T(i) \sim i|n$
 $f(i) \sim i$

$db := 0$

$i = 1..n$

$i n$
<i>true</i>
<i>false</i>

$db := db + 1$

$osztók[db] := i$



Példa – kitűnő tanuló visszavezetés

Adjuk meg egy osztály kitűnő tanulóit!

Feladatsablon

Be: $e \in \mathbb{Z}, u \in \mathbb{Z}$

Ki: $db \in \mathbb{N}, y \in H[1..db]$

Ef: -

Uf: $(db, y) =$

KIVÁLOGAT($i = e..u,$

$T(i),$

$f(i))$

$y \sim \text{osztók}$

$e..u \sim 1..n$

$T(i) \sim \text{diákok}[i].\text{jegy} = 5$

$f(i) \sim \text{diákok}[i].\text{név}$



Kitűnő tanuló

Be: $n \in \mathbb{N}, \text{diákok} \in \text{Diák}[1..n],$

$\text{Diák} = \text{Név} \times \text{Jegy}, \text{Név} = S, \text{Jegy} = N$

Ki: $db \in \mathbb{N}, \text{jelesek} \in S[1..db]$

Ef: -

Uf: $(db, \text{jelesek}) =$

KIVÁLOGAT($i = 1..n,$

$\text{diákok}[i].\text{jegy} = 5,$

$\text{diákok}[i].\text{név})$

$db := 0$

$i = e..u$

$T(i)$	
true	false
$db := db + 1$	
$y[db] := f(i)$	



$db := 0$

$i = 1..n$

$\text{diákok}[i].\text{jegy} = 5$	
true	false
$db := db + 1$	
$\text{jelesek}[db] := \text{diákok}[i].\text{név}$	

Gyakori programozási minta változatok



Minimumkiválasztás példa – specifikáció

minimumkiválasztás

Feladat:

Adjuk meg egy adott $f:[e..u] \rightarrow H$ függvény értékei között a legkisebb elemet!

Specifikáció:

Be: $e \in \mathbb{Z}, u \in \mathbb{Z}$

Ki: $\text{minind} \in \mathbb{Z}, \text{minért} \in H$

Ef: $e \leq u$

Uf: $\text{minind} \in [e..u]$ és

$\forall i \in [e..u]: (f(\text{minind}) \leq f(i))$ és

$\text{minért} = f(\text{minind})$

Uf: $(\text{minind}, \text{minért}) = \text{MIN}(i=e..u, f(i))$

egyetlen különbség a
maximumkiválasztáshoz
képest az elnevezések
mellett

Minimumkiválasztás algorithmus – analóg algoritmikus gondolkodással

Algoritmus:

másik reláció használata

maxért:=f(e);maxind:=e	
i=e+1..u	
<div><div>f(i)>maxért</div><div><div>true</div><div>false</div></div></div>	
maxért:=f(i)	-
maxind:=i	

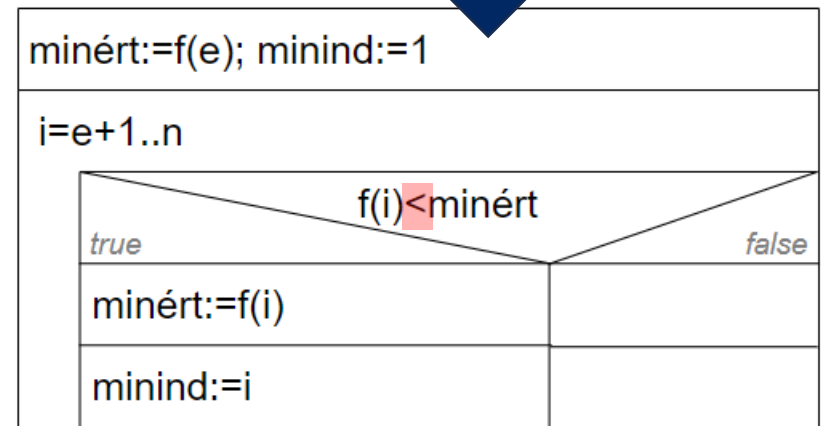
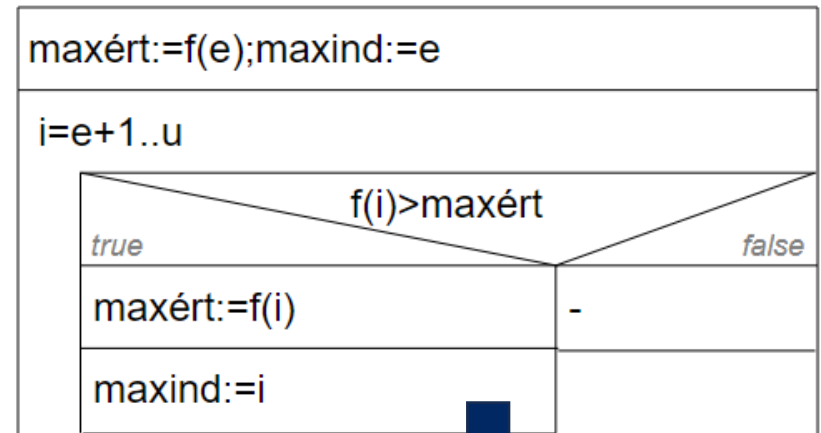


minért:=f(e); minind:=1	
i=e+1..n	
<div><div>f(i)<minért</div><div><div>true</div><div>false</div></div></div>	
minért:=f(i)	
minind:=i	

Minimumkiválasztás algoritmus – visszavezetéssel

Algoritmus:

- \leq „felüldefiniálása”
- eddig csak azt használtuk ki, hogy ez teljes rendezési reláció
- jelentése nem volt érdekes
- helyettesíthető bármilyen teljes rendezési relációval
- azaz $\leq(a,b)=a\geq b$



Minimumkiválasztás algoritmus – visszavezetéssel

Algoritmus:

- ötlet: alakítsuk át a feladatot!
- a függvényértékek **ellentettjei** között keressünk maximumot!
- a megtalált maximum a keresett minimum **ellentettje** lesz

Maximumkiválasztás

Be: $e \in \mathbb{Z}, u \in \mathbb{Z}$

Ki: $\text{maxind} \in \mathbb{Z}, \text{maxért} \in \mathbb{H}$

Ef: $e \leq u$

Uf: $(\text{maxind}, \text{maxért}) =$
 $\text{MAX}(i=e..u, f(i))$

Visszavezetés:

$\text{maxind}, \text{maxért}$	\sim	$\text{minind}, -\text{minért}$
$f(i)$	\sim	$-f(i)$

Minimumkiválasztás

Be: $e \in \mathbb{Z}, u \in \mathbb{Z}$

Ki: $\text{minind} \in \mathbb{Z}, \text{minért} \in \mathbb{H}$

Ef: $e \leq u$

Uf: $(\text{minind}, -\text{minért}) =$
 $\text{MAX}(i=e..u, -f(i))$

ötlet!

Minimumkiválasztás algoritmus – visszavezetéssel

$\text{maxind}, \text{maxért} \sim \text{minind}, -\text{minért}$
 $f(i) \sim -f(i)$

maximumkiválasztás sablonja

Algoritmus:

$\text{maxért} := f(e); \text{maxind} := e$	
$i = e+1..u$	
$f(i) > \text{maxért}$	
true	false
$\text{maxért} := f(i)$	-
$\text{maxind} := i$	



nem megengedett értékadás

$-\text{minért} := -f(e); \text{minind} := 1$	
$i = e+1..n$	
$-f(i) > -\text{minért}$	
true	false
$-\text{minért} := -f(i)$	
$\text{minind} := i$	



szorozzunk -1-gyel a szükséges helyeken!

algoritmikus gondolkodással

$\text{minért} := f(e); \text{minind} := 1$	
$i = e+1..n$	
$f(i) < \text{minért}$	
true	false
$\text{minért} := f(i)$	
$\text{minind} := i$	



$\text{minért} := f(e); \text{minind} := 1$	
$i = e+1..n$	
$f(i) < \text{minért}$	
true	false
$\text{minért} := f(i)$	
$\text{minind} := i$	

Minimumkiválasztás sablon

Feladat

Adott az egész számok egy $[e..u]$ intervalluma és egy $f:[e..u] \rightarrow H$ függvény. A H halmaz elemein értelmezett egy teljes rendezési reláció. Határozzuk meg, hogy az f függvény **hol** veszi fel az $[e..u]$ nem üres intervallumon a legkisebb értéket, és mondjuk meg, **mekkora** ez a minimális érték!

Specifikáció

Be: $e \in \mathbb{Z}$, $u \in \mathbb{Z}$

Ki: $\text{minind} \in \mathbb{Z}$, $\text{minért} \in H$

Ef: $e \leq u$

Uf: $\text{minind} \in [e..u]$ és
 $\forall i \in [e..u]: (f(\text{minind}) \leq f(i))$ és
 $\text{minért} = f(\text{minind})$

Rövidítve:

Uf: $(\text{minind}, \text{minért}) = \text{MIN}(i=e..u, f(i))$

Algoritmus

$\text{minért} := f(e); \text{minind} := e$	
$i = e+1..u$	
$f(i) < \text{minért}$	
true	false
$\text{minért} := f(i)$	
$\text{minind} := i$	

Hátulról keresés példa

Feladat:

Keressük meg a legutolsó T tulajdonságú elemet!

Specifikáció:

Be: $e \in \mathbb{Z}$, $u \in \mathbb{Z}$

Ki: $van \in \mathbb{L}$, $ind \in \mathbb{Z}$

Ef: -

Uf: $van = \exists i \in [e..u] : (T(i))$ és

$van \rightarrow (ind \in [e..u] \text{ és } T(ind) \text{ és}$

$\forall i \in [ind+1..u] : (\text{nem } T(i)))$

vö. az előlről keresésnél:
[e..ind-1]

Hátulról keresés algoritmus – algoritmikus gondolkodással

Algoritmus:

induljunk hátulról, lépegezzünk előre, amíg az intervallumon belül vagyunk és nem találtunk T tulajdonságú elemet

ind:=e
ind<=u és nem T(ind)
ind:=ind+1
van:=ind<=u

ind:=u
ind>=e és nem T(ind)
ind:=ind-1
van:=ind>=e

Hátulról keresés algoritmus – visszavezetéssel

Algoritmus:

- ötlet: alakítsuk át a feladatot!
- az $[e..u]$ intervallumot tükrözzük az origóra, és az így kapott **$[-u..-e]$ intervallumban keressük** az elsőt adott tulajdonságú elemet
- a talált ind helyett a **$-ind$ lesz az eredmény**

Előlről keresés sablonja

Be: $e \in Z, u \in Z$

Ki: $van \in L, ind \in Z$

Ef: -

Uf: $(van, ind) = KERES(i=e..u, T(i))$

Hátulról keresés

Be: $e \in Z, u \in Z$

Ki: $van \in L, ind \in Z$

Ef: -

Uf: $(van, -ind) = KERES(i=-u..-e, T(i))$

ötlet!

Visszavezetés:

ind	\sim	$-ind$
$e..u$	\sim	$-u..-e$

Hátulról keresés algoritmus – visszavezetéssel

keresés sablonja

```
ind:=e  
ind<=u és nem T(ind)  
  ind:=ind+1  
van:=ind<=u
```



$e..u \sim -u..-e$

```
ind:=-u  
ind<=-e és nem T(-ind)  
  ind:=ind+1  
van:=ind<=-e
```

nem megengedett értékadás

$ind \sim -ind$
 $e..u \sim -u..-e$

$-ind:=-u$

$ind \sim -ind$

```
-ind<=-e és nem T(ind)  
  -ind:=-ind+1  
van:=-ind<=-e
```

algoritmikus gondolkodással

```
ind:=u  
ind>=e és nem T(ind)  
  ind:=ind-1  
van:=ind>=e
```



```
ind:=u  
ind>=e és nem T(ind)  
  ind:=ind-1  
van:=ind>=e
```

szorozzuk -1-gyel a szükséges helyeken!

Mind eldöntés példa

Feladat:

Adjuk meg, hogy egy $[e..u]$ intervallum minden eleme rendelkezik-e egy adott T tulajdonsággal!

Specifikáció:

Be: $e \in \mathbb{Z}, u \in \mathbb{Z}$

Ki: $\text{mind} \in \mathbb{L}$

Ef: -

Uf: $\text{mind} = \forall i \in [e..u] : (T(i))$

Rövidítve:

Uf: $\text{mind} = \mathbf{MIND}(i=e..u, T(i))$

eldöntésben: $\exists i \in [e..u] : (T(i))$

Mind eldöntés algoritmus – algoritmikus gondolkodással

Algoritmus:

Addig lépkedjünk előre, amíg jót találunk. Ha lelépünk az intervallumról, akkor mindegyik jó volt.

$i := e$
$i \leq u$ és nem $T(i)$
$i := i + 1$
$\text{van} := i \leq u$

$i := e$
$i \leq u$ és $T(i)$
$i := i + 1$
$\text{mind} := i > u$

Mind eldöntés algorithmus – visszavezetéssel

Algorithmus:

- ötlet: alakítsuk át a feladatot!
- **van-e** olyan elem, ami **nem rendelkezik a T tulajdonsággal**?
- a **válasz negáltja** lesz a keresett eredmény

Van-e olyan?

Be: $e \in Z, u \in Z$

Ki: $\text{van} \in L$

Ef: -

Uf: $\text{van} = \text{VAN}(i=e..u, T(i))$

Visszavezetés:

van	\sim	nem mind
$T(i)$	\sim	$\text{nem } T(i)$

Mind olyan-e?

Be: $e \in Z, u \in Z$

Ki: $\text{mind} \in L$

Ef: -

Uf: $\text{mind} = \text{VAN}(i=e..u, \text{nem } T(i))$

Uf: $\text{nem mind} = \text{VAN}(i=e..u, \text{nem } T(i))$

ötlet!

Mind eldöntés algorithmus – visszavezetéssel

van	~	nem mind
$T(i)$	~	nem $T(i)$

eldöntés sablonja

$i := e$
$i \leq u$ és nem $T(i)$
$i := i + 1$
van: $i \leq u$

nem megengedett értékadás

$i := e$
$i \leq u$ és nem nem $T(i)$
$i := i + 1$
nem mind: $i \leq u$

algorithmikus gondolkodással

$i := e$
$i \leq u$ és $T(i)$
$i := i + 1$
mind: $i > u$

negálunk a szükséges helyeken

$i := e$
$i \leq u$ és $T(i)$
$i := i + 1$
mind: $i > u$

Mind eldöntés (vagy optimista eldöntés) sablon

Feladat

Adott az egész számok egy $[e..u]$ intervalluma és egy $T:[e..u] \rightarrow \text{Logikai feltétel}$. Határozzuk meg, hogy az $[e..u]$ intervallumnak mindegyik eleme olyan-e, amely kielégíti a T feltételt!

Specifikáció

Be: $e \in \mathbb{Z}, u \in \mathbb{Z}$

Ki: $\text{mind} \in \mathbb{L}$

Ef: -

Uf: $\text{mind} = \forall i \in [e..u]: (T(i))$

Rövidítve:

Uf: $\text{mind} = \text{MIND}(i=e..u, T(i))$

Algoritmus

$i := e$	
$i \leq u$ és $T(i)$ <table><tr><td>$i := i + 1$</td></tr></table>	$i := i + 1$
$i := i + 1$	
$\text{mind} := i > u$	

Futóindex mint sablonrész



Példa – legmelegebb vasárnap

viSSZavezetés

Hétfői naptól kezdve mértük a déli hőmérsékletet.
Add meg a legmelegebb vasárnapot!

Feladatsablon

Be: $e \in \mathbb{Z}$, $u \in \mathbb{Z}$

Ki: $\text{maxind} \in \mathbb{Z}$, $\text{maxért} \in \mathbb{H}$

Ef: $e \leq u$

Uf: $(\text{maxind}, \text{maxért}) =$
 $\text{MAX}(i = e..u, f(i))$

ViSSZavezetés:

Algoritmus:

Legmelegebb vasárnap

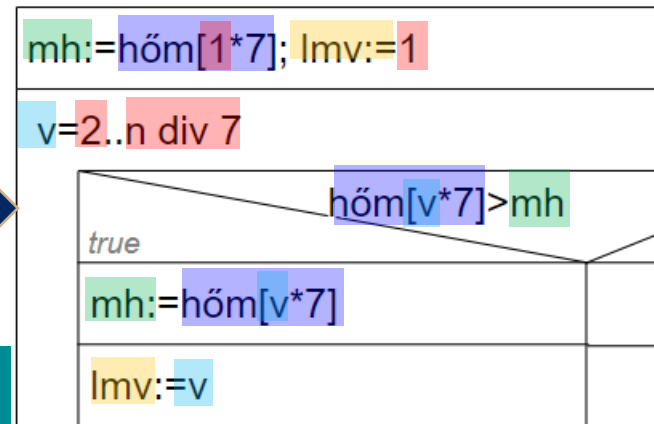
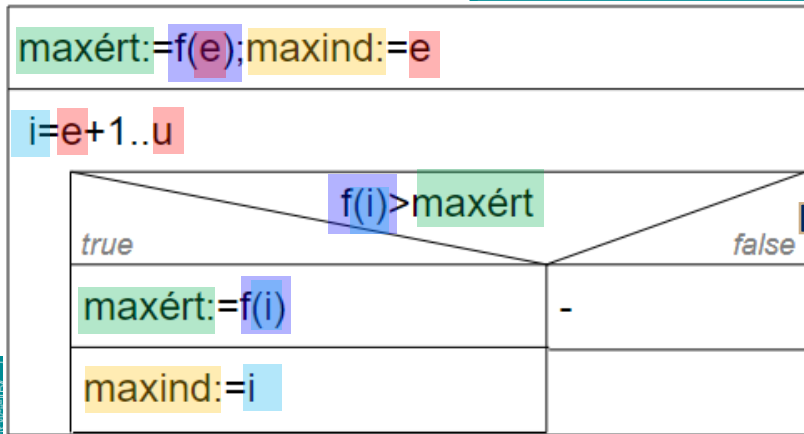
Be: $n \in \mathbb{N}$, $\text{hőm} \in \mathbb{R}[1..n]$

Ki: $\text{lmv} \in \mathbb{N}$, $\text{mh} \in \mathbb{R}$

Ef: $n \geq 7$

Uf: $(\text{lmv}, \text{mh}) =$
 $\text{MAX}(v = 1..n \text{ div } 7, \text{hőm}[v*7])$

$\text{maxind}, \text{maxért}$	\sim	lmv, mh
i	\sim	v
$e..u$	\sim	$1..n \text{ div } 7$
$f(i)$	\sim	$\text{hőm}[v*7]$



i	hőm
1	23,3
2	20,1
3	
4	
5	
6	
e 7	23,3
8	
9	
10	
11	
12	
13	
e+1 14	12,7
15	
16	
17	
18	
19	
20	
e+2 21	25,6
22	
23	

Összefoglalás



Programozási minták

1. Összegzés
2. Megszámolás
3. Maximumkiválasztás
 - a. Minimumkiválasztás
4. Feltételes maximumkeresés
5. Keresés
6. Eldöntés
 - a. Mind eldöntés
7. Kiválasztás
8. Másolás
9. Kiválogatás
 - a. Értékek