



ELTE | IK

# PROGRAMOZÁS

## Programtranszformációk

Horváth Győző, Szlávi Péter



# Ismétlés



# 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

Most Common DUPLO Parts



# Több programozási minta használata egymás után



# Több minta alkalmazása

---

- **Összetettebb feladatok** nem vezethetők vissza csupán egyetlen programozási mintára
- **Több programozási minta** együttes használata szükséges!
- Egyelőre foglalkozzunk olyan feladatokkal, ahol a mintákat **egymás után** kell alkalmazni!

# Maximumkiválasztás+kiválogatás

**Feladat:** Adott számok sorozata. Add meg az összes maximális elemet.

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

Sa:  $\text{maxért} \in \mathbb{Z}$

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

Ef:  $n > 0$

Uf:  $(, \text{maxért}) = \text{MAX}(i=1..n, x[i])$  és

$(\text{db}, \text{maxI}) = \text{KIVÁLOGAT}(i=1..n, x[i] = \text{maxért}, i)$

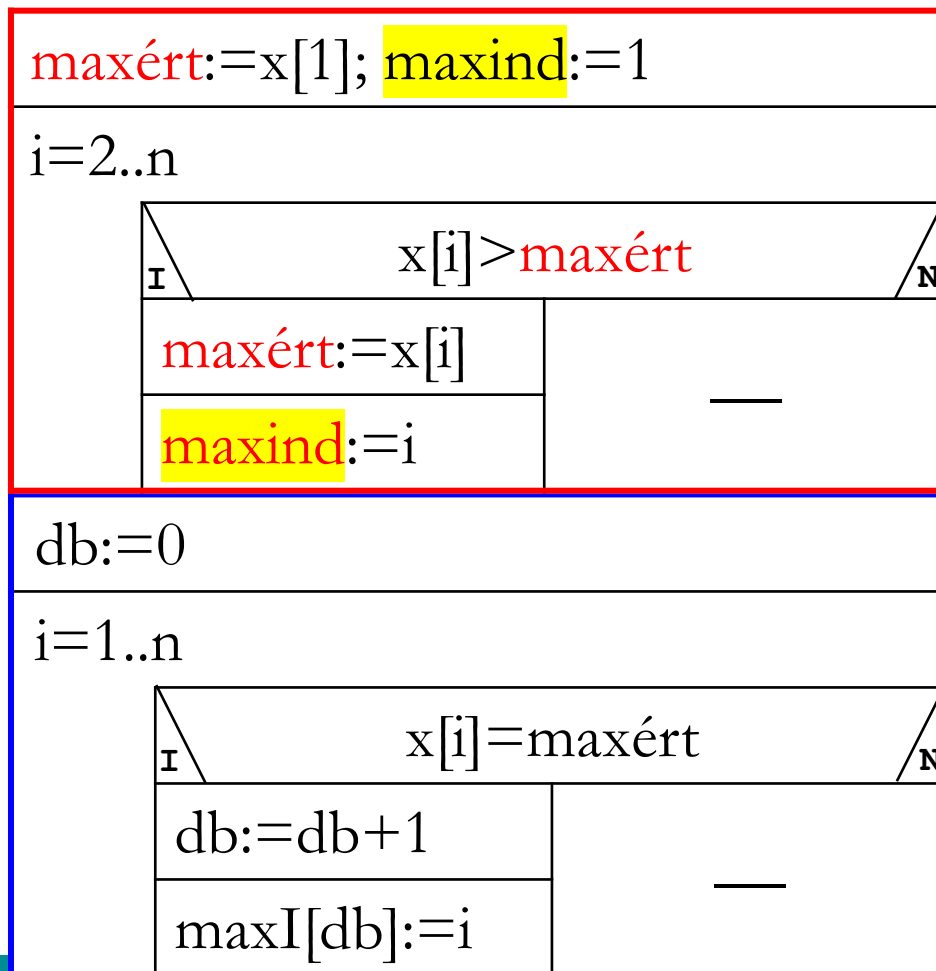
1. lépés: maximális érték meghatározása

A két lépés közötti kapcsolatot egy közbülső segédadat biztosítja

2. minta: maximális elemek indexeinek kiválogatása

# Maximumkiválasztás+kiválogatás

## Algoritmus:



Változó

$i, \text{maxind}$ : Egész

$\text{maxért}$ : TH

Észrevétel:

Az eredmény helyes,  
de bántóan nem  
hatékony.

Próbáljuk hatékonyabbra  
írni a kapott algoritmust!

# Programtranszformációk





# Cél, szerkezet...

Az algoritmus ekvivalens átalakítása, melynek célja

- hatékonyabbra írás
- egyszerűsítés
- megvalósíthatóság

Vö. Programozási tétel szerkezetével!

- specifikáció
- algoritmus

**Szerkezete:**

- $algoritmus_1, algoritmus_2$
- *feltétel*

Vö. Programozási tétel állításával!

Ha a specifikáció beli  $E_f$  a bemeneti adatokra teljesül, akkor az algoritmus végrehajtása után az  $U_f$  teljesül.

**Állítás:**

Ha *feltétel* teljesül, akkor

$algoritmus_1 \approx_{\text{szemantikusan}} algoritmus_2$

# Bevezető példa

## Maximumkiválasztás:

<Távolság

Melyik az origótól legmesszebb levő mP pont ( $p \in \text{Pont}[1..n]$ ,  $\text{Pont} = X \times Y$ )

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:  $\text{maxind} \in [e..u]$  és  
 $\forall i \in [e..u]: (f(\text{maxind}) \geq f(i))$  és  
 $\text{maxért} = f(\text{maxind})$

Rövidítve:

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

Be:  $n \in \mathbb{N}$ ,  $p \in \text{Pont}[1..n]$ ,  
 $\text{Pont} = X \times Y$ ,  $X, Y = \mathbb{R}$

Ki:  $mP \in \mathbb{N}$

Ef:  $n > 0$

Uf:  $mP \in [1..n]$  és  
 $\forall i \in [1..n]:$   
 $\sqrt{p[mP].x^2 + p[mP].y^2} \geq$   
 $\sqrt{p[i].x^2 + p[i].y^2}$  ) és  
 $\text{maxért} = \sqrt{p[mP].x^2 + p[mP].y^2}$

Rövidítve:

Uf:  $(mP, ) =$   
 $\text{MAX}(i = 1..n, \sqrt{p[i].x^2 + p[i].y^2})$

# Bevezető példa

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:  $\text{maxind} \in [e..u]$  és  
 $\forall i \in [e..u]: (f(\text{maxind}) \geq f(i))$  és  
 $\text{maxért} = f(\text{maxind})$

Rövidítve:

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

Be:  $n \in \mathbb{N}, p \in \text{Pont}[1..n],$   
 $\text{Pont} = X \times Y, X, Y = \mathbb{R}$

Ki:  $mP \in \mathbb{N}$

Ef:  $n > 0$

Uf:  $mP \in [1..n]$  és  
 $\forall i \in [1..n]:$   
 $\sqrt{p[mP].x^2 + p[mP].y^2} \geq$   
 $\sqrt{p[i].x^2 + p[i].y^2})$  és  
 $\text{maxért} = \sqrt{p[mP].x^2 + p[mP].y^2}$

Rövidítve:

Uf:  $(mP, ) =$   
 $\text{MAX}(i = 1..n, \sqrt{p[i].x^2 + p[i].y^2})$

Visszavezetés:

|                                |        |                            |
|--------------------------------|--------|----------------------------|
| $\text{maxind}, \text{maxért}$ | $\sim$ | $mP, \text{maxért}$        |
| $e..u$                         | $\sim$ | $1..n$                     |
| $f(i)$                         | $\sim$ | $\sqrt{p[i].x + p[i].y^2}$ |

# Bevezető példa

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

|                |   |                              |
|----------------|---|------------------------------|
| maxind, maxért | ~ | mP, maxért                   |
| e..u           | ~ | 1..n                         |
| f(i)           | ~ | $\sqrt{p[i].x^2 + p[i].y^2}$ |

Négyzetgyök függvény

|  |   |
|--|---|
| mP:=1; maxért:=gyök(p[1].x <sup>2</sup> +p[1].y <sup>2</sup> ) |   |
| i=2..n   |   |
| gyök(p[i].x <sup>2</sup> +p[i].y <sup>2</sup> )>maxÉrt         |   |
| I  | N |
| mP:=i  | — |
| maxért:=gyök(p[i].x <sup>2</sup> +p[i].y <sup>2</sup> )        |   |

Változó  
i:Egész  
maxért:Valós

# Bevezető példa

|   |  |                                    |          |
|---|--|------------------------------------|----------|
| mP:=1; maxért:= <b>gyök</b> (p[1].x <sup>2</sup> +p[1].y <sup>2</sup> ) |  | Változó<br>i:Egész<br>maxért:Valós |          |
| i=2..n  |  |                                    |          |
| <b>I</b>  | <b>gyök</b> (p[i].x <sup>2</sup> +p[i].y <sup>2</sup> )>maxért |                                    | <b>N</b> |
| mP:=i   |  |                                    | —        |
| maxért:= <b>gyök</b> (p[i].x <sup>2</sup> +p[i].y <sup>2</sup> )        |  |                                    |          |

A **gyök** függvényt tartalmazó kifejezéseket emeljük négyzetre!

# Bevezető példa

|   |  |              |
|---|--|--------------|
| mP:=1; maxért:=gyök(p[1].x <sup>2</sup> +p[1].y <sup>2</sup> ) <sup>2</sup> |  | Változó      |
| i=2..n  |  | i:Egész      |
| gyök(p[i].x <sup>2</sup> +p[i].y <sup>2</sup> ) <sup>2</sup> >maxért        |  | maxért:Valós |
| I   | mP:=i  | —            |
|   | maxért:=gyök(p[i].x <sup>2</sup> +p[i].y <sup>2</sup> ) <sup>2</sup> |              |
| N   |  |              |

A **maxért** jelentését újrafogalmaztuk: **legyen a távolság-négyzetek maximuma!** Így kap értéket a 2 értékadásban. Ez legális mivel,  $0 \leq a \leq b \rightarrow a^2 \leq b^2$ , ezért a feltételben szereplő reláció szemantikusan változatlan marad.

A maxért nem kimeneti adat, így a specifikációt sem sérti.

# Bevezető példa

|   |  |                                    |
|---|--|------------------------------------|
| mP:=1; maxért:=gyök(p[1].x <sup>2</sup> +p[1].y <sup>2</sup> ) <sup>2</sup> |  | Változó<br>i:Egész<br>maxért:Valós |
| i=2..n  |  |                                    |
| I   | gyök(p[i].x <sup>2</sup> +p[i].y <sup>2</sup> ) <sup>2</sup> >maxért |                                    |
| N   |  |                                    |
| mP:=i   |  | —                                  |
| maxért:=gyök(p[i].x <sup>2</sup> +p[i].y <sup>2</sup> ) <sup>2</sup>        |  |                                    |

A négyzetre emelés és gyökvonás egymás inverzei, tehát kiejtik egymást, és  $\text{gyök}(A) < \text{gyök}(B) \rightarrow A < B$ , miatt a feltétel szemantikusan változatlan.

# Bevezető példa

|                                     |                                   |                                    |
|-------------------------------------|-----------------------------------|------------------------------------|
| mP:=1; maxért:= $p[1].x^2+p[1].y^2$ |                                   | Változó<br>i:Egész<br>maxért:Valós |
| i=2..n                              |                                   |                                    |
| I                                   | $p[i].x^2+p[i].y^2>\text{maxért}$ |                                    |
| mP:=i                               | —                                 |                                    |
| maxért:= $p[i].x^2+p[i].y^2$        |                                   |                                    |

Itt még **ugyanazt** a képletet többször számítjuk ki (a ciklusban).

Használjunk egy **segéd változót**!



# Bevezető példa

|   |            |
|---|------------|
| mP:=1; maxért:=p[1].x <sup>2</sup> +p[1].y <sup>2</sup> |            |
| i=2..n  |            |
| táv:=p[i].x <sup>2</sup> +p[i].y <sup>2</sup>           |            |
| I   | táv>maxért |
| mP:=i   | —          |
| maxért:=táv   |            |
|   | N          |

Változó  
i:Egész  
maxÉrt,  
táv:Valós

# Nevezetes programtranszformációk

---

## Párhuzamos értékadás kifejtése:

$$a,b,c:=f(x),g(x),h(x)$$

Egymás utáni kiszámításra bontható, ha az összefüggés körmentes:

$$a:=f(x); b:=g(x); c:=h(x)$$

# Nevezetes programtranszformációk

## Párhuzamos értékadás kifejtése (ellenpélda):

$$a,b,c:=b,c,a$$

A szabály szerinti „szekvenciális párja”:

$$a:=b; b:=c; c:=a$$

Baj van: a kör-körös hivatkozás miatt a szekvenciális végrehajtás során megváltozott érték kerül a később értéket kapó változóba.

# Nevezetes programtranszformációk

## Párhuzamos értékadás kifejtése<sub>2</sub>:

$$a,b,c:=b,c,a$$

segédváltozóval egymás utáni kiszámításra bontható, ha az összefüggés kört tartalmaz:

$$\text{segéd}:=a; a:=b; b:=c; c:=\text{segéd}$$

Változó  
 $\text{segéd}:\text{TH}$

# Nevezetes programtranszformációk

## Párhuzamos értékadás kifejtése **általános**:

$$a,b,c := f(x,a,b,c), g(x,a,b,c), h(x,a,b,c)$$

segédváltozókkal egymás utáni kiszámításra bontható, ha az összefüggés kört tartalmaz:

$$\textcolor{red}{sa} := a; \textcolor{red}{sb} := b; \textcolor{red}{sc} := c$$
$$a := f(x, \textcolor{red}{sa}, \textcolor{red}{sb}, \textcolor{red}{sc}); b := g(x, \textcolor{red}{sa}, \textcolor{red}{sb}, \textcolor{red}{sc}); c := h(x, \textcolor{red}{sa}, \textcolor{red}{sb}, \textcolor{red}{sc})$$

Változó  
 $\textcolor{red}{sa}, \textcolor{red}{sb}, \textcolor{red}{sc} : TH$

# Nevezetes programtranszformációk

---

## Függvénykompozíció:

$$B := g(A); C := f(B)$$

Ha

1. az  $f$  függvény nem változtatja meg a paraméterét (B-t),  
és
2. a B értékre nincs később szükség,  
akkor

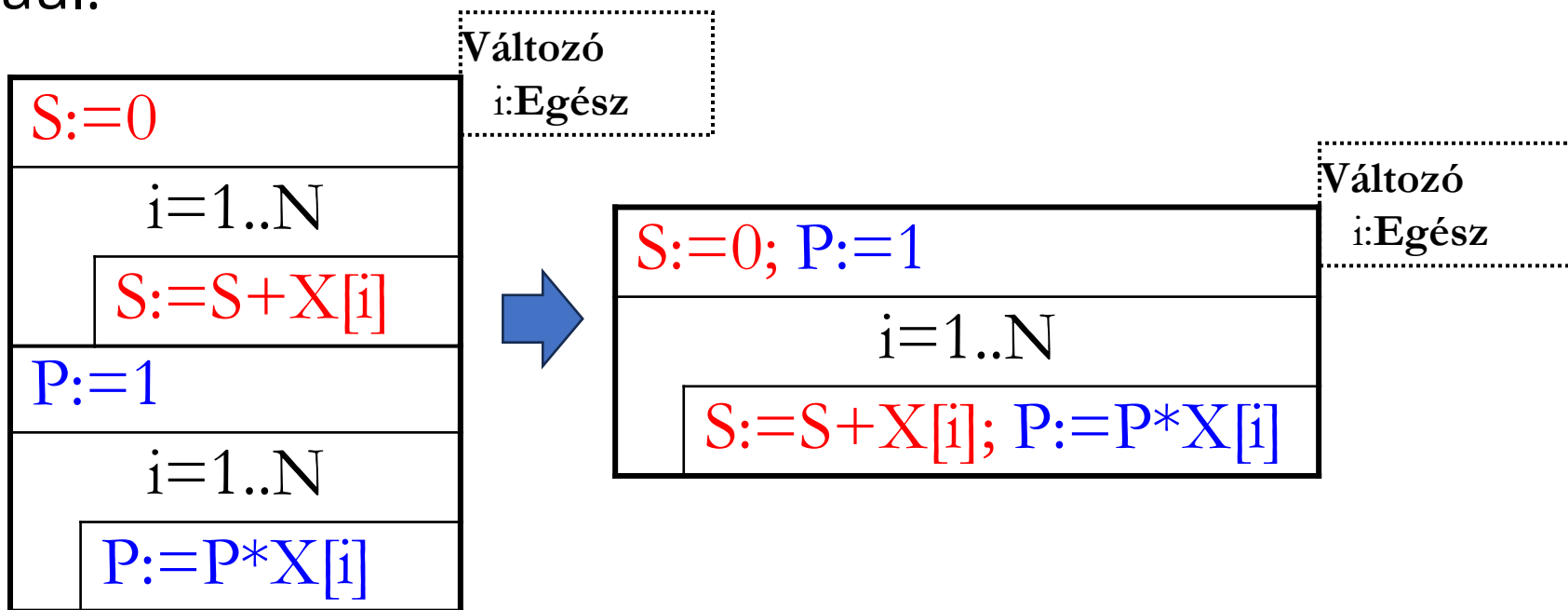
$$C := f(g(A))$$

# Nevezetes programtranszformációk

## Ciklusok összevonása:

Azonos lépésszámú ciklusok összevonhatóak, ha függetlenek egymástól.

Például:

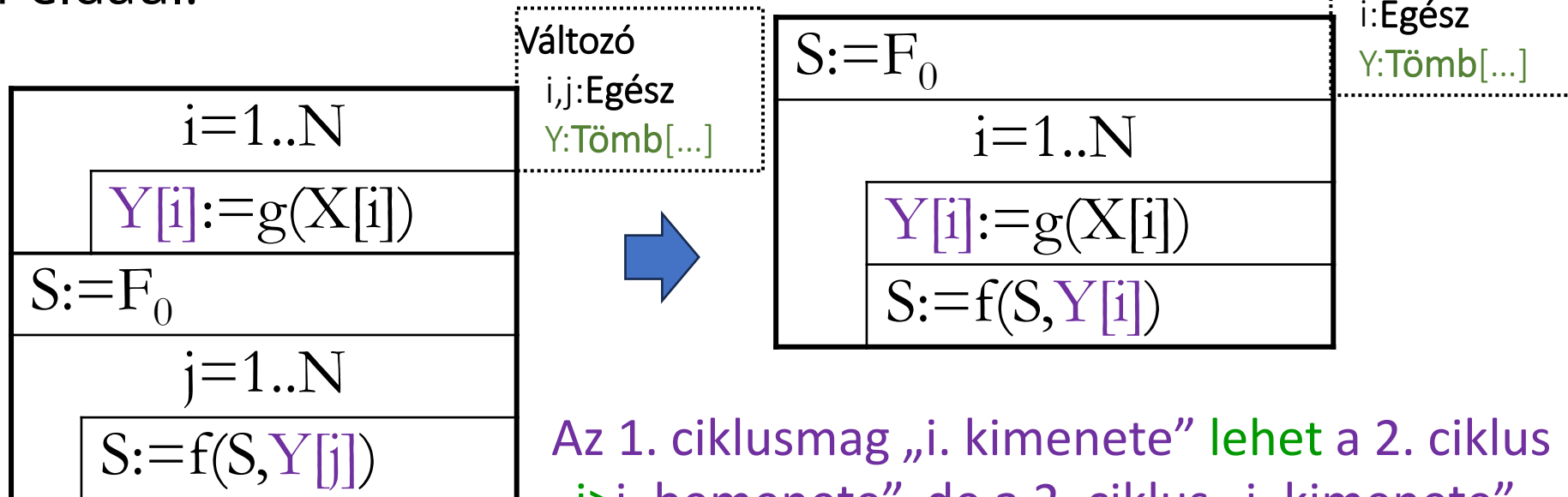


# Nevezetes programtranszformációk

## Ciklusok összevonása (gyenge függés):

„Gyenge” függés megengedhető.

Például:



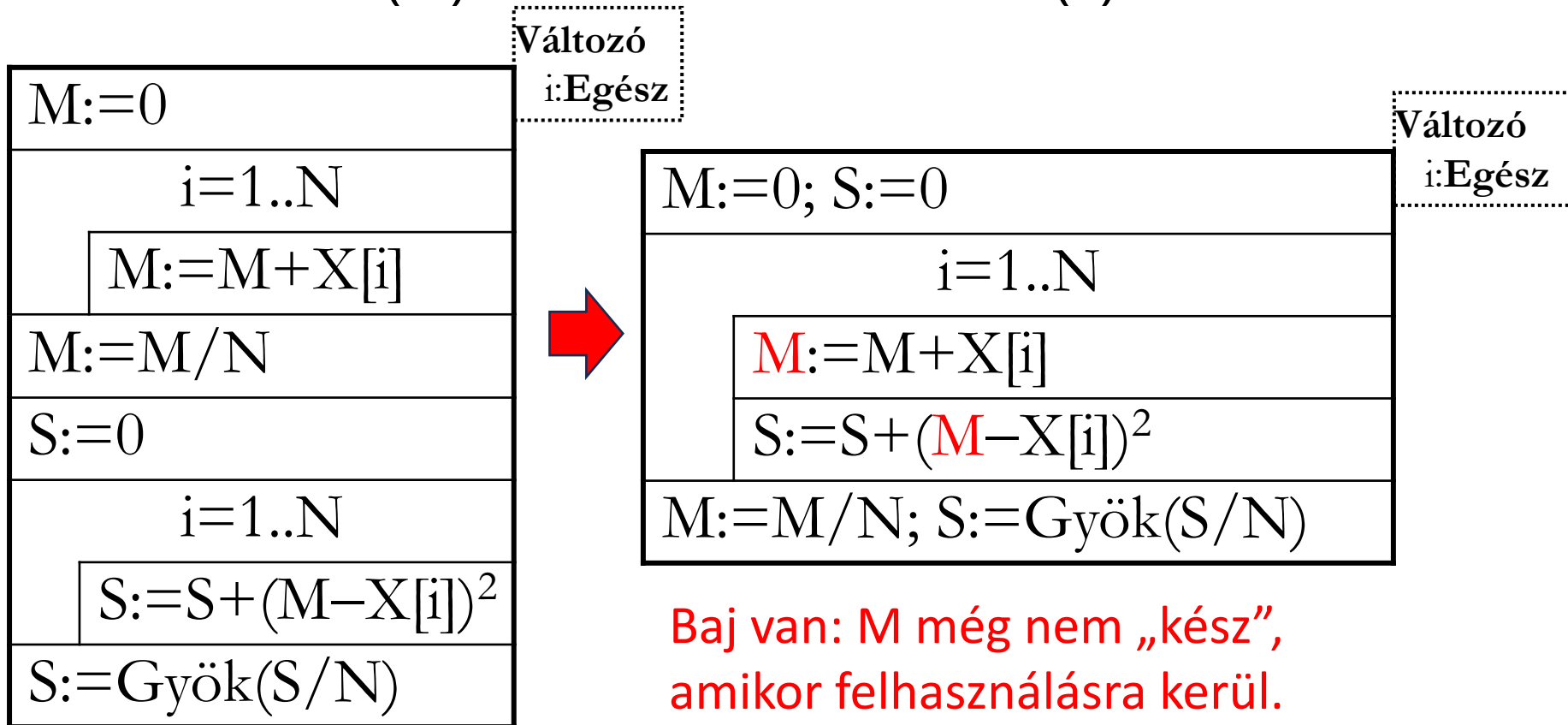
Az 1. ciklusmag „i. kimenete” lehet a 2. ciklus „j>i. bemenete”, de a 2. ciklus „i. kimenete” nem lehet az 1. ciklus „j>i bemenete”.



# Nevezetes programtranszformációk

## Ciklusok összevonása (ellenpélda):

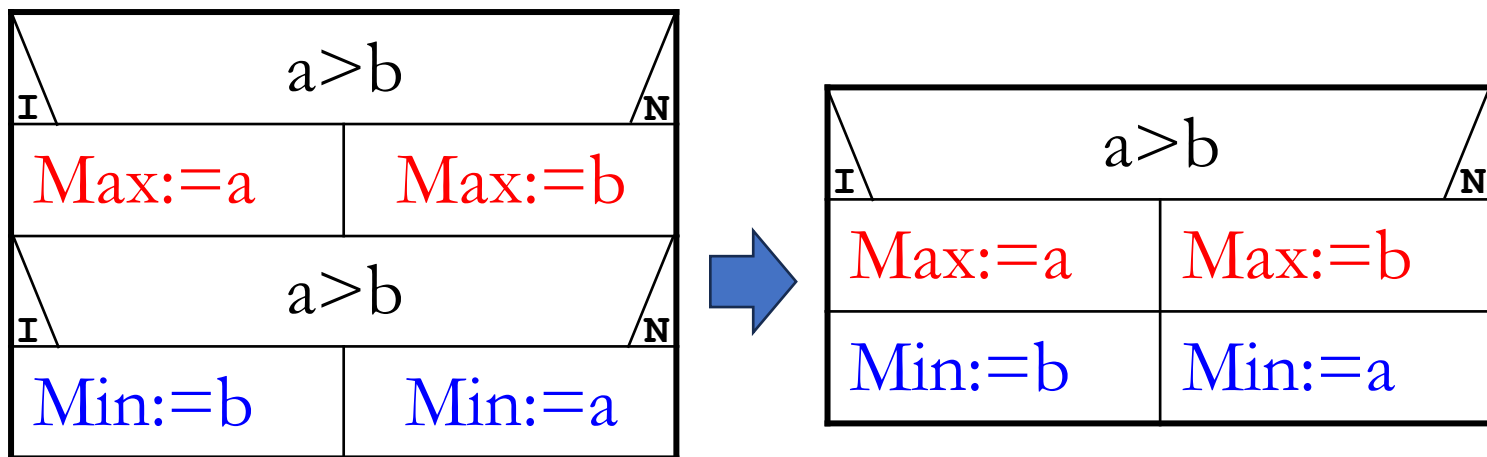
A várhatóérték (M) és szórás kiszámolása (S):



# Nevezetes programtranszformációk

## Elágazások összevonása:

Azonos feltételű elágazások összevonhatóak, ha függetlenek egymástól.



Függetlenek, ha az 1. feltétel egyik ágán sem változik meg sem az  $a'$ , sem a  $b'$  változó (kifejezés). Gondolja meg: mikor nem független a két elágazás, ha  $\text{feltétel}(a,b)$  függvény a közös feltétel?

# Nevezetes programtranszformációk

## Elágazások összevonása (ellenpélda):

| T(x)          |              |
|---------------|--------------|
| I             | N            |
| a:=x          | <b>x</b> :=a |
| T( <b>x</b> ) |              |
| I             | N            |
| b:= <b>x</b>  | x:=b         |



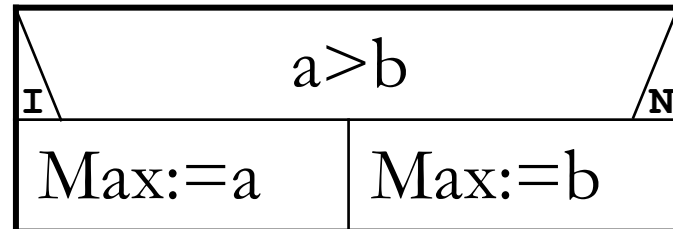
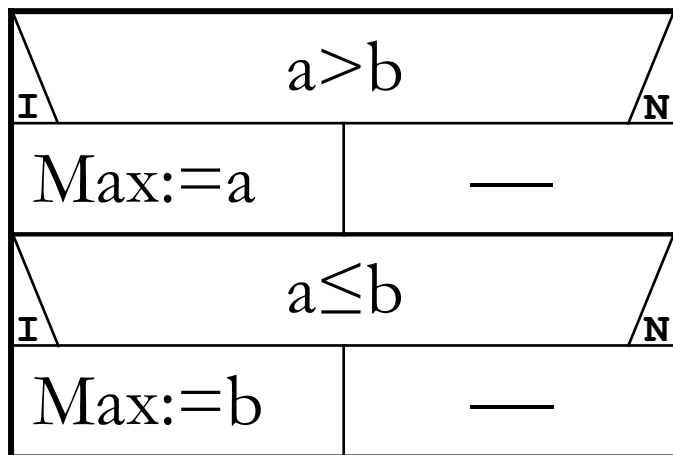
| T(x)         |              |
|--------------|--------------|
| I            | N            |
| a:=x         | <b>x</b> :=a |
| <b>b</b> :=x | <b>x</b> :=b |

Baj van: x megváltozhat  
a második elágazásig.

# Nevezetes programtranszformációk

## Elágazások összevonása:

Kizáró feltételű, teljes (egyágú) elágazások is összevonhatók, ha függetlenek egymástól.

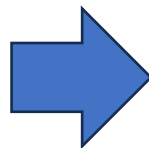


# Nevezetes programtranszformációk

## Elágazások összevonása:

Kizáró feltételű elágazásokkal is összevonhatók, ha függetlenek egymástól.

|  |   |
|--|---|
| $\swarrow$ $X[\text{max}] < X[i]$ $\searrow$ $\text{I}$ $\text{N}$ |   |
| $\text{max} := i$  | — |
| $\swarrow$ $X[\text{min}] > X[i]$ $\searrow$ $\text{I}$ $\text{N}$ |   |
| $\text{min} := i$  | — |

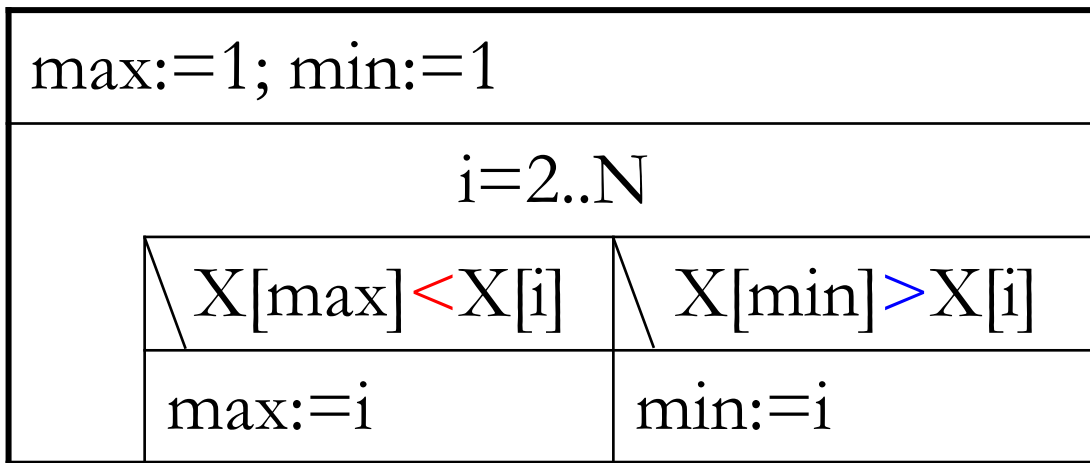


|  |  |
|--|--|
| $\swarrow$ $X[\text{max}] < X[i]$ $\searrow$ | $\swarrow$ $X[\text{min}] > X[i]$ $\searrow$ |
| $\text{max} := i$                            | $\text{min} := i$                            |

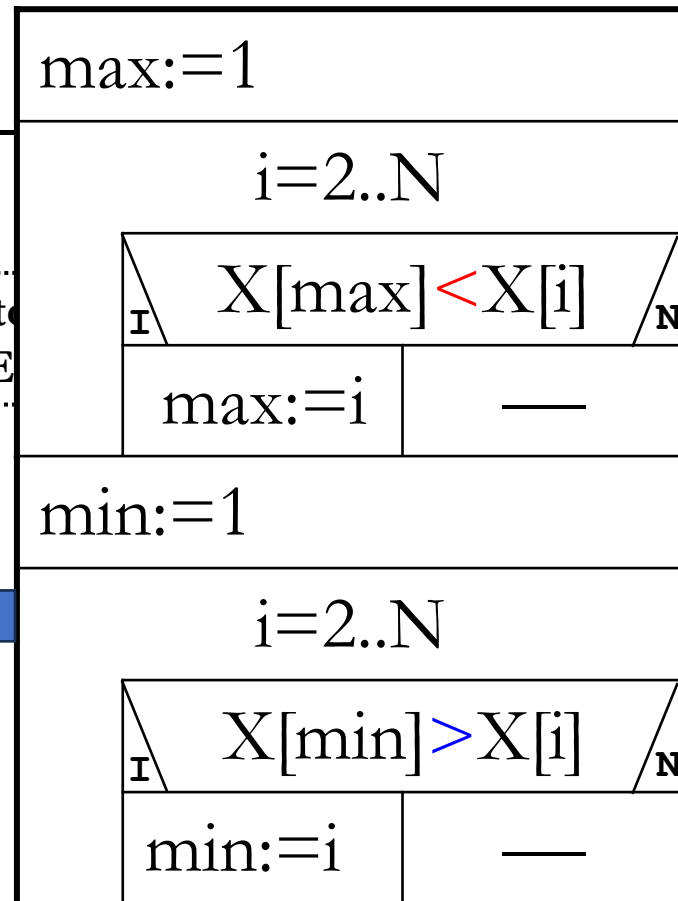
# Nevezetes programtranszformációk

## Ciklusok és elágazások összevonása:

Azonos lépésszámú ciklusok, bennük kizáró feltételű elágazásokkal is összevonhatók, ha függetlenek egymástól.



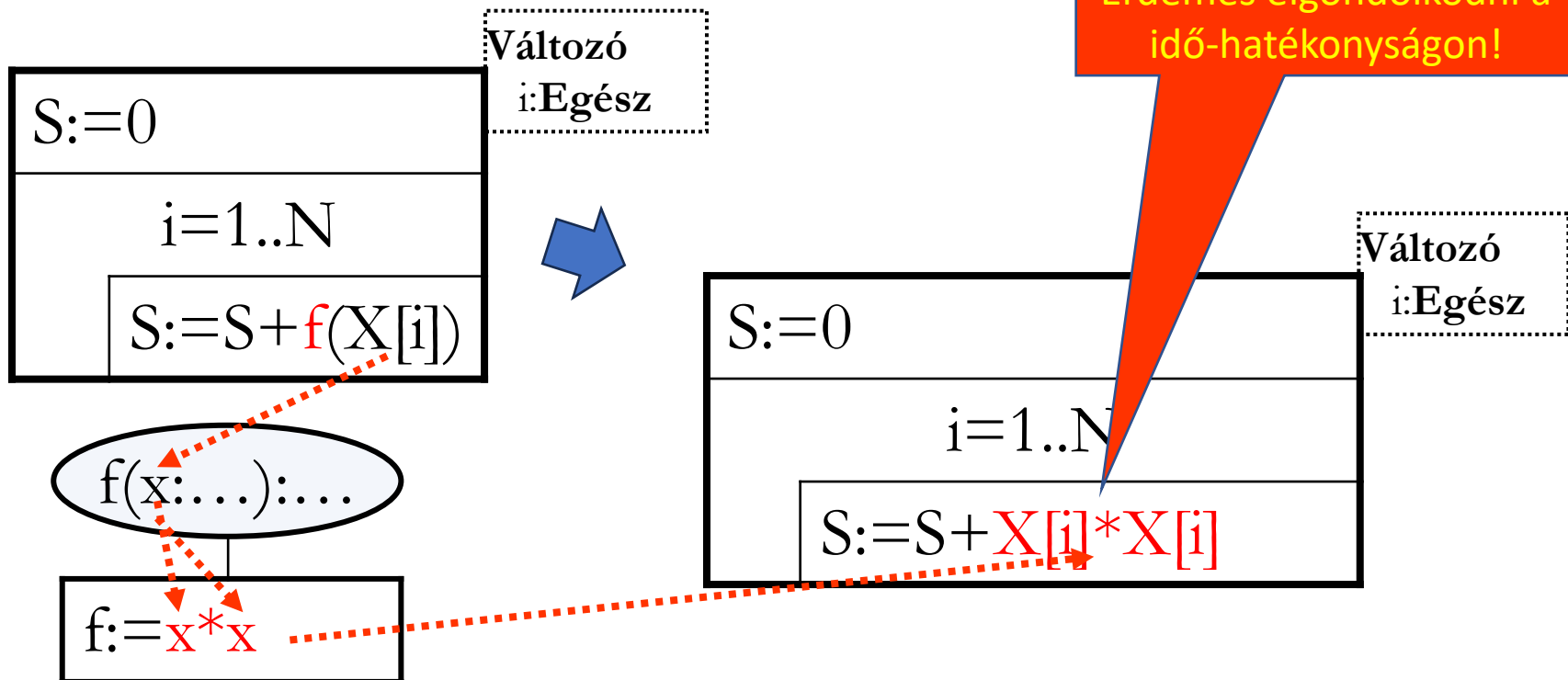
Változó  
i:E



# Nevezetes programtranszformációk

## Függvény behelyettesítése:

Függvényhívás helyére egy (egyszerű) függvény képlete (a függvény törzse) behelyettesíthető.

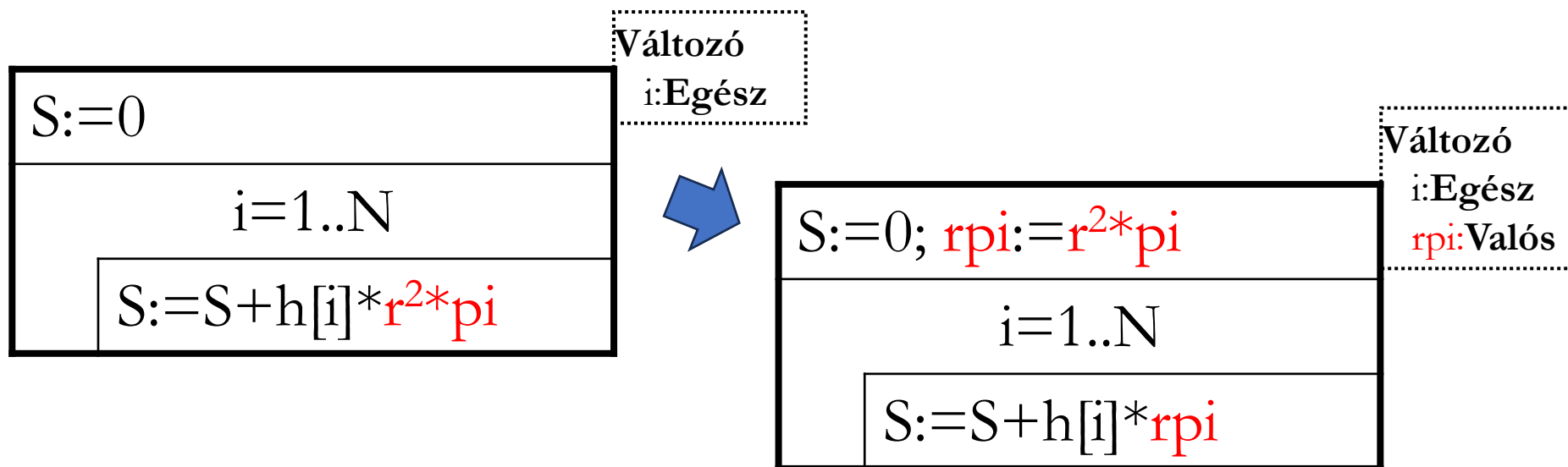


# Nevezetes programtranszformációk

## Utasítás kiemelése ciklusból:

A ciklus magjából a ciklustól független utasítások kiemelhetők.

(A fordítók ilyen optimalizálást többnyire el tudnak végezni.)

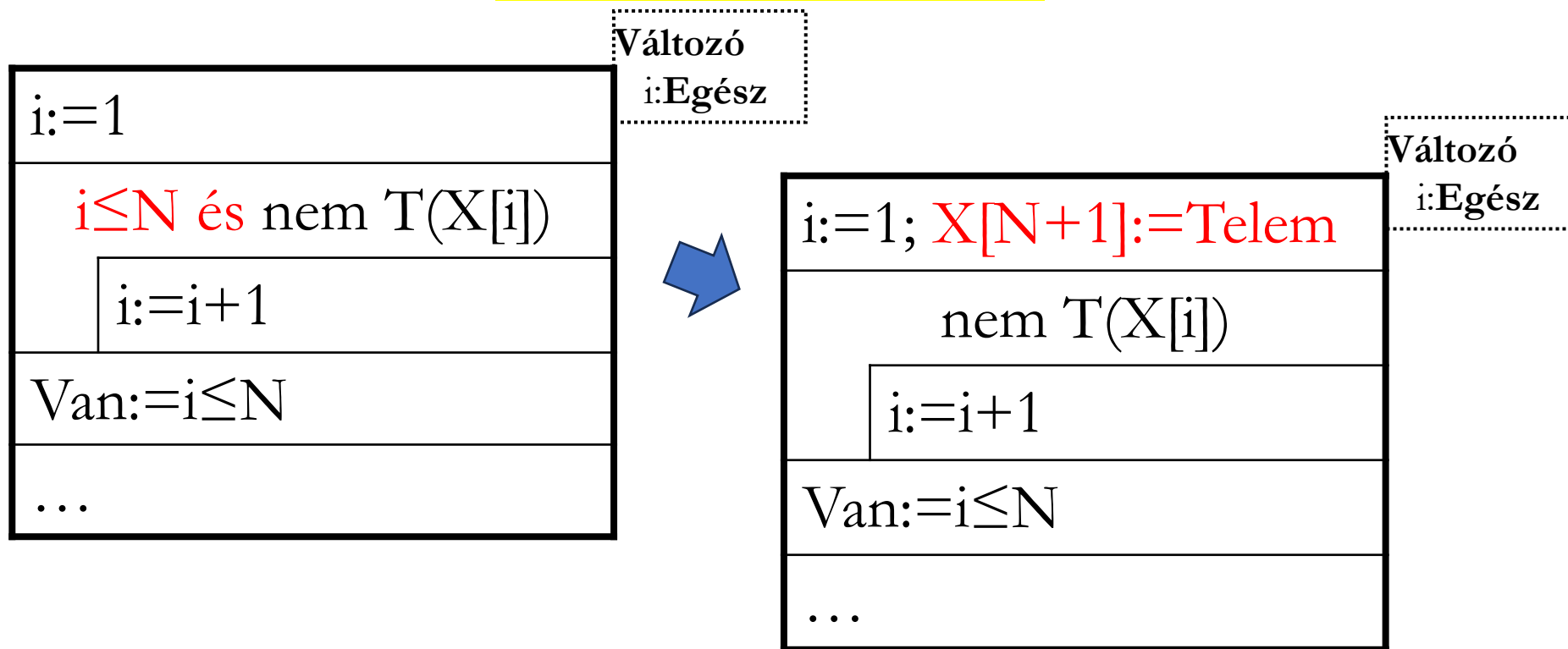




# Nevezetes programtranszformációk

„Keresés, **eldöntés** → kiválasztás” transzformáció:

A vizsgálandó sorozat végére helyezzünk egy T tulajdonságú elemet (=Telem) → **biztosan találunk ilyen!**



# Programtranszformációk alkalmazása Tételek összeépítése



# Tételek összeépítése elé...

---

- Tétel-kombinálás „módszertana”:
  - mechanikusan vagy
  - „okosan”
- Az elkövetkezőkben sorozatokon értelmezzük a programozási tételeket

# Tételek összeépítése elé...

---

Milyen tételek lehetnek „**főszereplők**” ( $T_0$ ) a kombináláskor?

Tekintsük a tételeket mint függvényeket (függvény-szignatúra):

*tétel: bemenet (értelmezési tartomány)  $\rightarrow$  kimenet (értékkészlet)*

A bemenet minden tétel esetében legalább 1 sorozat.

# Tételek összeépítése elé...

---

Milyen tételek lehetnek „**főszereplők**” ( $T_0$ ) a kombináláskor?



- $T_0$  kimenete (legalább 1) sorozat, akkor  
 $T_0$  : másolás, kiválogatás (halmazos tételek, rendezés)  
 $T_i$  : bármely tétel

# Tételek összeépítése elé...

Milyen tételek lehetnek „**főszereplők**” ( $T_0$ ) a kombináláskor?



- $T_0$  kimenete logikai érték (is), akkor  
 $T_0$  implementálhat egy tulajdonság-függvényt,  
pl. eldöntés (keresés)  
 $T_i$  : megszámlálás, eldöntés, kiválasztás, keresés,  
feltételes tételek

# Másolással összeépítés

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

Ki:  $y \in H[1..n]$

Ef: -

Uf:  $\forall i \in [1..n]: (y[i] = f(x[i]))$

Rövidítve:

Uf:  $y = \text{MÁSOL}(i = e..u, f(x[i]))$

A **másolás** programozási tétellel összeépítés minden programozási tételre működik. (sorozat  $\rightarrow$  sorozat)

Csupán annyi a teendő, hogy a bemenetben szereplő sorozatértékek helyett az  $i$ -edik feldolgozandó elemként a másolásban szereplő  $f$ -transzformáltat kell írni. Például:

Összegzéssel összeépítés:

$\text{SZUMMA}(i=1..n, x[i]) \rightarrow \text{SZUMMA}(i=1..n, f(x[i]))$  vagy

Maximumkiválasztással összeépítés:

$\text{MAX}(i=1..n, x[i]) \rightarrow \text{MAX}(i=1..n, f(x[i]))$

Itt tehát a „másik” tétel bemeneti sorozatára vonatkozik az  $f$ -transzformálás.

# Másolással összeépítés

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

Ki:  $y \in H[1..n]$

Ef: -

Uf:  $\forall i \in [1..n]: (y[i] = f(x[i]))$

Rövidítve:

Uf:  $y = \text{MÁSOL}(i = e..u, f(x[i]))$

A **másolás** programozási tétellel összeépítés minden programozási tételre működik. (sorozat  $\rightarrow$  sorozat)

Csupán annyi a teendő, hogy a kimenetben szereplő sorozatértékek helyett az  $i$ -edik feldolgozandó elemként a másolásban szereplő  $f$ -transzformáltat kell írni. Például:

Kiválogatással összeépítés:

$\text{KIVÁLOGAT}(i=1..n, T(x[i]), x[i]) \rightarrow \text{KIVÁLOGAT}(i=1..n, T(x[i]), f(x[i]))$

Itt tehát a „másik” tétel kimeneti sorozatára vonatkozik az  $f$ -transzformálás.



# másolás+összegzés

---

**Feladat:** határozzuk meg az  $x$  sorozat elemei  $f$  transzformáltjainak az összegét !

**Megoldás** alapja az összegzés tétel.

**Kérdés:** Hogyan látható be a

$$\text{SZUMMA}(i=1..n, x[i]) \rightarrow \text{SZUMMA}(i=1..n, f(x[i]))$$

formula-átalakítás helyessége?

# másolás+összegzés

**Feladat:** határozzuk meg az  $x$  sorozat elemei  $f$  transzformáltjainak az összegét!

$H1, H2$  valamely számhalmaz

Be:  $n \in \mathbb{N}$ ,  $x \in H1[1..n]$

Fv:  $f: H1 \rightarrow H2$

Ki:  $s \in H2$

Ef: -

Uf:  $s = \text{SZUMMA}(i=1..n, f(x[i]))$

# másolás+összegzés

---

**Feladat:** határozzuk meg az  $x$  sorozat elemei  $f$  transzformáltjainak az összegét !

Visszavezetjük a másolás és az összegzés tétel egymásutánjára:

Be:  $n \in \mathbb{N}$ ,  $x \in H1[1..n]$

Sa:  $y \in H2[1..n]$

Fv:  $f: H1 \rightarrow H2$

Ki:  $s \in H2$

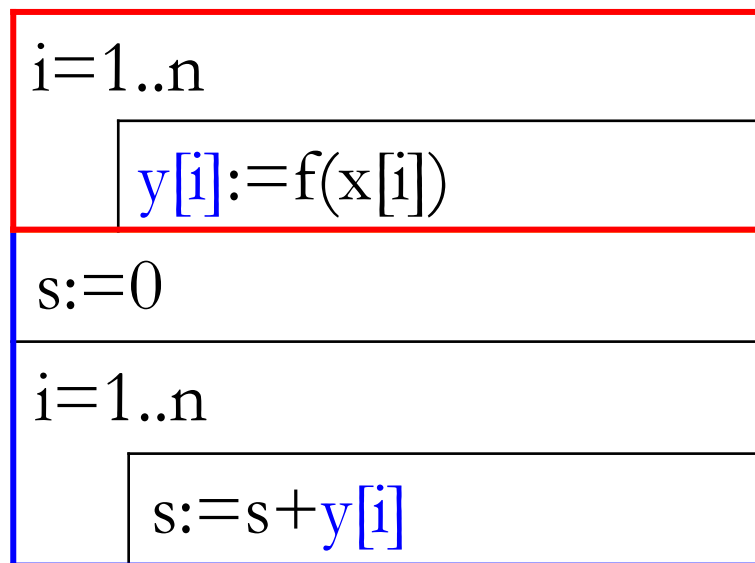
Ef: -

Uf:  $y = \text{MÁSOL}(i=1..n, f(x[i]))$  és  
 $s = \text{SZUMMA}(i=1..n, y[i])$

# másolás+összegzés

## Algoritmus:

Uf:  $y = \text{MÁSOL}(i=1..n, f(x[i]))$  és  
 $s = \text{SZUMMA}(i=1..n, y[i])$

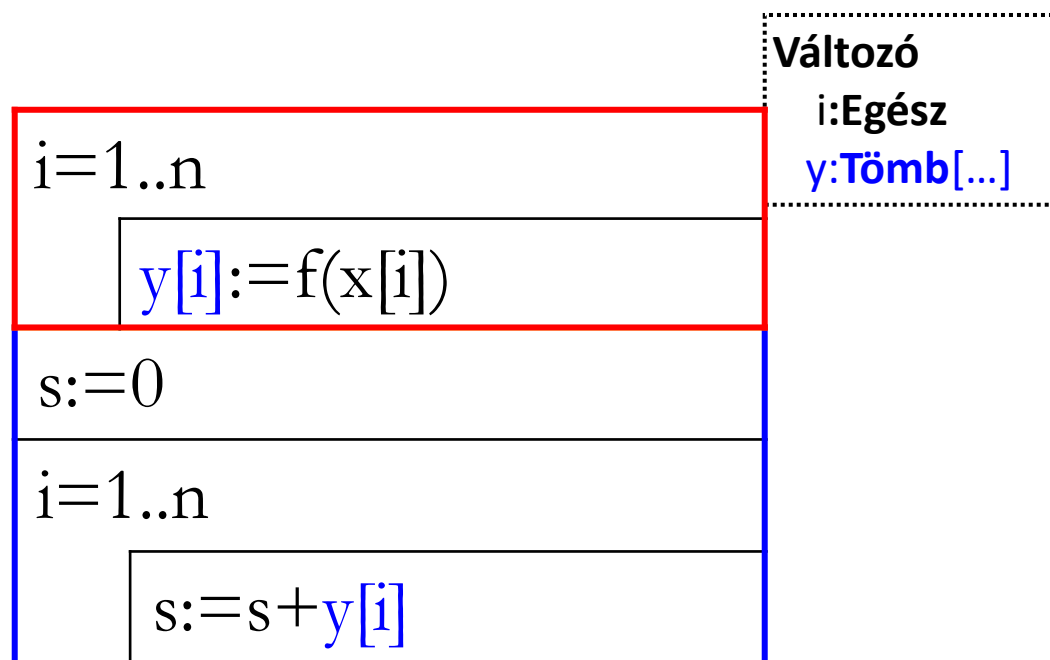


Változó  
 $i$ : Egész  
 $y$ : Tömb[...]

Észrevétel:  
Az eredmény helyes,  
de bántóan nem  
hatékony.

# másolás+összegzés

Az algoritmust programtranszformációkkal alakítsuk át!



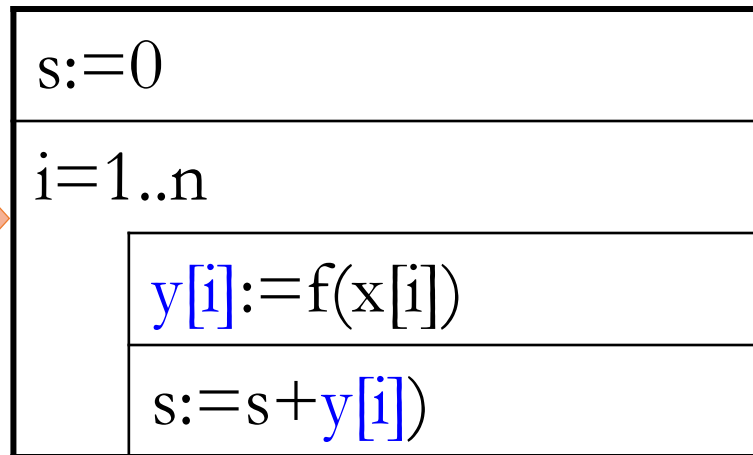
Észrevételek:

1. Azonos ciklus-szervezés.
2. A 2. ciklus  $i$ . lépésben csak az  $y[i]$  kell.

# másolás+összegzés

## 1. programtranszformáció: (gyenge függésű) ciklusok összevonása

Ekvivalens  
átalakítás



Változó  
 $i$ :Egész  
 $y$ :Tömb[...]

Észrevétel:

Az  $y[i]$  érték-  
„kapása” és  
felhasználása  
közvetlen egymás  
mellé került.

# másolás+összegzés

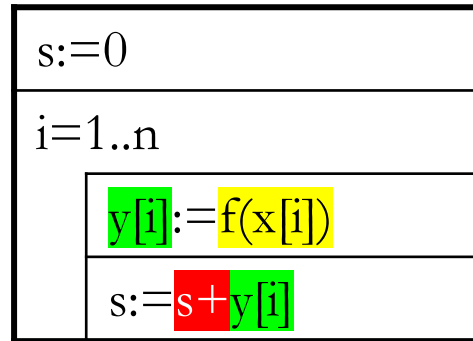
## 2. programtranszformáció: függvénykompozíció

Észrevétel:

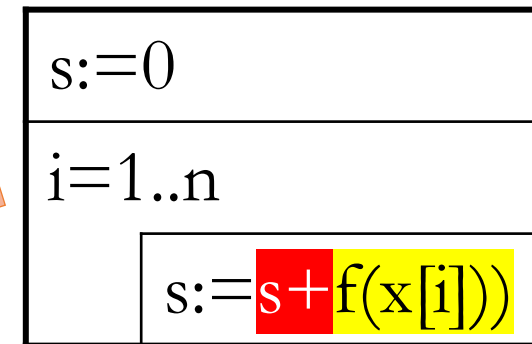
$B := g(A); C := f(B)$

$\leftrightarrow$

$C := f(g(A))$



Ekvivalens  
átalakítás



Változó  
 $i$ :Egész

Végülis beláttuk:  $SZUMMA(i=1..n, f(x[i]))$

# kiválogatás+összegzés

**Feladat:** adott tulajdonságúak összege (feltételes összegzés).

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

Ki:  $s \in H$

Ef: -

Uf:  $s = \text{SZUMMA}(i=1..n, x[i], T(x[i]))$

Visszavezetjük a kiválogatás és az összegzés tétel egymásutánjára:

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

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

Ki:  $s \in H^2$

Ef: -

Uf:  $(db, y) = \text{KIVÁLOGAT}(i=1..n, T(x[i]), x[i])$  és  
 $s = \text{SZUMMA}(i=1..db, y[i])$



# kiválogatás+összegzés

## Algoritmus:

Uf: ( $db, y$ ) = KIVÁLOGAT( $i=1..n, T(x[i]), x[i]$ ) és  
 $s = SZUMMA(i=1..db, y[i])$

|  |  |
|--|--|
| db:=0  |  |
| i=1..n   |  |
| $\begin{array}{c c} \text{I} & T(x[i]) \\ \hline db:=db+1 & \\ \hline y[db]:=x[i] & \end{array}$ | $\begin{array}{c} \text{N} \\ \hline \text{---} \end{array}$ |
| s:=0   |  |
| i=1..db  |  |
| s:=s+y[i]  |  |

Változó

i,db:Egész

y:Tömb[...]

Észrevétel:

Az eredmény helyes,  
de bántóan nem  
hatékony.

# kiválogatás+összegzés

Az algoritmust programtranszformációkkal alakítsuk át!

Változó

$i, db,$   
 $db1$ :Egész  
 $y$ :Tömb[...]

Észrevétel:

A két különböző  
 szervezésű **ciklus**  
**hasonlóvá** tétele, a  
 második ciklus  
**szemantikus**  
**ekvivalenciájának**  
 biztosítása mellett.

Ekvivalens  
 átalakítás

|              |             |
|--------------|-------------|
| db:=0        |             |
| i=1..n       |             |
| I            | $T(x[i])$   |
|              | db:=db+1    |
|              | y[db]:=x[i] |
| s:=0; db1:=0 |             |
| i=1..n       |             |
| I            | $T(x[i])$   |
|              | db1:=db1+1  |
|              | s:=s+y[db1] |

# kiválogatás+összegzés

Programtranszformáció: (gyengén függő) ciklusok  
összevonása

Ekvivalens  
átalakítás

|                     |   |
|---------------------|---|
| db:=0; s:=0; db1:=0 |   |
| i=1..n              |   |
| I                   | N |
| T(x[i])             |   |
| db:=db+1            | — |
| y[db]:=x[i]         |   |
| I                   | N |
| T(x[i])             |   |
| db1:=db1+1          |   |
| s:=s+y[db1]         |   |

Változó

i,db,

db1:Egész

y:Tömb[...]

# kiválogatás+összegzés

Észrevétel:

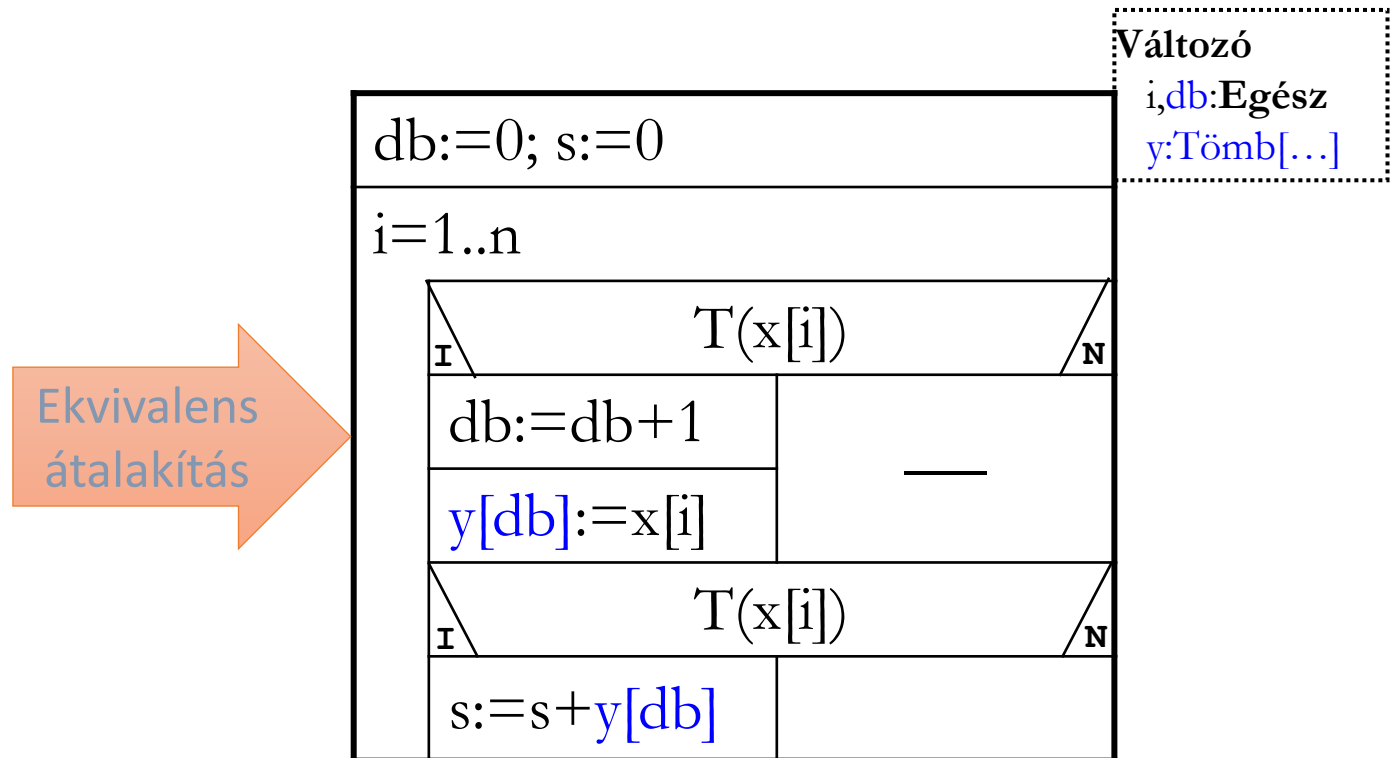
A db és db1 változók szinkronban változnak, minden cikluslépésben azonos értékűek. Így a db1 változó elhagyható, a rá vonatkozó értékadások elhagyhatók, az y[db1] helyett y[db] írandó.

Ekvivalens  
átalakítás

|                     |         |     |
|---------------------|---------|-----|
| db:=0; s:=0; db1:=0 |         |     |
| i=1..n              |         |     |
| I \                 | T(x[i]) | / N |
| db:=db+1            | —       |     |
| y[db]:=x[i]         |         |     |
| I \                 | T(x[i]) | / N |
| db1:=db1+1          |         |     |
| s:=s+y[db]          |         |     |

Változó  
i,db,  
db1:Egész  
y:Tömb[...]

# kiválogatás+összegzés



# kiválogatás+összegzés

## Programtranszformáció: elágazások összevonása

Észrevétel:

2, azonos feltételű  
elágazás összevon-  
ható (mivel a fel-  
tétel paramétere az  
elágazásban nem  
változik meg)

Ekvivalens  
átalakítás

|              |                    |
|--------------|--------------------|
| db:=0; s:=0  |                    |
| i=1..n       |                    |
| $\mathbf{i}$ | $\mathbf{T(x[i])}$ |
| db:=db+1     | —                  |
| y[db]:=x[i]  |                    |
| s:=s+y[db]   |                    |

Változó  
i,db:Egész  
y:Tömb[...]

# kiválogatás+összegzés

## Programtranszformáció: függvénykompozíció

Észrevétel:

$B := g(A); C := f(B)$

$\leftrightarrow$

$C := f(g(A))$

$\rightarrow$

$y$  és  $db$   
elhagyható

|             |         |
|-------------|---------|
| db:=0; s:=0 |         |
| i=1..n      |         |
| I           | T(x[i]) |
| db:=db+1    | —       |
| y[db]:=x[i] |         |
| s:=s+y[db]  |         |

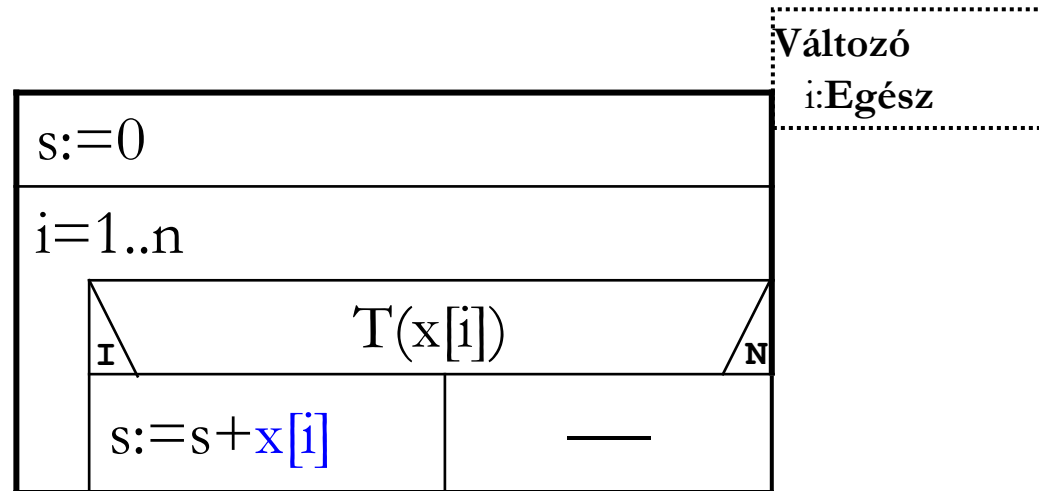
Ekvivalens  
átalakítás

| s:=0   |         |         |   |  |   |  |           |  |
|--|---------|---------|---|--|---|--|-----------|--|
| i=1..n   |         |         |   |  |   |  |           |  |
| <table><tr><th>I</th><th>T(x[i])</th><th>N</th></tr><tr><td></td><td rowspan="2">—</td><td rowspan="2"></td></tr><tr><td>s:=s+x[i]</td></tr></table> | I       | T(x[i]) | N |  | — |  | s:=s+x[i] |  |
| I  | T(x[i]) | N       |   |  |   |  |           |  |
|  | —       |         |   |  |   |  |           |  |
| s:=s+x[i]  |         |         |   |  |   |  |           |  |

Változó  
i :Egész

# kiválogatás+összegzés

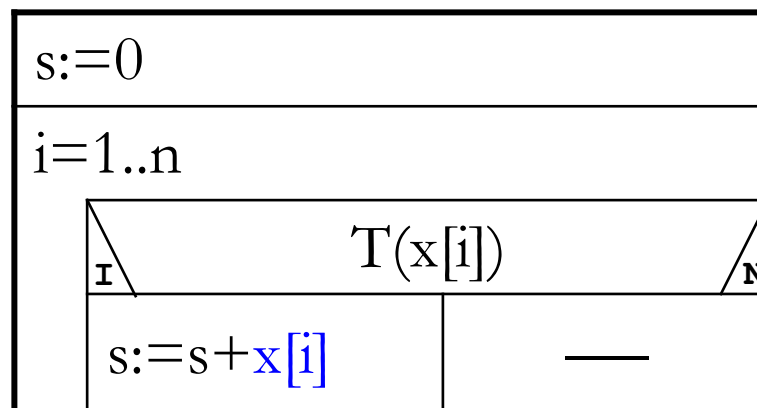
Végülis beláttuk:  $s = \text{SZUMMA}(i=1..n, x[i], T(x[i]))$





# kiválogatás+összegzés helyességbizonyítással

Algoritmikus gondolkodással: kiválogatás nélkül **azonnal adjuk össze** a megfelelő elemeket!



Változó  
i:Egész

Ez esetben bizonyítanunk kell a helyességet!

Bepillantunk a programozási tételek  
bizonyításának módszertanába is.

# kiválogatás+összegzés helyességbizonyítással

Helyességbizonyítás: az algoritmus kielégíti-e a specifikációt?

Uf:  $s = \text{SZUMMA}(i=1..n, x[i], T(x[i]))$

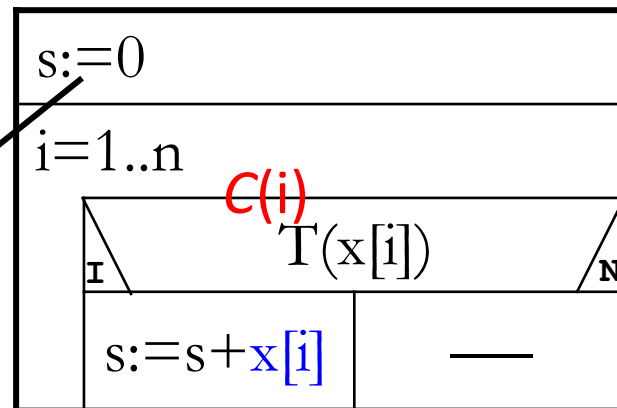
Ciklusinvariáns ( $C(i)$ ) állítás, a ciklus-  
magba belépéskor kiértékelendő:

$s = \text{SZUMMA}(j=1..i-1, x[j], T(x[j]))$

## Indukciós bizonyítás:

Ciklusba belépéskor ( $i=1$ ):

$$\begin{aligned} C(1) = & s = \text{SZUMMA}(j=1..i-1, x[j], T(x[j])) \\ & \text{SZUMMA}(j=1..1-1, x[j], T(x[j])) = 0 \\ & s = 0 \\ & 0 = 0 \end{aligned}$$



Változó  
 $i$ : Egész



# kiválogatás+összegzés helyességbizonyítással

Helyességbizonyítás: az algoritmus kielégíti-e a specifikációt?

Uf:  $s = \text{SZUMMA}(i=1..n, x[i], T(x[i]))$

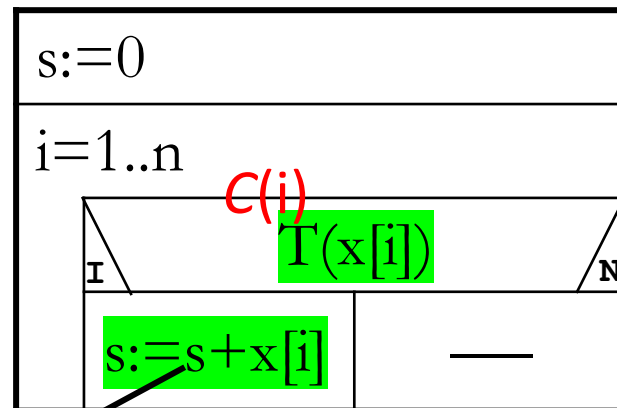
Ciklusinvariáns ( $C(i)$ ) állítás, a ciklusmagba belépéskor kiértékelendő:

$s = \text{SZUMMA}(j=1..i-1, x[j], T(x[j]))$

**Indukciós lépés:**

Ciklusmag egyszeri végrehajtása után ( $i \rightarrow i+1$ ):

$C(i)$  és  $T(x[i]) \rightarrow s := s + x[i] \rightarrow$   
 $s = \text{SZUMMA}(j=1..i-1, x[j], T(x[j])) + x[i] =$   
 $\text{SZUMMA}(j=1..i, x[j], T(x[j])) =$   
 $C(i+1)$



Változó  
 $i$ : Egész



# kiválogatás+összegzés helyességbizonyítással

Helyességbizonyítás: az algoritmus kielégíti-e a specifikációt?

Uf:  $s = \text{SZUMMA}(i=1..n, x[i], T(x[i]))$

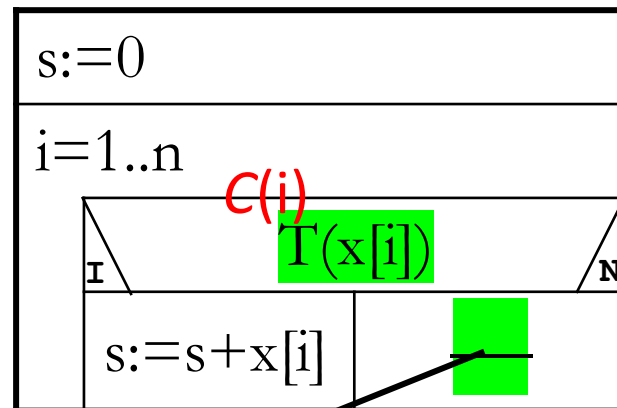
Ciklusinvariáns ( $C(i)$ ) állítás, a ciklusmagba belépéskor kiértékelendő:

$s = \text{SZUMMA}(j=1..i-1, x[j], T(x[j]))$

## Indukciós lépés:

Ciklusmag egyszeri végrehajtása után ( $i \rightarrow i+1$ ):

$C(i)$  és  $\text{nem } T(x[i]) \rightarrow s := s + 0 \rightarrow$   
 $s = \text{SZUMMA}(j=1..i-1, x[j], T(x[j])) + 0 =$   
 $\text{SZUMMA}(j=1..i, x[j], T(x[j])) =$   
 $C(i+1)$



Változó  
 $i$ : Egész



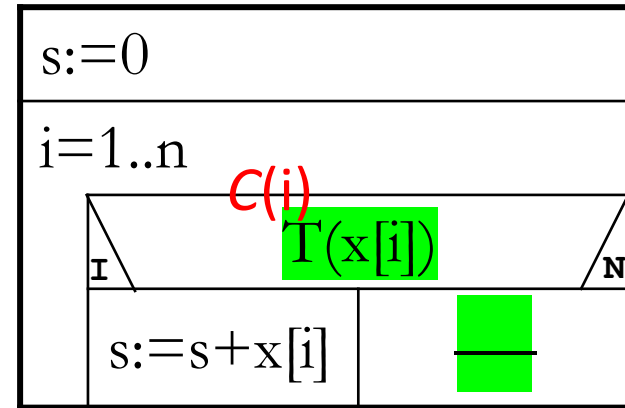
# kiválogatás+összegzés helyességbizonyítással

Helyességbizonyítás: az algoritmus kielégíti-e a specifikációt?

Uf:  $s = \text{SZUMMA}(i=1..n, x[i], T(x[i]))$

Ciklusinvariáns ( $C(i)$ ) állítás, a ciklus-  
magba belépéskor kiértékelendő:

$s = \text{SZUMMA}(j=1..i-1, x[j], T(x[j]))$



Változó  
 $i$ : Egész

Ciklusból kilépéskor ( $i \rightarrow n+1$ ):

$$\begin{aligned}
 C(n+1) &= \\
 s &= \text{SZUMMA}(j=1..i-1, x[j], T(x[j])) + 0 = \\
 &\text{SZUMMA}(j=1..n, x[j], T(x[j])) = \\
 &Uf
 \end{aligned}$$



# Maximumkiválasztás+kiválogatás

---

**Feladat:** összes maximális elem kiválogatása.

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

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

Sa:  $maxért \in H$

Fv:  $legnagyobb: H \rightarrow L$ ,  $legnagyobb(h) = h = maxért$

Ef:  $n > 0$

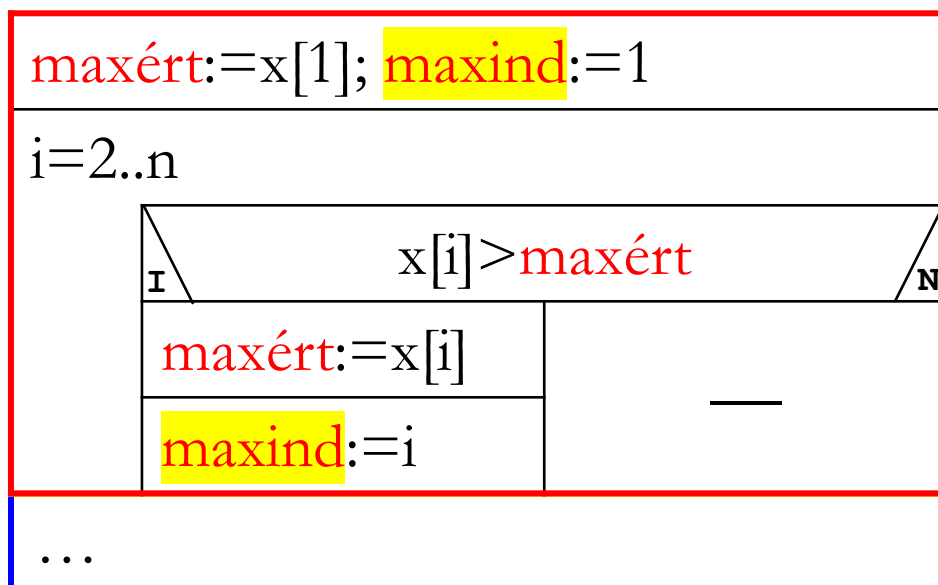
Uf:  $(, maxért) = \text{MAX}(i=1..n, x[i])$  és

$(db, maxI) = \text{KIVÁLOGAT}(i=1..n, legnagyobb(x[i]), i)$

# Maximumkiválasztás+kiválogatás

## Algoritmus:

Uf:  $(, \text{maxért}) = \text{MAX}(i=1..n, x[i])$  és  
 $(\text{db}, \text{maxI}) = \text{KIVÁLOGAT}(i=1..n, \text{legnagyobb}(x[i]), i)$



Változó

$i, \text{maxind}$ : Egész

$\text{maxért}$ : TH

# Maximumkiválasztás+kiválogatás

## Algoritmus:

Uf:  $(, \text{maxért}) = \text{MAX}(i=1..n, x[i])$  és  
 $(\text{db}, \text{maxI}) = \text{KIVÁLOGAT}(i=1..n, \text{legnagyobb}(x[i]), i)$

|             |                  |
|-------------|------------------|
| ...         |                  |
| db:=0       |                  |
| i=1..n      |                  |
| I           | legnagyobb(x[i]) |
| N           |                  |
| db:=db+1    | —                |
| maxI[db]:=i |                  |

Változó

i, maxind: Egész

maxért: TH



# Maximumkiválasztás+kiválogatás

## Algoritmus:

Változó

$i, \text{maxind}$ : Egész

$\text{maxért}$ : TH

$\text{maxért} := x[1]; \text{maxind} := 1$

$i = 2..n$

| I                       | $x[i] > \text{maxért}$ | N |
|-------------------------|------------------------|---|
| $\text{maxért} := x[i]$ | —                      |   |
| $\text{maxind} := i$    |                        |   |

$\text{db} := 0$

$i = 1..n$

| I                             | $\text{legnagyobb}(x[i])$ | N |
|-------------------------------|---------------------------|---|
| $\text{db} := \text{db} + 1$  | —                         |   |
| $\text{maxI}[\text{db}] := i$ |                           |   |

Észrevétel:

Az eredmény helyes,  
de bántóan nem  
hatékony.

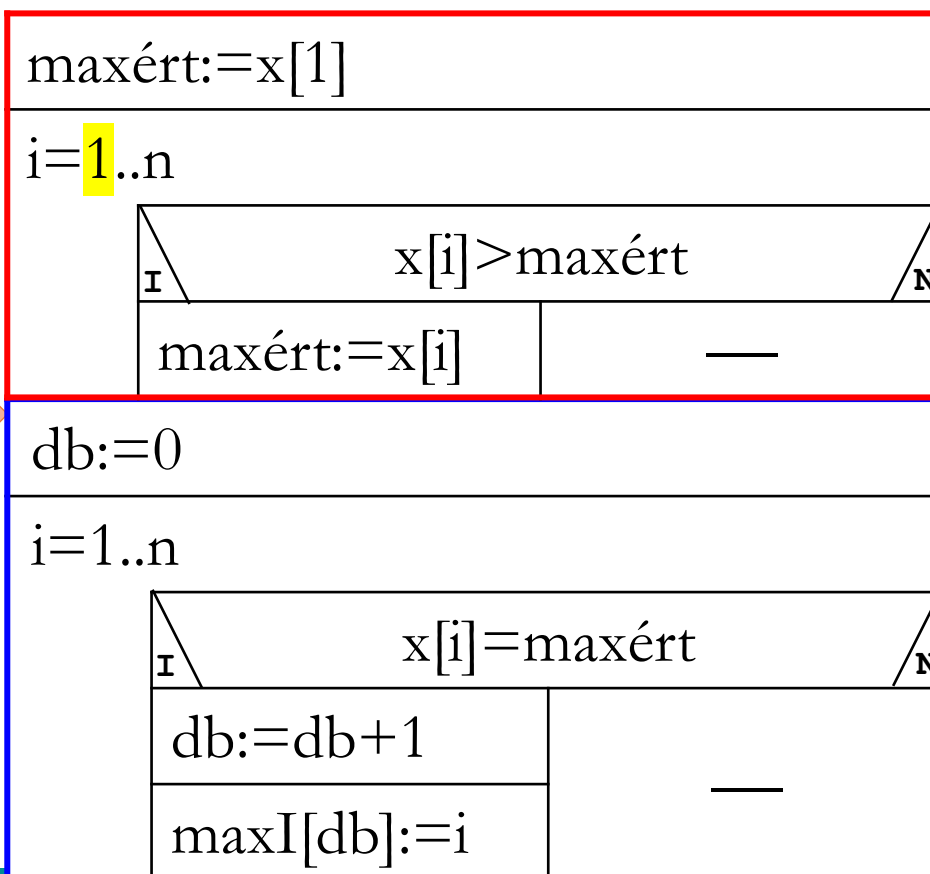
# Maximumkiválasztás+kiválogatás

Az algoritmust programtranszformációkkal alakítsuk át!

Észrevétel:

- A fölösleges maxind változót hagyjuk el!
- Függvénytörzset helyezük a hívás helyére!
- Hozzuk **szinkronba** a ciklus-szerve-zéseket:  $i=1..n$ .

Ekvivalens  
átalakítás



Változó  
i:Egész  
maxért:TH

# Maximumkiválasztás+kiválogatás

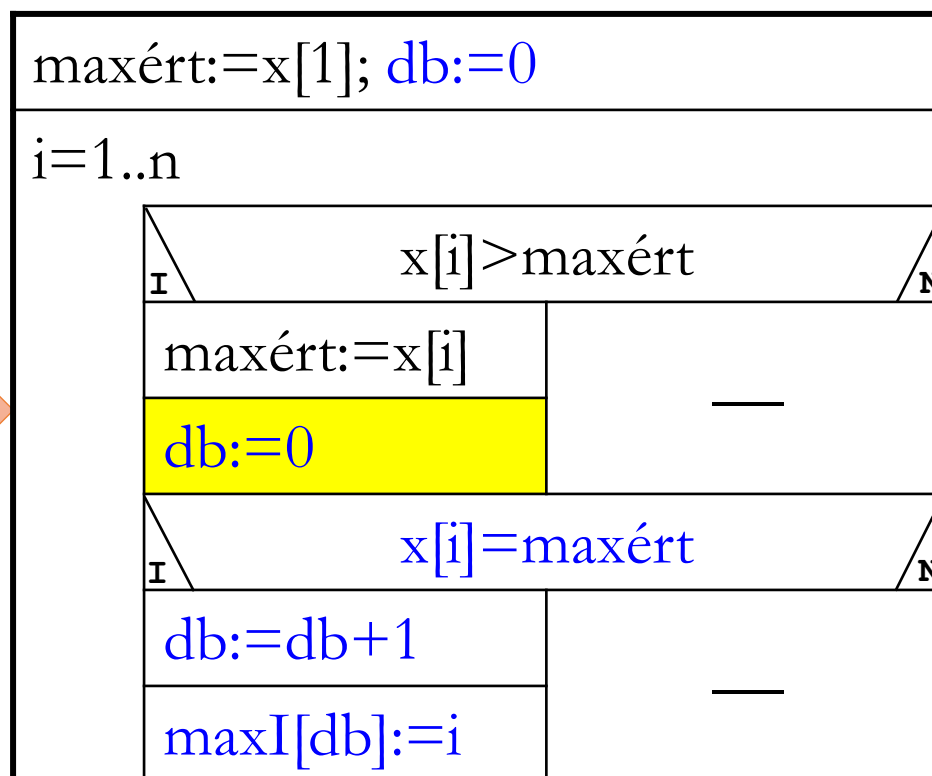
Programtranszformáció: ciklusok és elágazások összevonása

Észrevétel:

Az összevonás csak így lehetséges:

- ha a maxért megváltozik, akkor db nullázandó,
- a 2. feltételvizsgálat ez esetben igaz lesz, és az 1. max helye feljegyződik.

Ekvivalens  
átalakítás



Változó

i:Egész

maxért:TH

# Maximumkiválasztás+kiválogatás

Programtranszformáció: kizáró feltételű elágazások összevonása

Észrevétel:  
A program-  
transzformáció  
függetlensége  
feltétele nem  
teljesül, de  
ötletnek jó.  
Ez esetben az új  
maxért-et  
elsőként fel is  
kell jegyezni.

Ekvivalens  
átalakítás

|                        |                        |
|------------------------|------------------------|
| maxért:=x[1]; db:=0    |                        |
| i=1..n                 |                        |
| $x[i] > \text{maxért}$ | $x[i] = \text{maxért}$ |
| maxért:=x[i]           | db:=db+1               |
| db:=1                  | maxI[db]:=i            |
| maxI[db]:=i            |                        |

Változó

i:Egész

maxért:TH

# Maximumkiválasztás+kiválogatás

Észrevétel:  
A ciklus indítható  
2-től is „okos”  
inicializálások  
után.

Ekvivalens  
átalakítás

|                                  |                        |
|----------------------------------|------------------------|
| maxért:=x[1]; db:=1; maxI[db]:=1 |                        |
| i=2..n                           |                        |
| $x[i] > \text{maxért}$           | $x[i] = \text{maxért}$ |
| maxért:=x[i]                     | db:=db+1               |
| db:=1                            | maxI[db]:=i            |
| maxI[db]:=i                      |                        |

Változó  
i:Egész  
maxért:TH

# Eldöntés+megszámolás

**Feladat:** Van-e egy sorozatban legalább  $k$  darab adott tulajdonságú elem?

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

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

Sa:  $db \in \mathbb{N}$

Ef:  $k > 0$

Uf:  $db = \text{DARAB}(i=1..n, T(x[i]))$  és  $\text{van} = db \geq k$

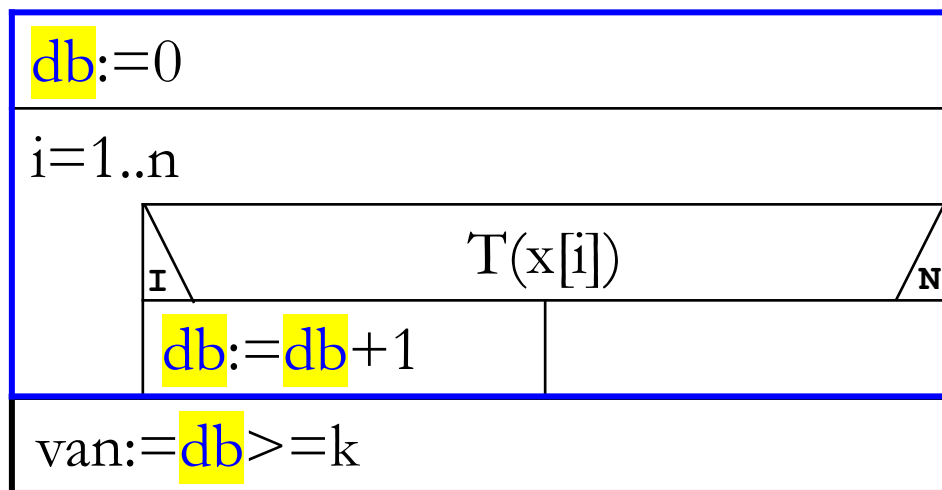
megszámolás

~~eldöntés~~

# Eldöntés+megszámolás

## Algoritmus:

Uf:  $db = \text{DARAB}(i=1..n, T(x[i]))$  és  $van = db \geq k$



Változó

$i, db$ : Egész

Észrevétel:

Helyes, de nem hatékony megoldás!

Ha már **találtunk** **K** **darab** adott tulajdonságút, akkor **ne nézzük tovább!**

# Eldöntés+megszámolás

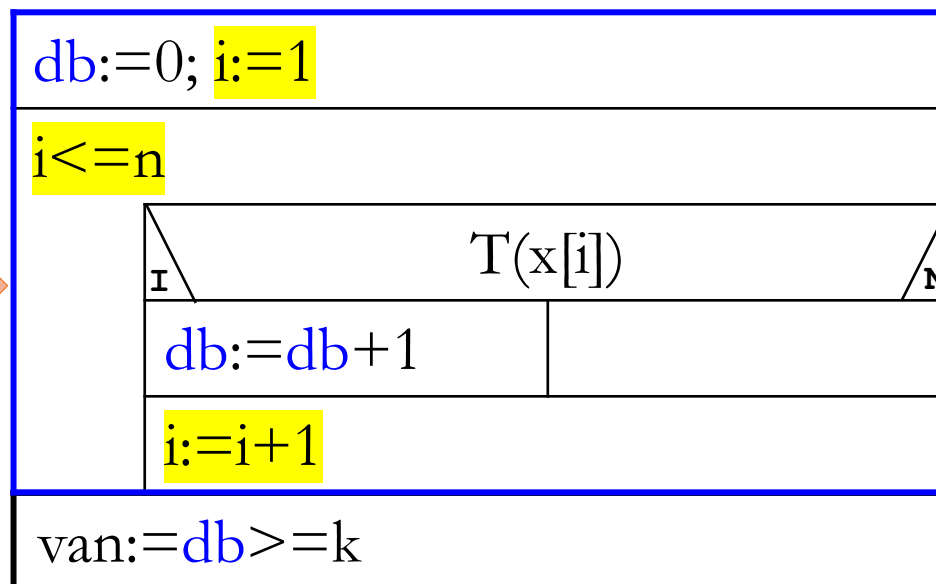
Az algoritmust programtranszformációkkal alakítsuk át!

Észrevétel:

Teremtsünk lehetőséget arra, hogy „időben” kiléphessünk a ciklusból!

Alakítsuk át a számlálós ciklust feltételes ciklussá!

Ekvivalens átalakítás



Változó  
i,db:Egész

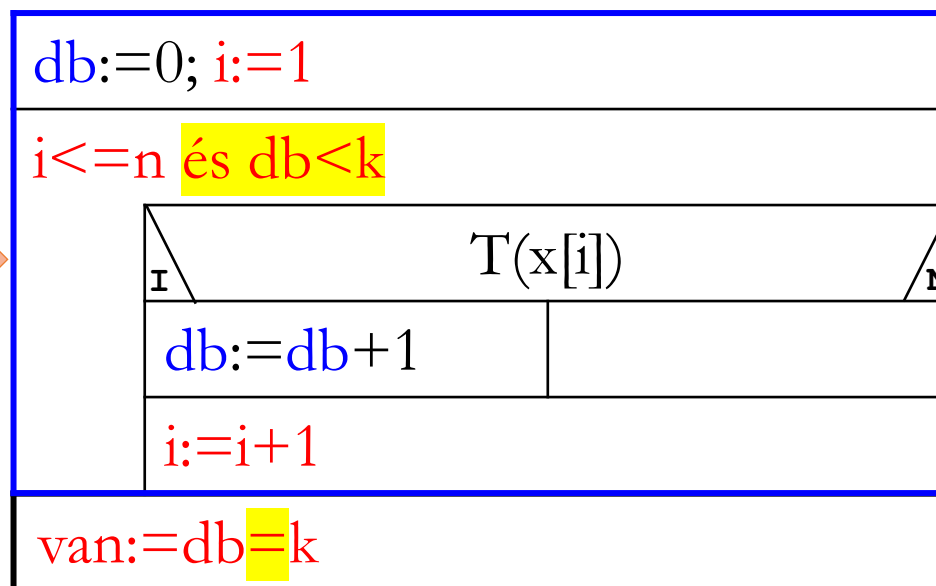


# Eldöntés+megszámolás

Észrevétel:

Bővítsük a ciklusfeltételt a kívánttal!

Ekvivalens  
átalakítás



Változó  
i,db:Egész

Megjegyzés: ehhez „illeszkedő” utófeltétel:

Uf: van=VAN(i=1..n, DARAB(j=1..i, T(x[j])=k)).

Igaz, ebből is csak programtranszformációkkal nyerhető a fenti algoritmus.

# Összefoglalás



# Összefoglalás

---

- Több programozási minta használata
  - bizonyos feladatok megoldásához több programozási minta használata szükséges
  - ezek egy része, amikor a mintákat egymás után használjuk
  - közbülső segédadatok
  - hatékonyság programtranszformációkkal