ELTE | IK

PROGRAMOZÁS

Rendezések

Horváth Győző, Horváth Gyula, Szlávi Péter

Rendezési feladat

Specifikáció:

```
Be: n∈N, x∈H[1..n],
    <=:H x H->L
Ki: y∈H[1..n]
Ef: -
Uf: RendezettE(y) és y∈Permutáció(x)
```

Rendezés(≤) és $\text{RendezettE}_{\leq}$(H)

Jelölések:

- $\text{RendezettE}_{\leq}$(X/H): X/H rendezett-e a ≤-ra?
- Y∈Permutáció(X): Y az X elemeinek egy permutációja-e?

Rendezési feladat

A rendezések egy részében olyan megvalósítást választunk, amiben a bemenetnek és a kimenetnek ugyanaz a sorozat felel meg, azaz helyben rendezünk.

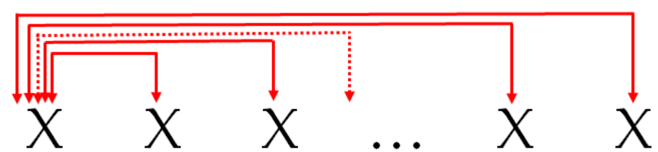
Specifikáció:

```
Be: n∈N, x∈H[1..n],
    <=:H x H->L
Ki: x'∈H[1..n]
Ef: -
Uf: RendezettE(x') és x'∈Permutáció(x)
```

Egyszerű cserés rendezés

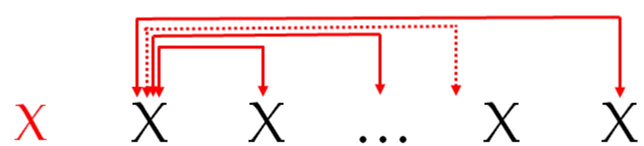
A lényeg:

- Hasonlítsuk az első elemet az összes mögötte levővel, s ha kell, cseréljük meg!



A minimum az „alsó" végére kerül.

- Ezután ugyanezt csináljuk a második elemre!
- …

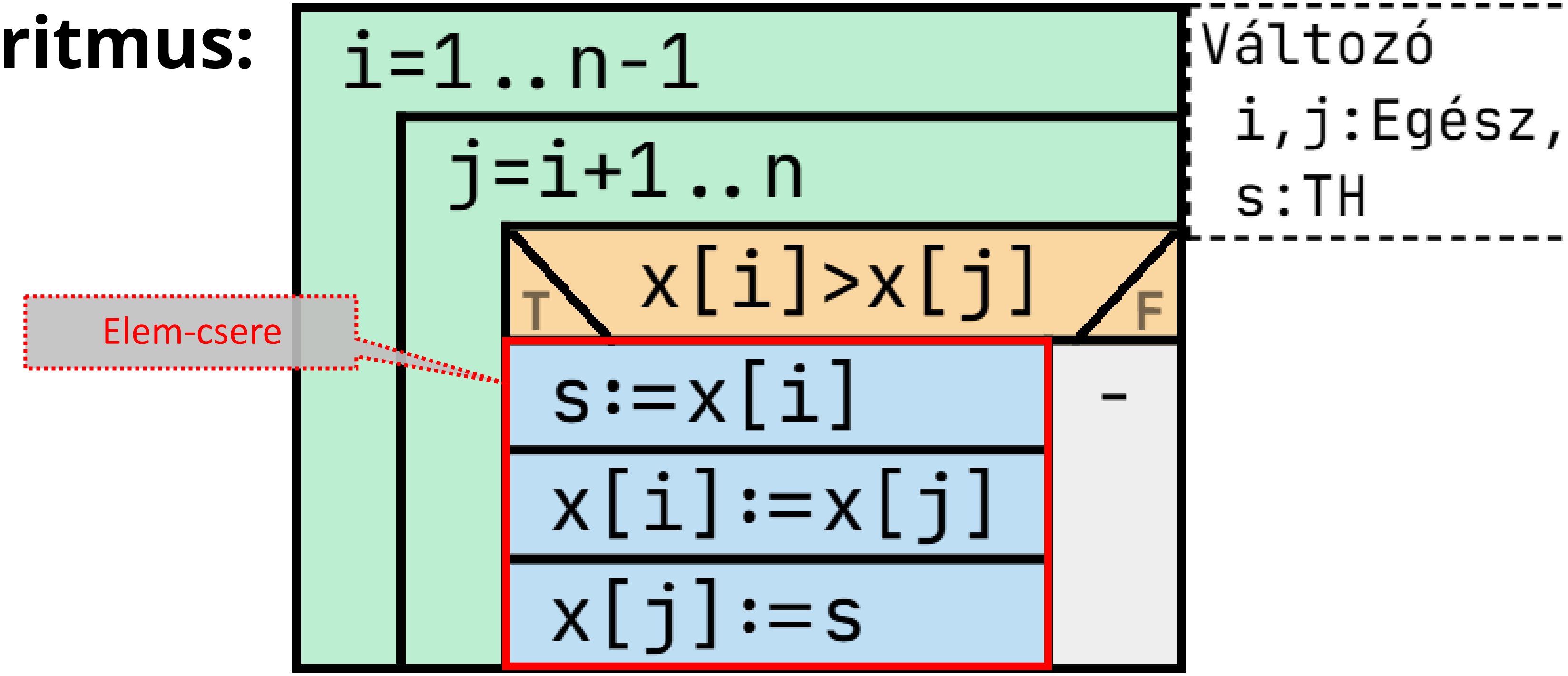


- Végül az utolsó két elemre!

A pirossal jelöltek már a helyükön vannak

Egyszerű cserés rendezés

Algoritmus:

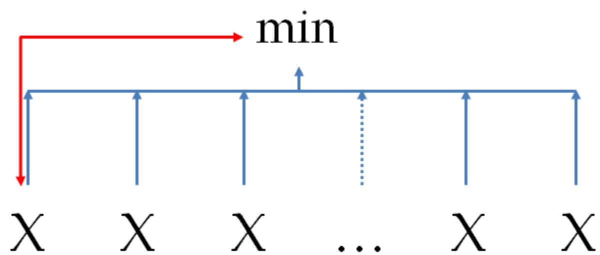


- Hasonlítások száma: 1+2+…+N–1= $N \cdot \frac{N-1}{2}$
- Mozgatások száma: 0 .. $3 \cdot N \cdot \frac{N-1}{2}$

Minimumkiválasztásos rendezés

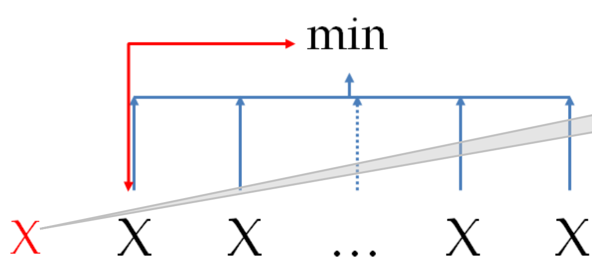
A lényeg:

- Határozzuk meg az 1..N elemek minimumát, s cseréljük meg az 1.-vel!



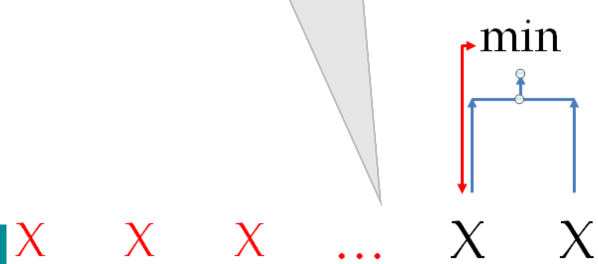
A minimum az „alsó” végére kerül.

- Ezután ugyanezt tegyük a 2..N elemre!
- …



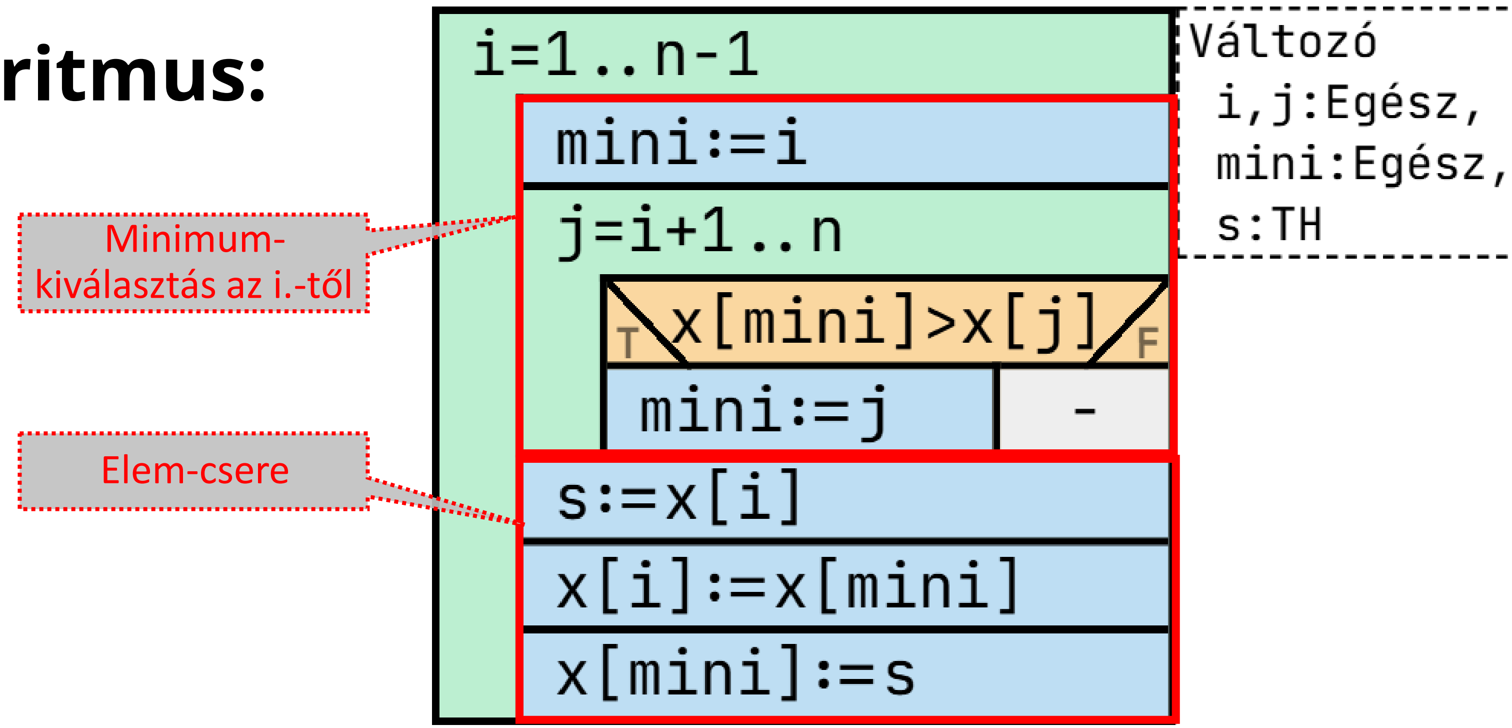
A pirossal jelöltek már a helyükön vannak

- Végül az utolsó két (N–1..N) elemre!



Minimumkiválasztásos rendezés

Algoritmus:

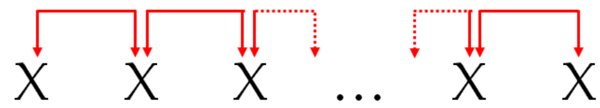


- Hasonlítások száma: 1+2+…+N–1= $N \cdot \frac{N-1}{2}$
- Mozgatások száma: 3*(N–1)

Buborékos rendezés

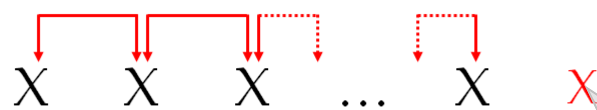
A lényeg:

- Hasonlítsunk minden elemet a mögötte levővel, s ha kell, cseréljük meg!



A maximum a „felső” végére kerül.

- Ezután ugyanezt csináljuk az utolsó elem nélkül!
- …



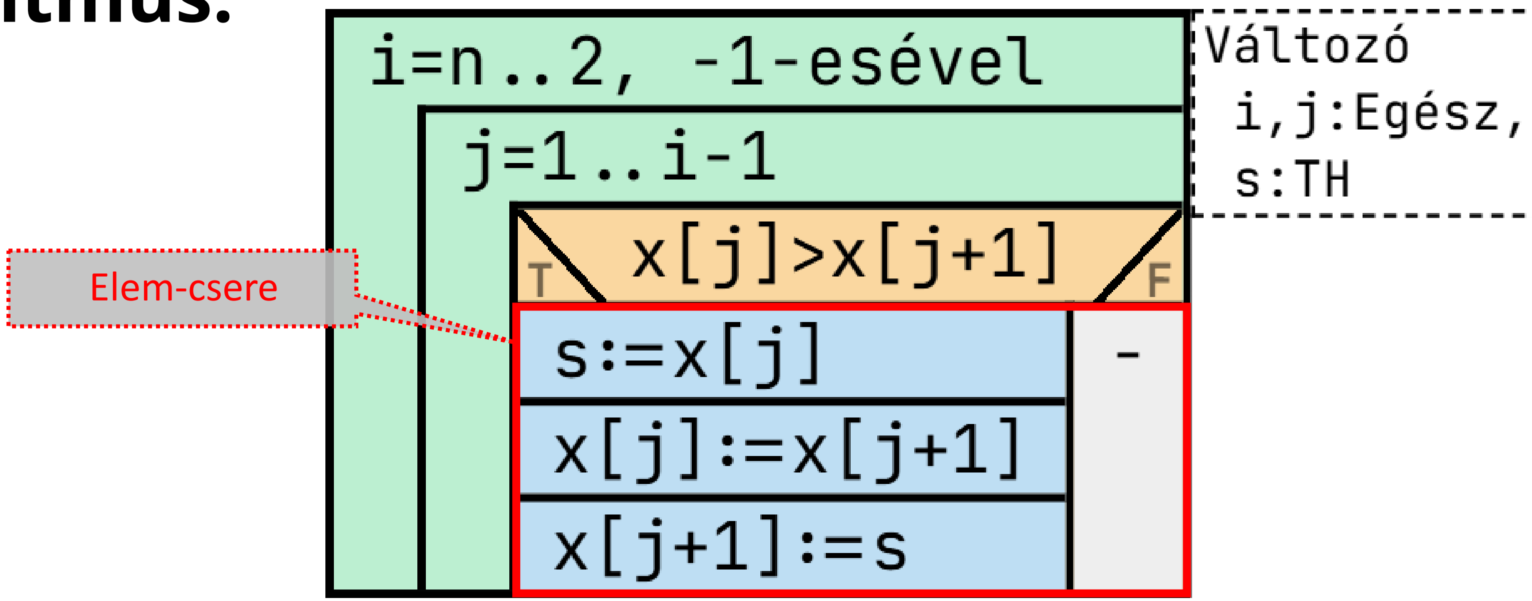
A többiek is tartanak a helyük felé.

- Végül az első két elemre!

A pirossal jelöltek már a helyükön vannak

Buborékos rendezés

Algoritmus:

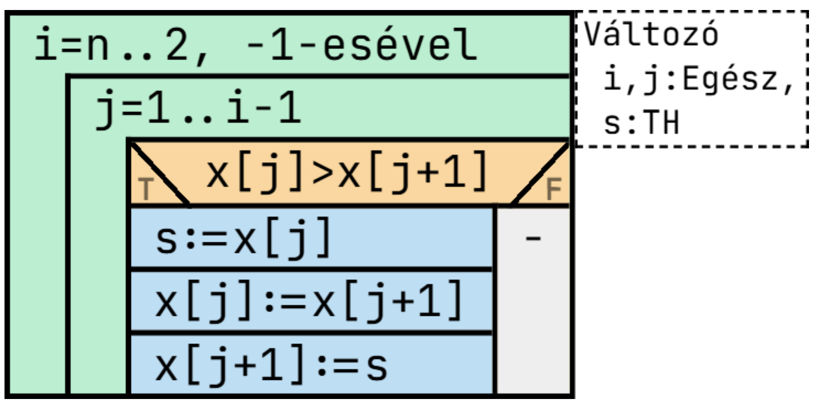


- Hasonlítások száma: 1+2+...+N–1= $N \cdot \frac{N-1}{2}$
- Mozgatások száma: 0 .. $3 \cdot N \cdot \frac{N-1}{2}$

Javított buborékos rendezés

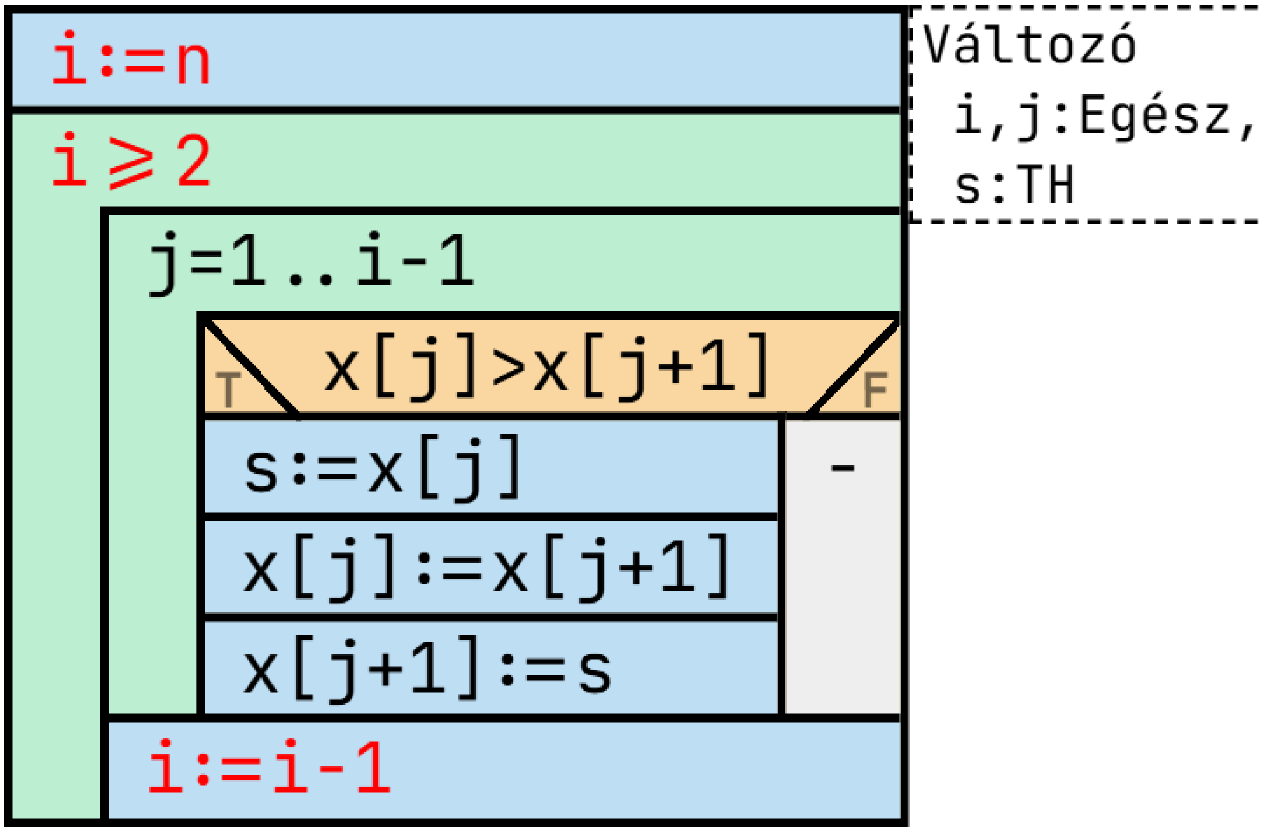
Megfigyelések:

- Ha a belső ciklusban egyáltalán nincs csere, akkor be lehetne fejezni a rendezést.
- Ha a belső ciklusban a K. helyen van az utolsó csere, akkor a K+1. helytől már biztosan jó elemek vannak, a külső ciklus-változóval többet is léphetnénk.

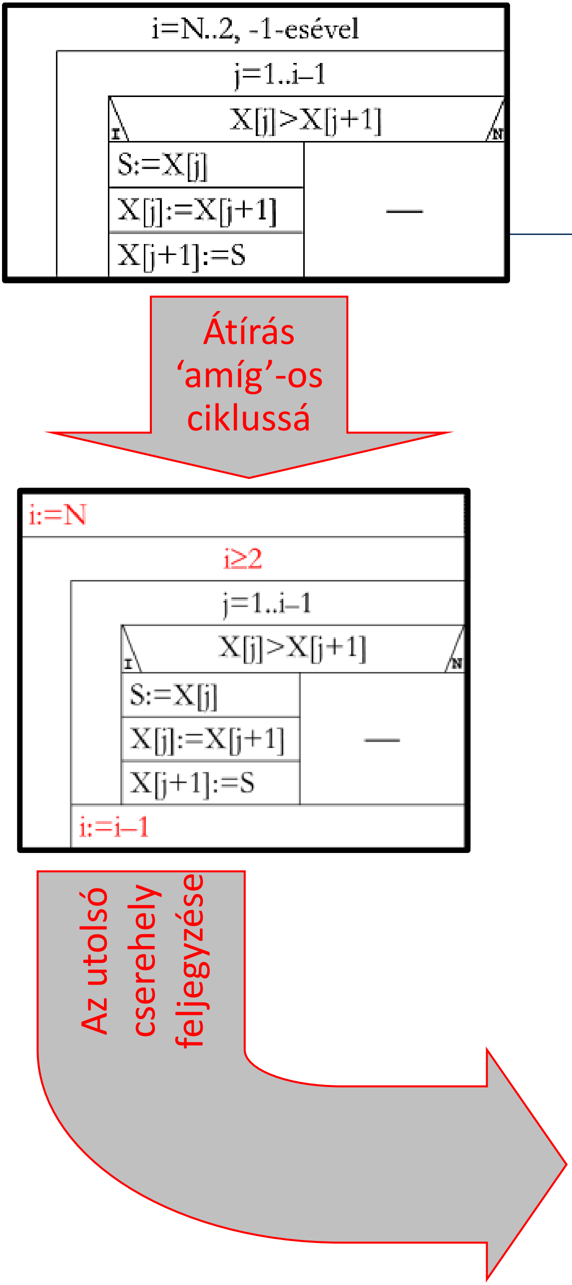


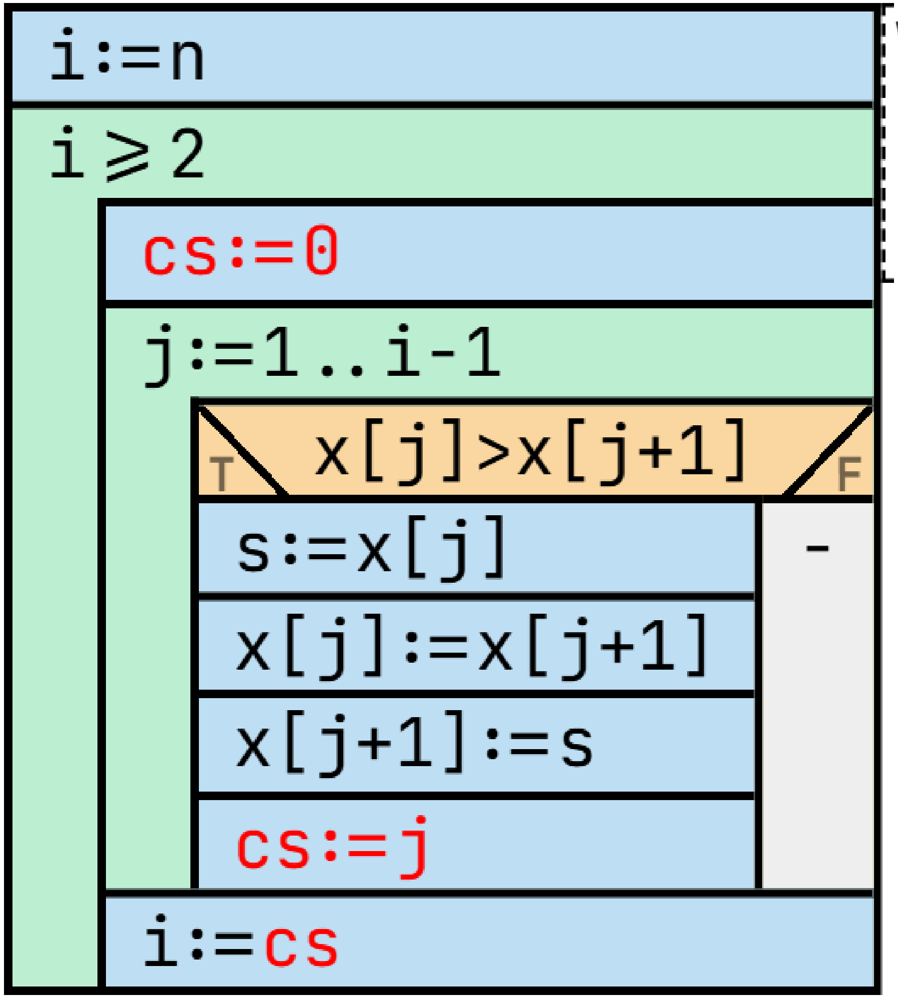
Javított buborékos rendezés

Algoritmus: (átalakítva feltételes ciklusúvá)



Javított buborékos rendezés





Változó
i,j:Egész,
cs:Egész,
s:TH

Beillesztéses rendezés

A lényeg:

- *Egy* elem *rendezett*.

- A másodikat vagy mögé, vagy elé tesszük, így már *ketten* is *rendezettek*.

- …

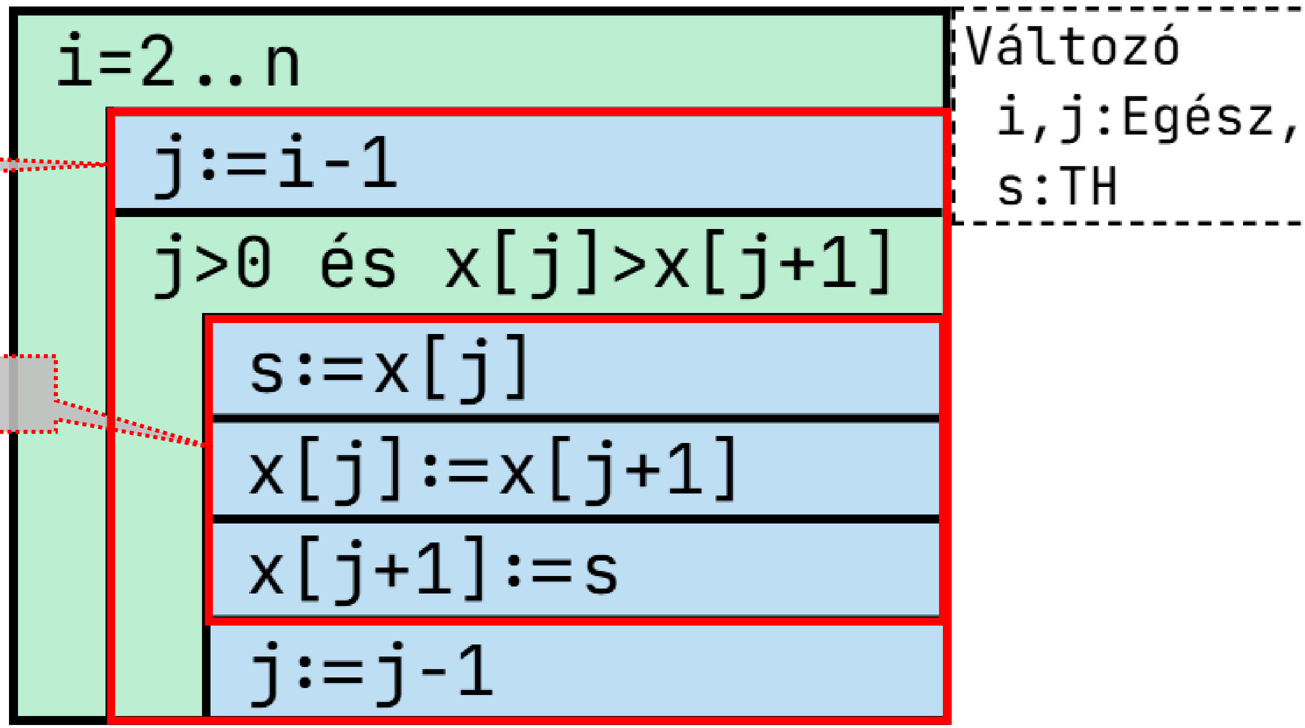
- Az i-ediket a kezdő, i–1 *rendezett*ben addig hozzuk előre **cserékkel**, amíg a helyére nem kerül; így már *i darab rendezett* lesz.

- …

- Az utolsóval ugyanígy!

Beillesztéses rendezés

Algoritmus:

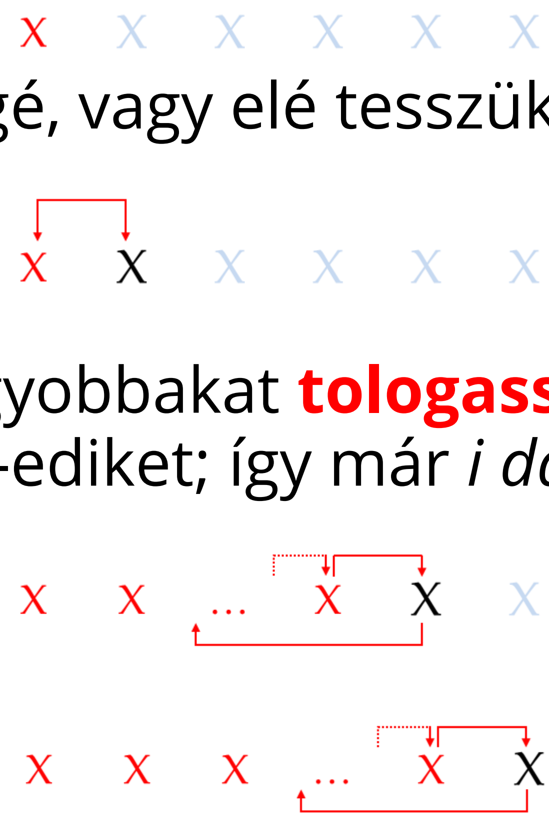


- Hasonlítások száma: N–1 .. $N \cdot \frac{N-1}{2}$
- Mozgatások száma: 0 .. $3 \cdot N \cdot \frac{N-1}{2}$

Javított beillesztéses rendezés

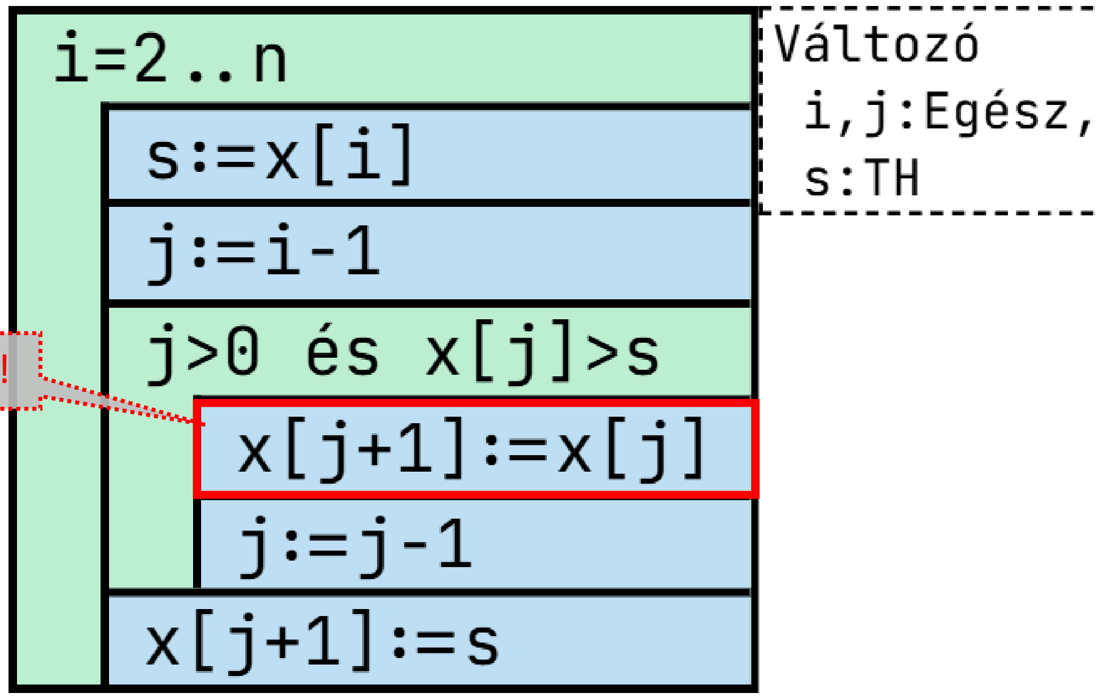
A lényeg:

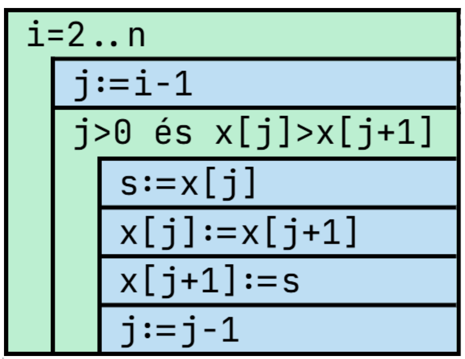
- *Egy* elem *rendezett*.
- A másodikat vagy mögé, vagy elé tesszük, így már *ketten* is *rendezettek*.
- …
- Az i-ediknél a nála nagyobbakat **tologassuk** hátra, majd illesszük be eléjük az i-ediket; így már *i darab rendezett* lesz.
- …
- Az utolsóval ugyanígy!



Javított beillesztéses rendezés

Algoritmus:





Hasonlítások száma: N–1 … $N\cdot\frac{N-1}{2}$

Mozgatások száma: 0 … $3\cdot N\cdot\frac{N-1}{2}$

- Hasonlítások száma: N–1 .. $N\cdot\frac{N-1}{2}$
- Mozgatások száma: 2*(N – 1) .. $(N+4)\cdot\frac{N-1}{2}$

Rendezésvizualizációk

- Inkább hatékonyság szemléltetésére, mint megértésére valók
- Vizualizációk
  - https://www.toptal.com/developers/sorting-algorithms
  - https://www.sortvisualizer.com/selectionsort/
  - https://sorting-algorithm-jet.vercel.app/
  - https://visualize-it.github.io/sorting_algos/simulation.html