# 9. előadás – Összehasonlító rendezések

## Feladat: Adott kulcsos rekord, rendezzük!

Elméletileg csak a kulcsok fontosak, a rekord többi részét elhagyjuk.

Ezek a kulcsok az számok általában. Azonos kulcsok megengedettek.

Adatszerkezetek/hordozók: Általában tömbben vannak. Néha láncolt listában. (Esetleg háttértárolón, fájlban, (szekvenciálisan), de ezzel nem foglalkozunk még.)

## Alap tevékenység: Két elem összehasonlítása (ciklusban)

Van a kulcsokat vödrökbe helyező rendezés is (pl.: pénztárban az aprópénzeket külön rekeszbe rakjuk.)

## A 7 rendezési eljárás

* -es rendezések:
  1. Buborék
  2. Beszúró
  3. Maximumkiválasztó
* -es rendezések:
  1. Verseny
  2. Kupac
  3. Gyors (QuickSort)
  4. Összefésülő

(Ez mind 1-1, összesen 7 db vizsgakérdés lesz.)

Mit használunk gyakorlatban: QuickSort. És a Beszúró, ha n≤10 vagy n≤40.

## Van-e más rendezés? Van. Shell

Nem tanuljuk.

## Van-e jobb rendezés? Nincs.

Tétel kimondja, hogy -nél nincs jobb.

## Mi a különbség és között?

-re

# Három -es rendezés

## Buborékrendezés

Menetenként egy maximum felbuborékol.

### 1. menet

40,10,50,30,20

40 és 10 csere:  
10,40,50,30,20

50↔30  
10,40,30,50,20

20↔50  
10,40,30, 20, 50

### 2. menet

10,40,30, 20

40↔30  
10,30,40,20

40↔20  
10,30,20,40

### 3. menet

10,30,20

És így tovább…

Struktogram volt előadáson.

### Hatékonyság

Összehasonlítások száma:

Cserék száma:

Átlagos csereszám:

## Beszúró rendezés

Kártyázunk. A lapokat, ahogy vesszük fel, egyből a megfelelő helyre rakjuk a kezünkben.

Van 5 lap a kezünkben, jobbról balra megnézzük egyesével, melyik lapnál nem nagyobb. Oda berakjuk.

40,10,50,30,20

TFL első három már rendezve: 10,40,50

Most a 30-at szúrjuk be a rendezett első szakaszba.

### n. menet

10,40,50|30,20

Kiemeljük 30-at egy változóba, helyén marad egy lyuk:

10,40,50|\_,20

Összehasonlítjuk 50-nel. Nem nagyobb, ezért 50-et jobbra toljuk.

10,40,\_|50,20

40-nel hasonlóan:

10,\_,40|50,20

10 már nem nagyobb, helyén marad. A lyukba betesszük a 30-at:

10,30,40,50|20

### Struktogram

-et szúrjuk be -be.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  | | |  |
|  | | | |
|  | | | |
|  |  | | |
|  | | |
|  |  | |
| \* | | |

\* vagy egy -nél elemre mutat, vagy , azaz lelépett a tömbről (balra).

### Hatékonyság

Nem megy mindig végig, mint a Buborékrendezés.

Összehasonlítások száma:

Elemmozgások (nem cserék) száma: Legjobb esetben , , és van.

Átlagos elemmozgások száma: Mérni gyakorlatban lehet, de szép elméleti képlet nincs rá.

## Az elemmozgások összehasonlítása a Buborékrendezés és a Beszúró rendezés között

### Legrosszabb esetben

vs.

Azért jött be a 3-mas szorzó, mert egy csere 3 elemmozgatás.

Leegyszerűsítve ezt kapjuk:

A beszúró rendezés határozottan jobb, kb. -ször.

### Átlagos esetben

…

## Beszúró rendezés láncolt listákra is működik

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| l→ |  | → |  | 40 | → |  | 10 | → |  | 50 | → |  | 30 | → |  | 20 |  |

40,10,50,30,20

Rekordok a helyükön maradnak, csak a pointerek változnak.

TFH itt tartunk: l→[]→10→40→50→30→20

Egy változó, , a még be nem rendezett rész elejét mutatja.

Befűzés menete: csak 2 pointer átállítása. 30 a helyén marad.

Átállított pointerek: 10→30 és 30→40

Eredmény: l→[]→10→30→40→50→20

## Maximumkiválasztó rendezés

40,10,50,30,20

### 1. menet

max→50

Megcseréljük a jobb szélső elemmel.

40,10,0,30|50

### 2. menet

És így tovább…

|  |  |  |
| --- | --- | --- |
|  | | |
|  | |  |
|  | | |
|  |  | |

annyiban különbözik az eddig tanult maximumkereséstől, hogy helyett csak -n keressük a maximumot, és csak a helyét tároljuk, értékét nem.

Kétszeresen (oda-vissza) láncolt listákra működik.

### Hatékonyság

Nem fogja legyőzni a Beszúró rendezést. De van egy jó dolog, ami nem tananyag.

# A többi rendezés

## Versenyrendezés (Tournament sort)

A tournament adatszerkezet = versenyfa

("A kupacrendezés a versenyrendezés helyben rendező változata." Ez a mondat így nem helyes, de mindenki használja.)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |
|  |  |
|  |
|  |  |  |
|  |
|  |  |
|  |

Ez is összehasonlítást használ.

Senkit nem szabad kihagyni az összehasonlításokból.

Ha maximumkiválasztásra használjuk ezt a struktúrát, akkor nyerünk. 3 összehasonlításból meg is van a maximum érték helye.

-at cseréljük -ra. Azon az ágon ahol eddig 8 volt, újraszámolja az eredményeket:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |
|  |  |
|  |
|  |  |  |
|  |
|  |  |
|  |

Kijött a második maximum.

### Versenyrendezés teljes műveletigénye -ra

Összehasonlítások száma: Maximumkiválasztó ezzel szemben: (Mégsem jobb!)

Írjuk le általánosan:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

### Struktogram maximumkiválasztáshoz

|  |  |  |
| --- | --- | --- |
|  | | |
|  | |  |
|  | | |
|  |  | |

### Struktogram rendezéshez

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  | |  | |
|  | | | |
|  | | | |
|  |  | | |
|  | |  |
|  | | | |
|  | | | |
|  |  | | |

### Struktogram

|  |  |  |
| --- | --- | --- |
|  | | |
|  | |  |
|  | | |
|  | | |
|  |  | |

Megvan az első -es rendezés.

Szépséghiba: Sok memóriát használ. A kupacrendezés ezt helyben le tudja bonyolítani

Szépséghiba: Csak 2 hatványban működik. Természetesen meg lehet oldani nem 2 hatványra is, előversennyel. (19=16 + 3)

Előadás vége.