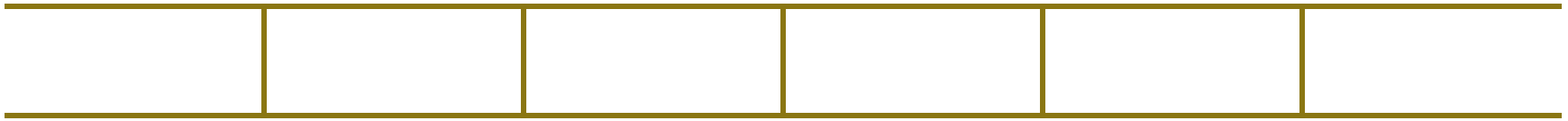
ADATSZERKEZETEK ÉS ALGORITMUSOK

Sor, Queue, FIFO
Prioritásos sor

Sor, Queue, FIFO

Sor (Queue)

- FIFO: First-in, First-out
- Köznapi fogalma



A sor ADT axiomatikus leírása

E alaptípus feletti S sor típus jellemzése

- Műveletek jelentése
  - Üres sor létrehozása
  - Üres a sor?
  - Elem betétele a sorba
  - Elem kivétele a sorból
  - Első elem lekérdezése
- Figyelem!
  - A műveletek között nem szerepel „isfull” művelet!

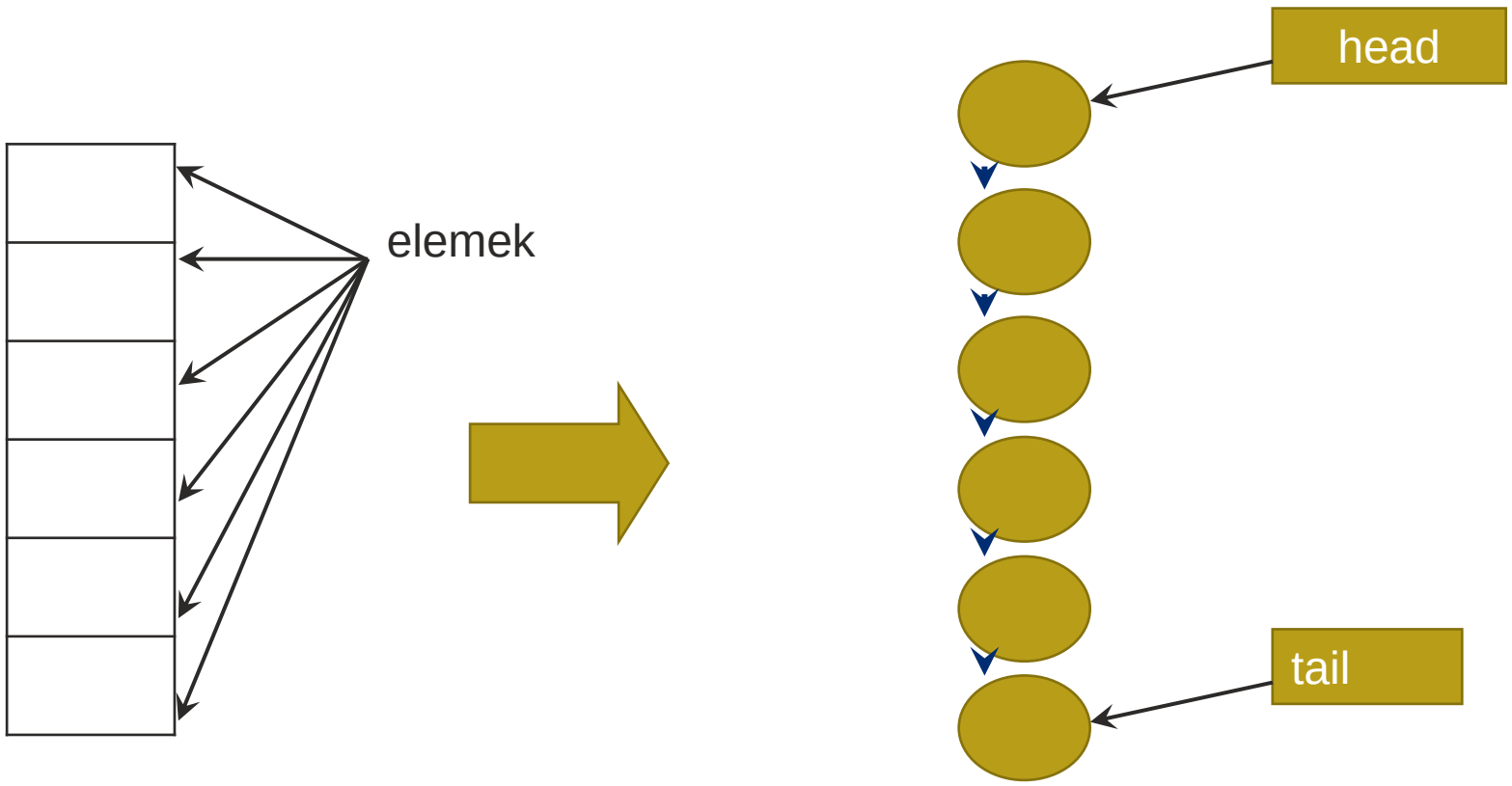
- Műveletek
  - empty $\rightarrow S$
  - isempty $S \rightarrow L$
  - in $S \times E \rightarrow S$
  - out $S \rightarrow S \times E$
  - first $S \rightarrow E$
- Megszorítások:
  - out és first értelmezési tartománya
    - $S \backslash \{\text{empty}\}$

Sor – ADT

- Axiómák:
  - $s = \text{empty} \rightarrow \text{isempty}(s)$
  - $\text{isempty}(s) \rightarrow s = \text{empty}$
  - $\neg\text{isempty}(\text{in}(s, e))$
  - $\neg\text{isempty}(s) \rightarrow \text{out}(\text{in}(s, e))_2 = \text{out}(s)_2$
  - $\neg\text{isempty}(s) \rightarrow \text{in}(\text{out}(s)_1, e) = \text{out}(\text{in}(s, e))_1$
  - $\text{isempty}(s) \rightarrow \text{out}\big(\text{in}(s, e)\big)_2 = e$
  - $\text{first}(s) = \text{out}(s)_2$
  - Ahol: $(\text{s,e})_1 = s$, $(\text{s,e})_2 = e$
- Végig kell gondolni, önállóan – vizsgakérdés!

Sor – ADS

- Lineáris adatszerkezet

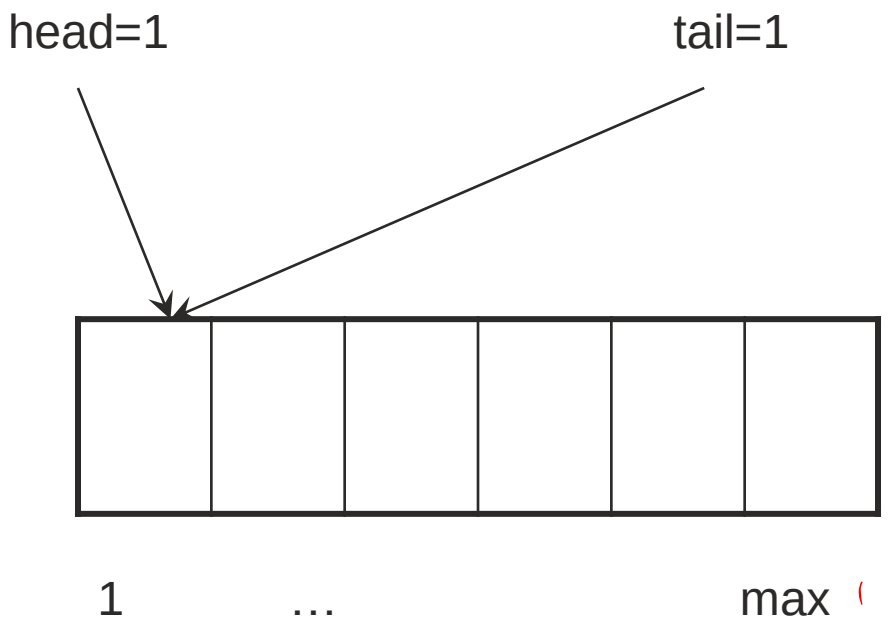


Reprezentáció

- Aritmetikai ábrázolás:
  - egy $\max$ hosszú vektor
    - ez az elemek tömbje
    - $\text{elements}[1..\max]$
  - és a sor első elemének mutatója
    - $\text{head} \in [1, \max]$
  - és a sor első üres (utolsó) helyének mutatója
    - $\text{tail} \in [1, \max]$
- Vegyük észre, hogy az aritmetikai ábrázolás három részből áll!

Reprezentáció

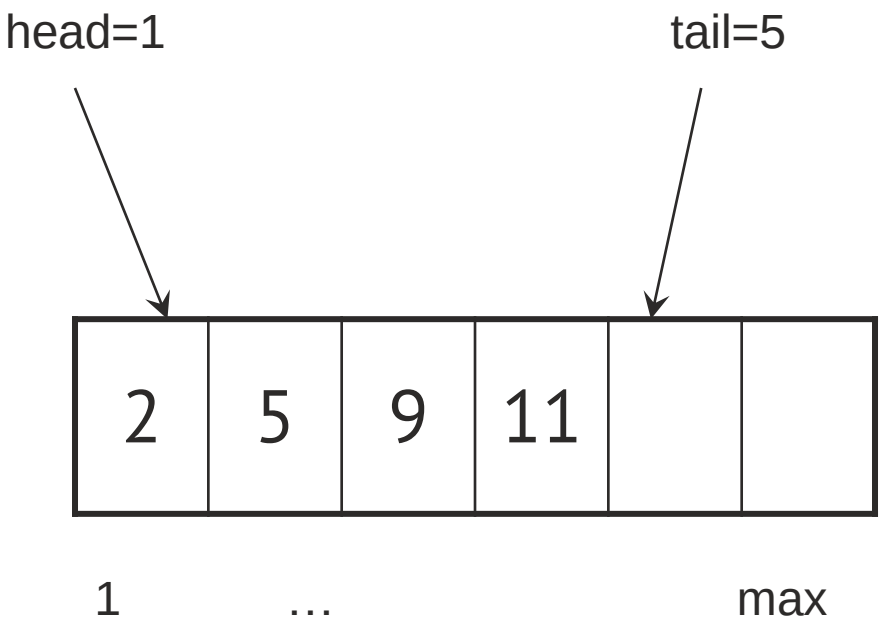
- Ciklikus ábrázolással
  - Kezdetben



Üres a sor

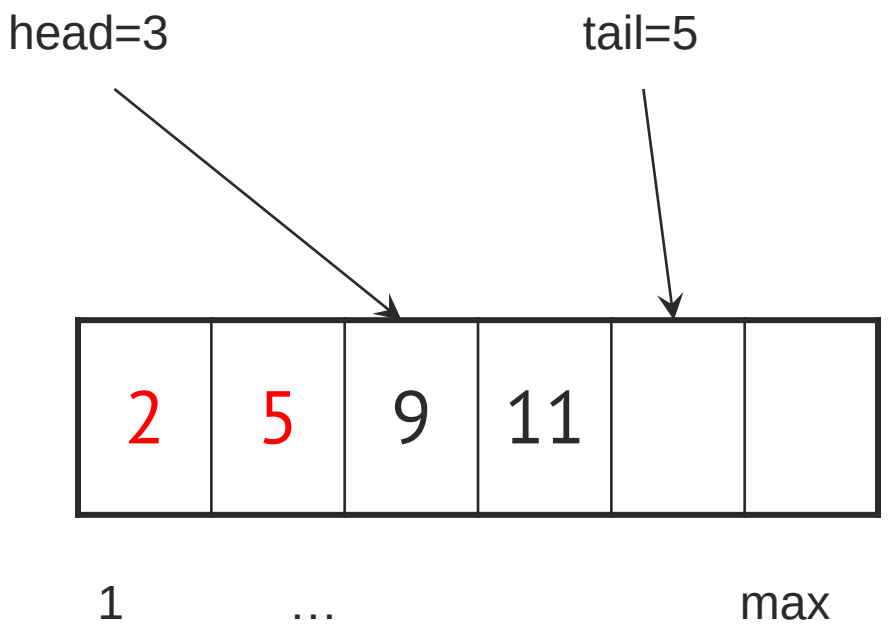
Reprezentáció

- 2, 5, 9, 11 betétele után:



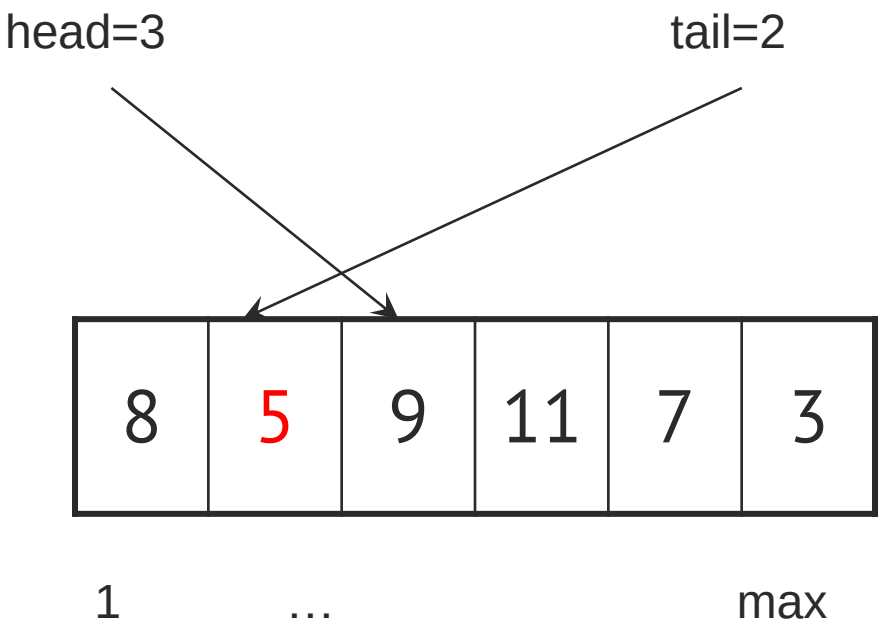
Reprezentáció

- 2, 5 kivétele után:



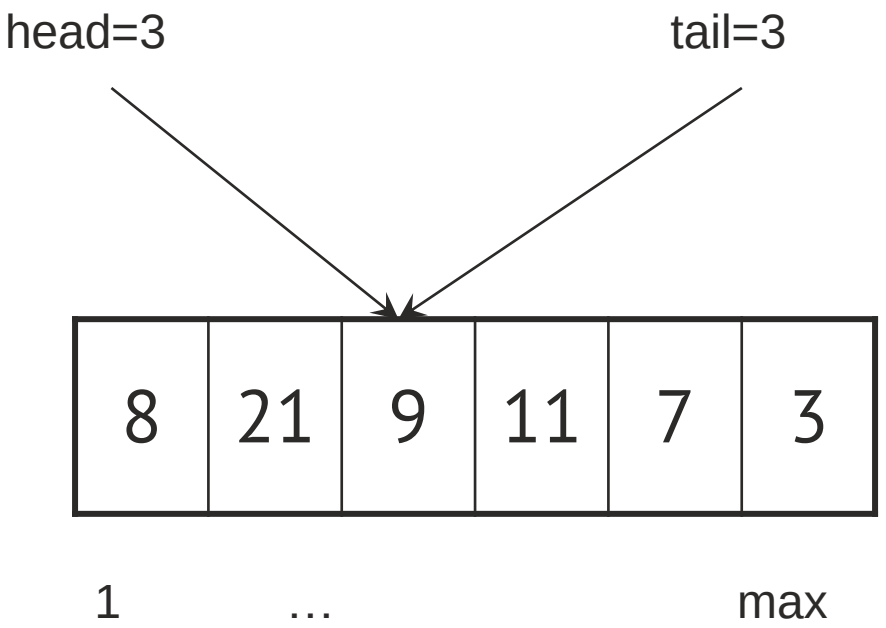
Reprezentáció

- 7, 3, 8 betétele után:



Reprezentáció

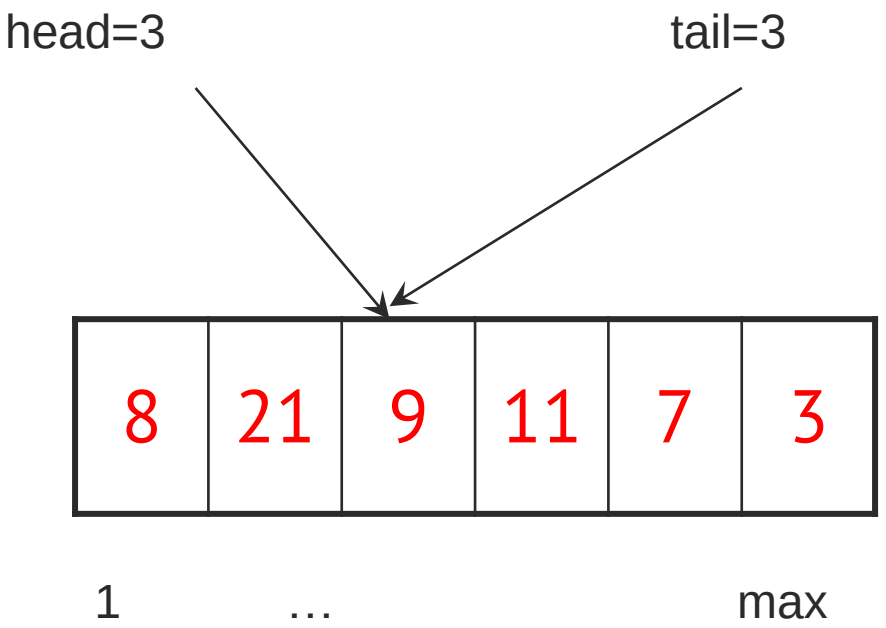
- 21 betétele után:



Tele a sor

Reprezentáció

- 9, 11, 7, 3, 8, 21 kivétele után:



Üres a sor

Reprezentáció

- Mit tegyünk? – Milyen lehetőségek vannak:
  - Vezessünk be még egy jelzőt a reprezentációba, ami mutatja, hogy a sor üres-e
    - `empt`
    - kezdetben igaz, később vizsgáljuk, és megfelelően állítjuk
  - Vezessünk be még egy attribútumot a reprezentációba, ami mutatja, hogy hány elem van a sorban
    - `count`

Implementáció

- Műveletek pszeudokódja:
  - s.empty
    -- üresre állítja a sort
    s.head ← 1; s.tail ← 1; s.empt ←true
  - s.isempty
    -- üres a sor? - logikai értéket ad vissza
    return s.empt
  - s.isfull
    -- tele van a sor?
    return ((not s.empt) and (s.head = s.tail))

Implementáció

- s.In(e)

```
-- e-t beteszi az s sor végére
-- s.tail-t ciklikusan növeli
if s.IsFull
  then error "túlcsordulás"
  else s.empt ← false
    s. elements[s.tail] ← e
    if s.tail=max
      then s.tail ← 1
      else s.tail ← s.tail+1
    end if
end if
```

Implementáció

- s.Out

```
-- kiveszi és visszaadja az s sor első elemét
-- s.head-et ciklikusan növeli
-- figyeli, hogy nem üres-e a sor
if s.empt
  then error "alulcsordulás";
  else e ← s.elements[s.head]
       if s.head=max
         then s.head ← 1
         else s.head ← s.head+1
       end if
       if s.head=s.tail then s.empt ← true end if
       return e
end if
```

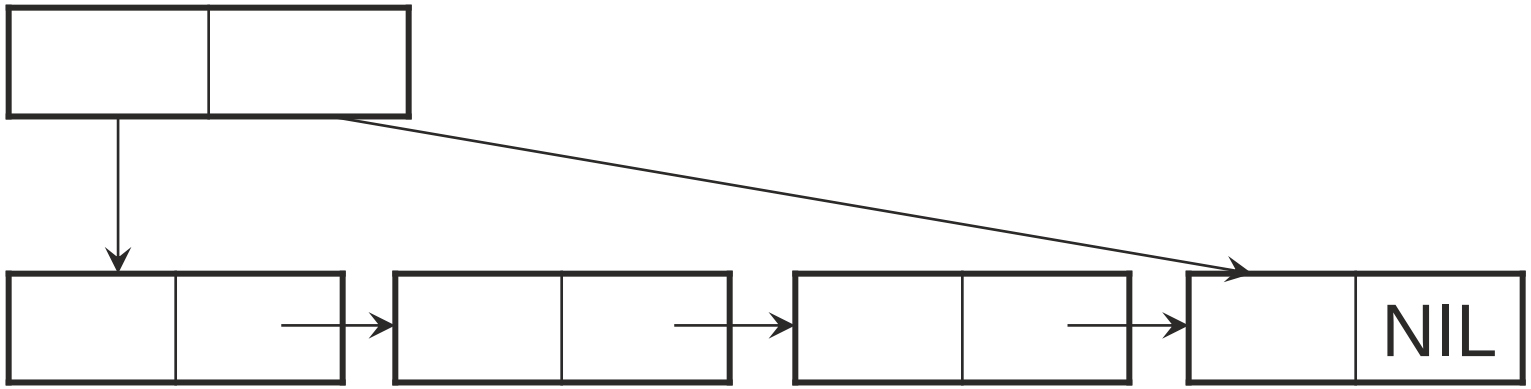
Implementáció

- ```
  s.First
  -- visszaadja az s sor első elemét,
  -- figyeli, hogy nem üres-e a sor
  if s.empt
    then error "alulcsordulás"
    else return s. elements[s.head]
  end if
  ```

- Lehetne az is, hogy a darabszámot tároljuk
  - Házi feladat: átgondolni
  ```

Reprezentáció

- Lesz még láncolt ábrázolás is!



Implementáció

- Vigyázni kell, amikor a programokat a választott programnyelven megvalósítjuk!
  - Például
    - C++ nyelv esetén a vektorok indexelése nullával kezdődik!
  - Értékadás jele
  - Egyenlőség vizsgálat jele

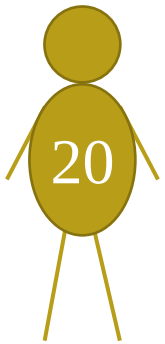
Prioritásos sor

Egyszerű sor → prioritásos sor

- Egyszerű sor:
  - FIFO szemantika
  - Elem hozzáadása, törlése konstans ($\mathcal{O}(1)$) igényű
- Mit tegyünk, ha a sorban lévő elemeknek valamifajta rendezése is van?
  - Ezt általában prioritásnak nevezzük.
  - Úgy kell rendezzük az elemeket, hogy a legnagyobb (legkisebb) prioritású elemet töröljük először.

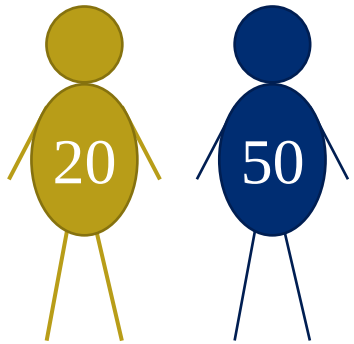
Elsőbbségi (prioritásos) sor

- Példa
  - sürgősségi osztály
    - különböző súlyosságú esetek



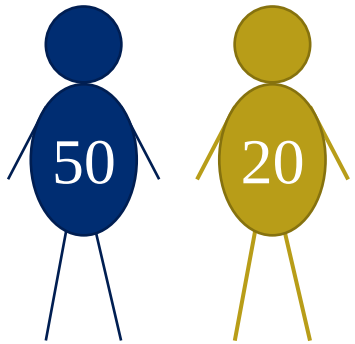
Elsőbbségi (prioritásos) sor

- Példa
  - sürgősségi osztály
    - különböző súlyosságú esetek



Elsőbbségi (prioritásos) sor

- Példa
  - sürgősségi osztály
    - különböző súlyosságú esetek

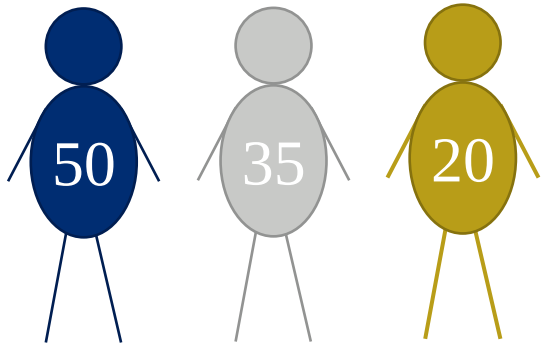


Elsőbbségi (prioritásos) sor

- Példa
  - sürgősségi osztály
    - különböző súlyosságú esetek

Elsőbbségi (prioritásos) sor

- Példa
  - sürgősségi osztály
    - különböző súlyosságú esetek



Elsőbbségi (prioritásos) sor

- Mire lehet használni?
  - Tennivalók, melyiket kell először elvégezni
  - Operációs rendszer, mely prioritásos feladatokat (jobokat) dolgoz fel
    - Itt különböző további algoritmusok, amelyek a prioritást meghatározzák egy-egy folyamat (job) számára
  - Telekommunikációban a csomagok továbbításánál is

Elsőbbségi (prioritásos) sor

- Az elsőbbségi sor ADT axiomatikus leírása
- E alaptípus feletti P elsőbbségi sor típus jellemzése:
  - Egyszerűsítés: csak prioritásokat teszünk bele (N)
  - Műveletek „full" nem szerepel
    - empty $\rightarrow P$ az üres prior. sor konstruktor – létrehozás
    - isempty $P \rightarrow L$ üres a prioritásos sor?
    - insert: $P \times N \rightarrow P$ elem betétele a prioritásos sorba
    - delmax: $P \rightarrow P \times N$ maximális elem kivétele a pr. sorból
    - max: $P \rightarrow N$ maximális elem lekérdezése
  - Megszorítások:
    - delMax és max értelmezési tartománya $P \setminus \{empty\}$

Elsőbbségi (prioritásos) sor

- Axiómák:
  1. $\text{isempty}(\text{empty})$ vagy $p = \text{empty} \rightarrow \text{isempty}(p)$
  2. $\text{isempty}(p) \rightarrow p = \text{empty}$
  3. $\neg\text{isempty}(\text{insert}(p, n))$
  4. $\text{insert}(\text{delmax}(p)) = p$
  5. $\max(p) = \text{delmax}(p)_2$
  6. $\text{delmax}(p)_1 \neq \text{empty} \rightarrow \max(p) \geq \max(\text{delmax}(p)_1)$
  7. $n \geq \max(p) \rightarrow \text{delmax}(\text{insert}(p, n))_1 = p \wedge \max(\text{insert}(p, n)) = n$
  8. $n < \max(p) \rightarrow \max(\text{insert}(p, n)) = \max(p)$
  9. $\text{delmax}(\text{insert}(\text{empty}, n)) = (\text{empty}, n)$
  10. $\max(\text{insert}(\text{empty}, n)) = n$
  - (7. és 8.-nál feltettük, hogy nem üres a prioritásos sor – ez az értelmezési tartomány megszorítása!)

Elsőbbségi (prioritásos) sor

- Kérdés: hogyan ábrázoljuk, mivel reprezentáljuk?
  - Rendezetlen tömbbel, a beérkezési idő szerint
    - max műveletigénye mindig egy maxker, vagyis $\Theta(n)$
  - Rendezett tömbbel
    - insert műveletigénye:
      - A hely megkeresése $\rightarrow$ logaritmikus keresés $\Theta(\log_2 n)$
      - Tőle jobbra léptetés $\Theta(n)$
      - Összesen $\Theta(n)$
  - Heap (kupac) adatszerkezettel

Lengyel forma

Következő téma