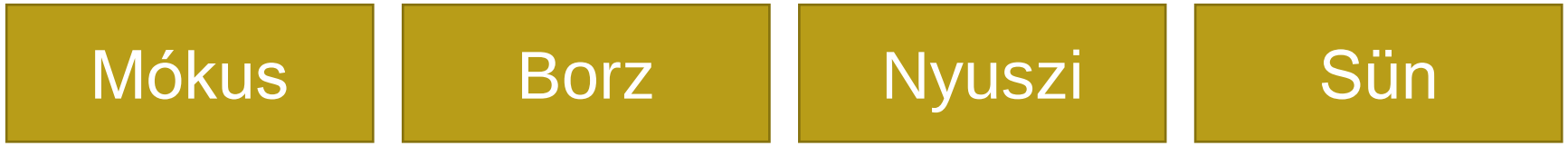
ADATSZERKEZETEK ÉS ALGORITMUSOK

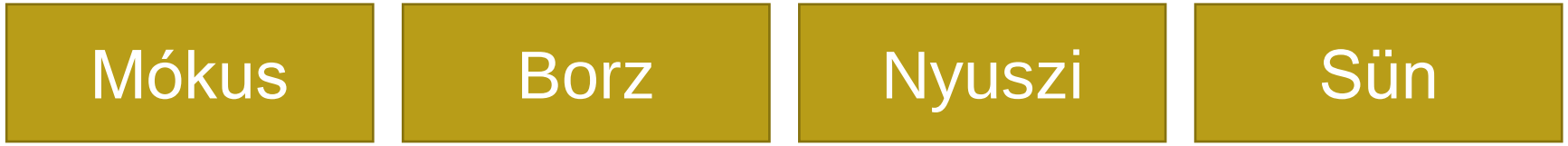
Verem, Stack, LIFO

Verem fogalma

- LIFO: last-in, first-out
- Köznapi fogalma



Sorrend az elemek betétele előtt:



Sorrend miután az összes elem kikerült:

A verem ADT axiomatikus leírása

E alaptípus feletti V verem típus jellemzése

- Műveletek jelentése
  - Üres verem létrehozása
  - Üres a verem?
  - Elem betétele a verembe
  - Elem kivétele a veremből
  - Felső elem lekérdezése
- Figyelem!
  - A műveletek között nem szerepel „isfull” művelet!

- Műveletek
  - empty $\rightarrow V$
  - isempty $V \rightarrow L$
  - push $V \times E \rightarrow V$
  - pop $V \rightarrow V \times E$
  - top $V \rightarrow E$
- Megszorítások:
  - pop és top értelmezési tartománya
    - $V \backslash \{\text{empty}\}$

A verem ADT axiomatikus leírása

- Axiómák
  - $\text{isempty}(\text{empty})$
    - Vagy: $v = \text{empty} \rightarrow \text{isempty}(v)$
  - $\text{isempty}(v) \rightarrow v = \text{empty}$
  - $\neg\text{isempty}(\text{push}(v, e))$
  - $\text{pop}(\text{push}(v, e)) = (v, e)$
  - $\text{push}(\text{pop}(v)) = v$
  - $\text{top}(\text{push}(v, e)) = e$

Verem ADT – Műveletek matematikai magyarázata

- A típushoz tartozó műveleteket formálisan specifikáljuk.
  - Ezek függvények: egy adott halmaz (értelmezési tartomány) elemeihez rendelnek hozzá egy másik halmaz (értékkészlet) elemei közül.
    - Saját típus (Verem) esetén:
      isempty $V \to L$
      push $V \times E \to V$
  - Ez utóbbi esetén a művelethez felírhatók példák: adott v verem és e elem párhoz hozzárendelünk egy v’ vermet.
  - Ahhoz, hogy a művelet „megfelelően” működjön matematikai módon leírjuk a viselkedését.
    - Ezek az axiómák!
    - $\text{pop}(\text{push}(v, e)) = (v, e)$
    - $\text{push}(\text{pop}(v)) = v$

A verem ADT funkcionális leírása

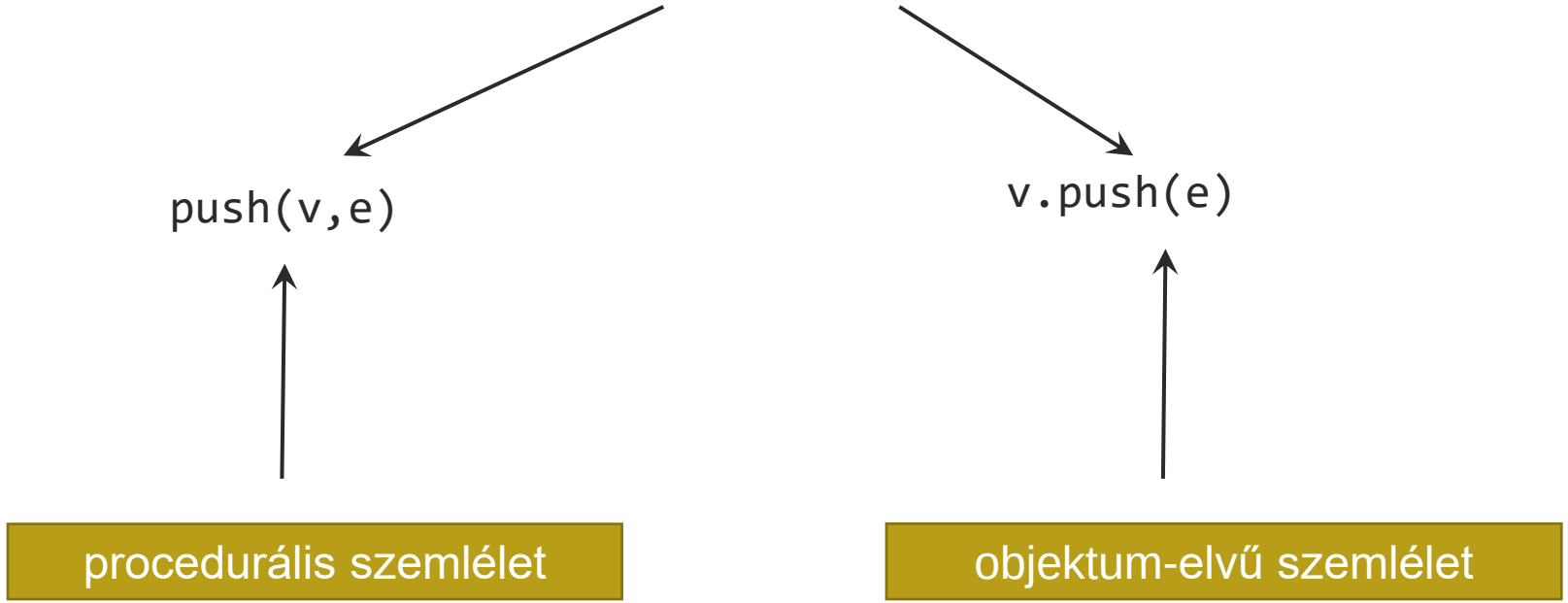
- Matematikai reprezentáció
  - a verem rendezett párok halmaza
    $v = \{(e_1, t_1), \ldots (e_i, t_i) |\ a\ t_j$ komponensek különbözőek$\}$
  1. komponens: a veremben elhelyezett (push) érték
  2. komponens: a verembe helyezés (push) időpontja
- Megszorítás (invariáns)
  - az idő értékek különbözők
- A valóságban nem így implementáljuk!

A verem ADT funkcionális leírása

- A „pop” specifikációja:
  - Állapottér
    - $V \times E$ (Állapottér típusai: $V = \{(e_i, t_i), \dots\}$
    - $v \quad e$ (Állapottér változói)
  - Paramétertér
    - V és v'
  - Előfeltétel és utófeltétel:
  - $\mathrm{Ef} = (v = v' \wedge v' \neq \emptyset)$
  - $\mathrm{Uf} =$
    $\Big((v = v' \backslash \{(e_j, t_j)\}) \wedge (e = e_j) \wedge \big((e_j, t_j) \in v'\big) \wedge \big(\forall i\big((e_i, t_i) \in v' \wedge i \neq j\big): t_j > t_i\big)\Big)$

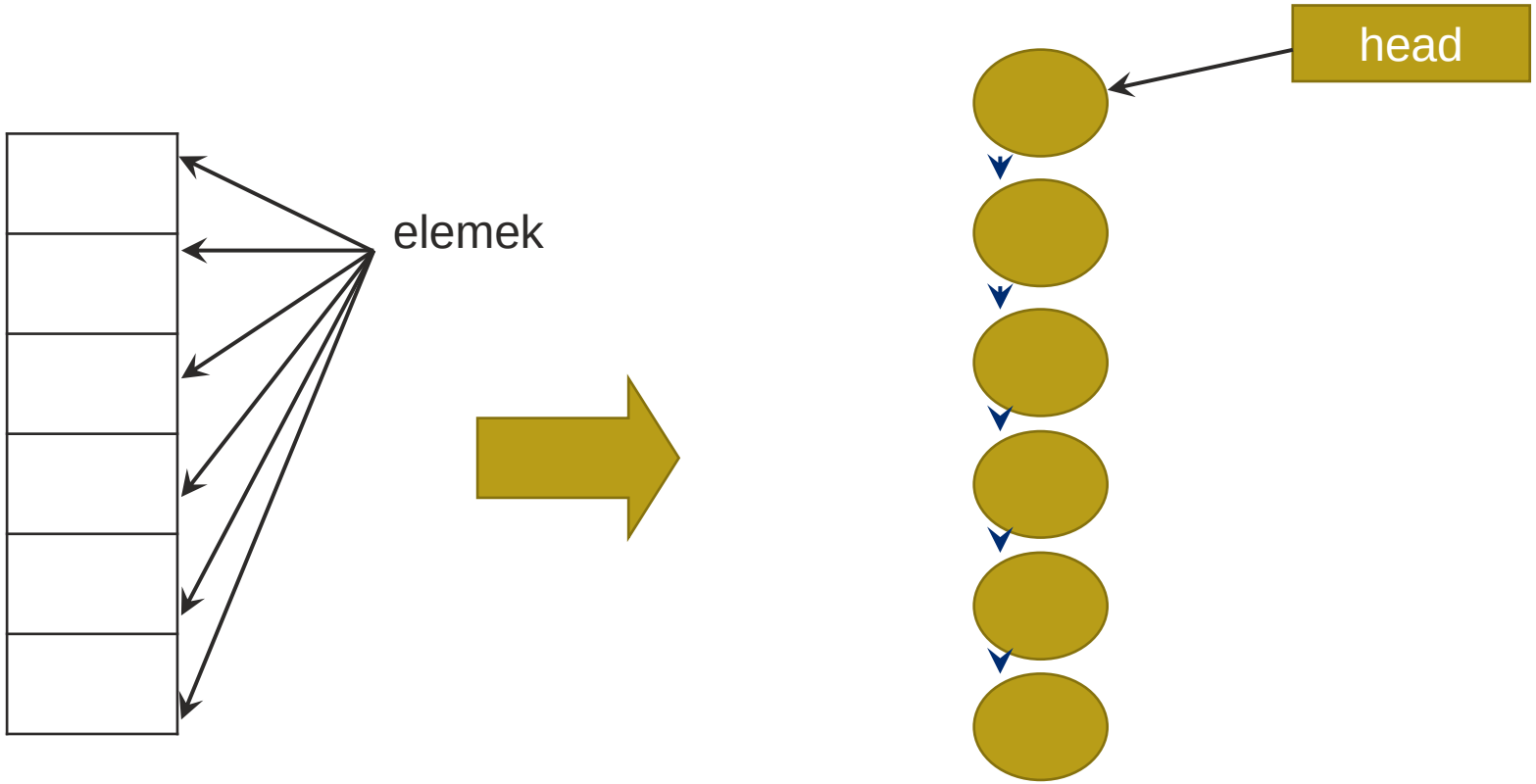
Verem

Műveletek jelölése



Verem – ADS

- Lineáris adatszerkezet

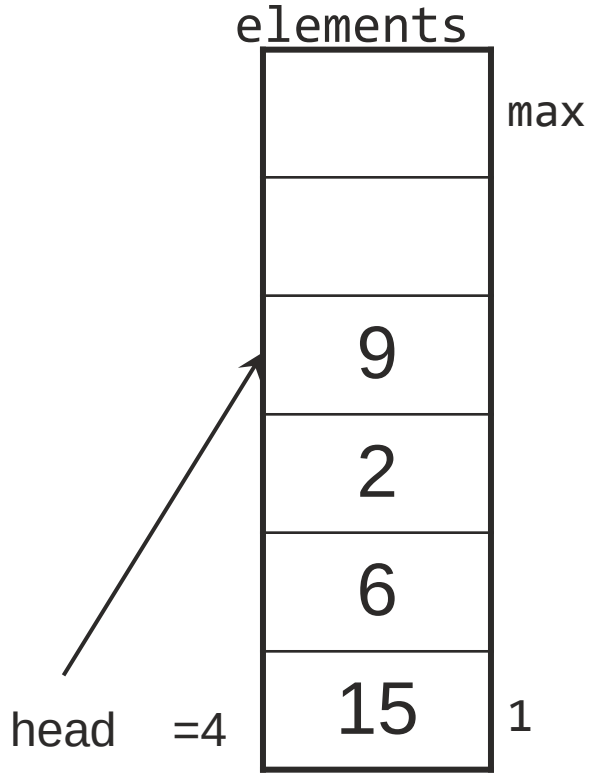


Elemek száma

- Felépített adatszerkezet
  - Az adatszerkezet elemeinek száma a feldolgozás során rögzített vagy változtatható
  - A rögzített nem jelenti, hogy a tárolt adatok nem megváltoztathatók
- Adatelemek száma
  - Fix
    - A tárolható adatelemek számának felső korlátja a létrehozáskor (esetleg fordítási időben) rögzített.
  - Változó
    - A memória mérete (illetve kapcsolódó technikai korlátok) szab határt az adatelemek számának

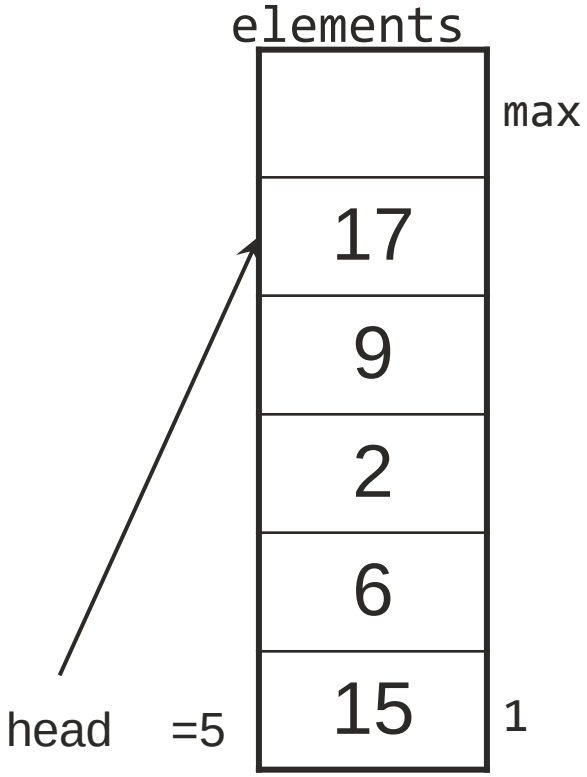
Reprezentáció

- Aritmetikai ábrázolás:
  - egy max hosszú vektor (ez az elemek tömbje) elements[1..max]
  - a verem tetejének mutatója head $\in$ [0, max] head=0 $\Leftrightarrow$ üres a verem
  - Választási lehetőség, hogy hova mutat a head
    - Az első szabad helyre
    - Az utolsó elfoglalt helyre



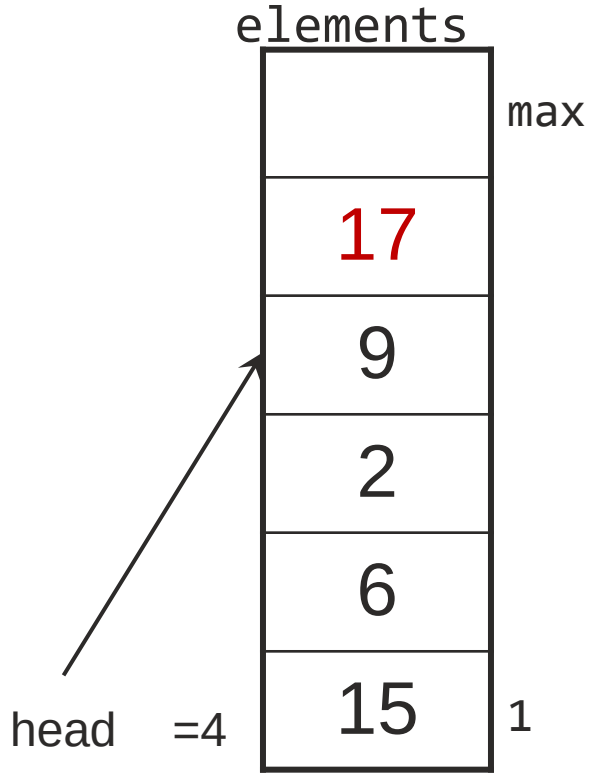
Reprezentáció

v.push(17) után



Egyenlő a két verem?

v.pop() után



Implementáció

- Műveletek pszeudokódja/struktogramja:
  - `v.empty`

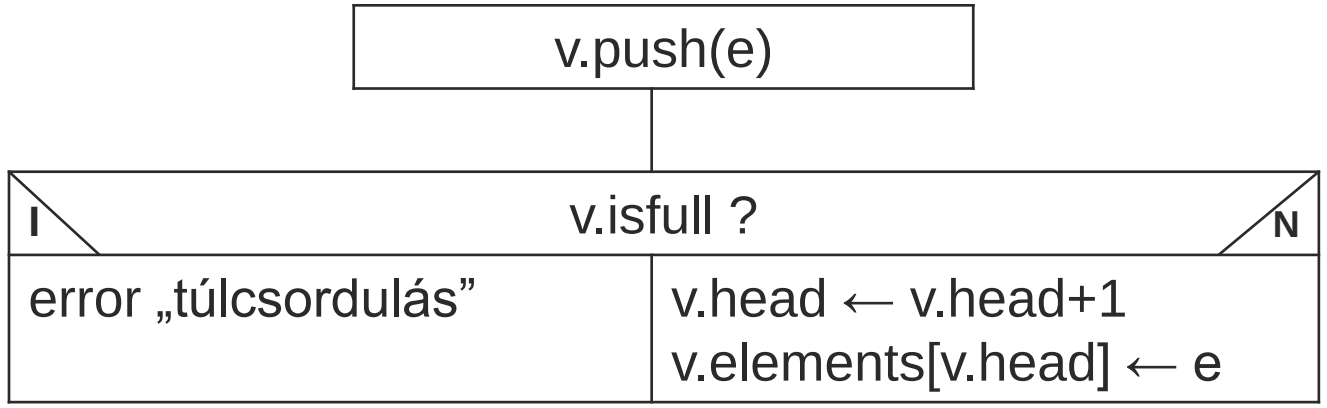
    ```
    -- üresre állítja a vermet
    v.head ← 0
    ```
  - `v.isempty`

    ```
    -- Üres a verem? – Logikai értéket ad vissza
    return (v.head=0)
    ```
  - `v.isfull`

    ```
    -- Tele van a verem? – Logikai értéket ad vissza
    return (v.head=max)
    ```

Implementáció

- v.push(e)
  -- e-t beteszi a v verem tetejére
  if v.isfull
    then error "túlcsordulás"
    else v.head ← v.head +1
    v.elements[v.head] ← e
  end if



Implementáció

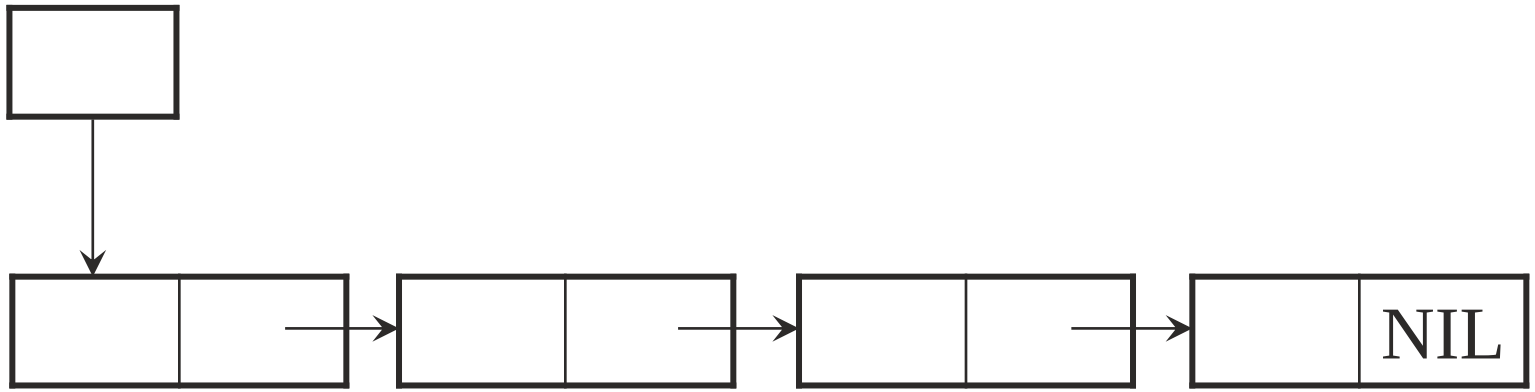
- ```
  v.pop
  -- kiveszi a legfelső elemet és visszaadja
  if v.isempty
    then error "alulcsordulás"
    else v.head ←v.head -1
      return v.elements[v.head+1]
  end if
  ```
  ```

Implementáció

- ```
  v.top
  -- lekérdezi a legfelső elemet
  if v.isempty
    then error "alulcsordulás"
    else return v.elements[v.head]
  end if
  ```
  ```

Reprezentáció

- Láncolt ábrázolás (gyakorlaton)



Sor, Prioritásos sor

Következő téma