Algoritmika

3. szeminárium

Többségi elem (elnökválasztás)

Adott egy n elemű, természetes számokból álló tömb.
Határozzuk meg, hogy van-e olyan elem, amely több mint n/2 példányban fordul elő a tömbben. (többségi elem)

Példák:

n =10	
1, 2, 1, 1, 1, 2, 2, 2, 1, 1	**Többségi elem: 1**
1, 2, 1, 2, 1, 2, 1, 2, 1, 2	**Nincs többségi elem**
1, 2, 1, 2, 1, 2, 1, 2, 1, 1	**Többségi elem: 1**
1, 1, 1, 1, 2, 3, 4, 5, 1, 1	**Többségi elem: 1**
1, 2, 3, 4, 5, 6, 1, 1, 1, 1	**Nincs többségi elem**

Többségi elem (elnökválasztás)

Naív algoritmus:

- a többségi elem biztosan a tömb “leggyakoribb” eleme

- minden elemről megszámoljuk hányszor fordul elő, meghatározzuk a leggyakoribb elemet

- ha a leggyakoribb elem több mint n/2-szer fordul elő a tömbben, akkor van többségi elemünk, különben nincs

```
Algoritmus TöbbségiElem (a, n)
    max ← 0, sorszám ← 0
    Minden i = 1, n Végezd el
        db ← 0
        Minden j = 1, n Végezd el
            Ha a[i] = a[j] akkor
                db ← db + 1
            Vége (Ha)
        Vége (Minden)
        Ha db > max akkor
            max ← db, sorszám = i
        Vége (Ha)
    Vége (Minden)
    Ha max > n/2 akkor
        Vissza sorszám
    Különben
        Vissza -1
Vége (Algoritmus)
```

Többségi elem (elnökválasztás)

Gyorsítás:

- minden elem előfordulását csak egyszer számoljuk

- ha találtunk egy olyan elemet, amely többségi szám, ne számoljunk tovább

- a többségi elemet elég csak a tömb első felében keresni (mivel ahhoz, hogy többségi elem legyen, legalább fele+1-szer kell szerepeljen a tömbben)

Többségi elem (elnökválasztás)

Gyorsítás:
- minden elem előfordulását csak egyszer számoljuk
- ha találtunk egy olyan elemet, amely többségi szám, ne számoljunk tovább
- a többségi elemet elég csak a tömb első felében keresni (mivel ahhoz, hogy többségi elem legyen, legalább fele+1-szer kell szerepeljen a tömbben)

Más ötlet:
- használjunk gyakorisági tömböt

Kérdés: Lehet-e jobban?

Többségi elem (elnökválasztás)

Moore algoritmus:

1. **Keressük meg a potenciális többségi elemet:**
   - **Amennyiben létezik többségi elem, akkor a visszatérített érték biztosan a többségi elem lesz**
   - **Amennyiben nem létezik többségi elem, egy potenciális “jelöltet” térít vissza**

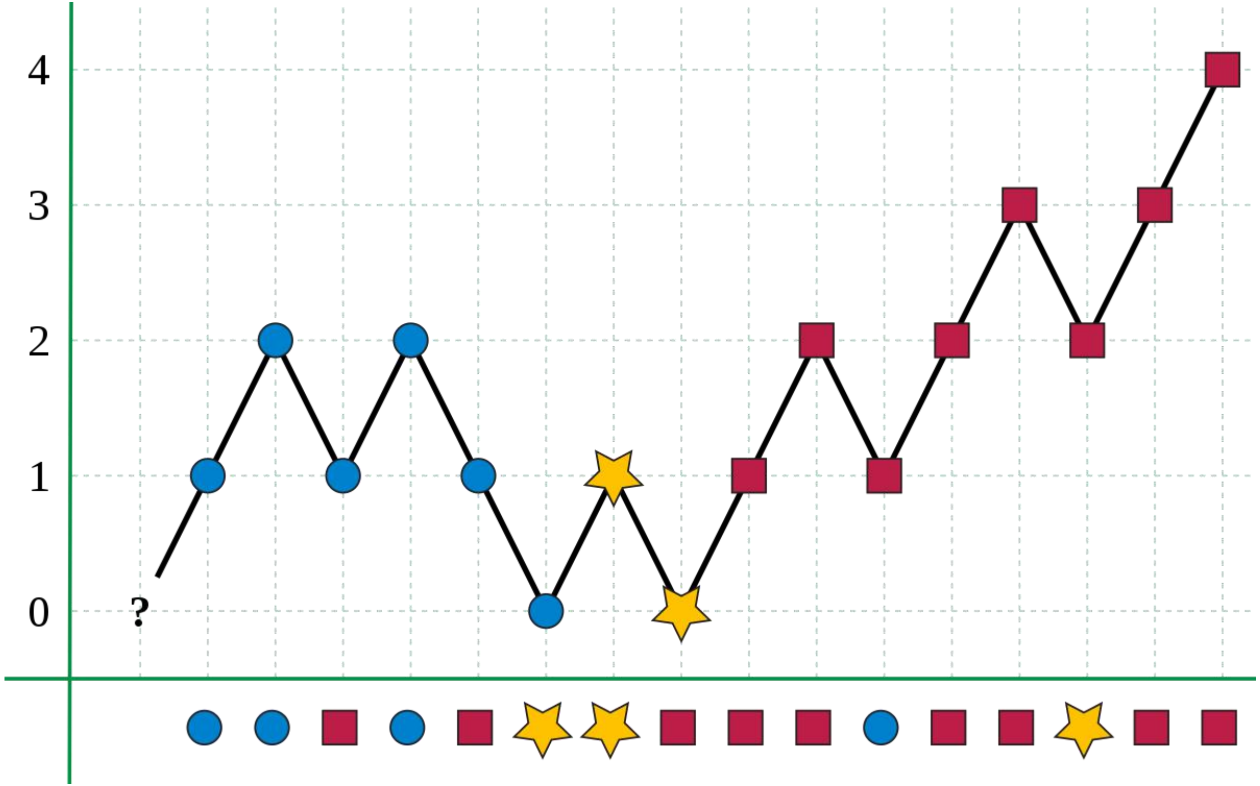
2. **Ellenőrizzük a kapott elemről, hogy ténylegesen többségi elem-e**

Algoritmus TöbbségiElem (a, n):

```
{jelölt keresése}

jelölt ← a[1],
hány ← 1
Minden i=2, n végezd el:
   Ha a[i] = jelölt akkor
      hány ← hány + 1
   Különben
     Ha hány = 0 akkor
        jelölt ← a[i]
        hány ← 1
     Különben
        hány ← hány -1
     Vége (Ha)
   Vége (Ha)
Vége (Minden)
              ...
```

```
          ...
{ellenőrzés, hogy valóban többségi elem-e}

   hány ← 0
   Minden j = 1, n végezd el:
      Ha a[i] = jelölt akkor
         hány ← hány + 1
      Vége (Ha)
   Vége(Minden)
   Ha hány > n/2 akkor
      Visszatérít jelölt
   Különben
   Visszatérít -1

Vége (Algoritmus)
```

Többségi elem (elnökválasztás)



Többségi elem (elnökválasztás)

Moore algoritmus:

1. **Keressük meg a potenciális többségi elemet:**
   - **Amennyiben létezik többségi elem, akkor a visszatérített érték biztosan a többségi elem lesz**
   - **Amennyiben nem létezik többségi elem, egy potenciális “jelöltet” térít vissza**

2. **Ellenőrizzük a kapott elemről, hogy ténylegesen többségi elem-e**

Az algoritmus előnye: - O(n) futási idő + nem szükséges tömb használata

Sorozatszámítás

Számoljuk ki $\sqrt{a}$ **értékét (a természetes szám), felhasználva a következő sorozatot:**

$$x_n = \frac{1}{2} \cdot \left(x_{n-1} + \frac{a}{x_{n-1}} \right)$$

$$x_0 = a$$

Az eredményt $\varepsilon = 10^{-6}$ **pontossággal írassuk ki.**

Sorozatszámítás

- **Leállási feltétel: ha két egymás után kiszámolt tag értéke közötti különbség kisebb vagy egyenlő mint ε**

- **Ilyen körülmények között x egy tömb, de nem ismerjük az elemek számát**

- **Az x_n sorozat konvergens és határértéke éppen** $\sqrt{a}$

- **A sorozatból annyi tagot számítunk ki amennyi egy adott pontosságot biztosít, azaz eljárásunk az n-edik tag kiszámítása után megáll, ha** $|x_n - x_{n-1}| < \varepsilon$

Sorozatszámítás

Algoritmus Négyzetgyök(a, eps):

```
        x ← a      { bemeneti adatok: a, eps; kimeneti adat: y }
        y ← (a + 1) / 2

        Amíg |x - y| > eps végezd el:
                x ← y
                y ← 1/2 * (x + a/x)
        Vége (Amíg)

        Visszatérít: y

Vége(algoritmus)
```