Keressünk meg adott tulajdonságú elemet az adattömbben!

Lineáris keresés
Alap tétel, már vettük az algoritmusoknál.

I=Kezdőindex
CIKLUS AMÍG I<= *Végsőindex* ÉS *Tömb*[I] nem adott tulajdonságú
    I = I+1
CIKLUS VÉGE
HA I<=*Végsőindex*
    AKKOR Ki: I a megtalált elem indexe
    EGYÉBKÉNT Ki: nem találtunk megfelelő elemet
FELTÉTEL VÉGE

A végrehajtás >=N/2. Ha minden keresés sikeres, akkor éppen N/2 az átlagos lépésszám. Minél nagyobb a sikertelen keresések aránya, annál nagyobb lesz az átlagos lépésszám. (Max N, ha minden keresés sikertelen.)

Ha több adott tulajdonságú elem is van, akkor az elsőt találja meg az algoritmus. Esetleg az utolsót, ha visszafelé haladunk az indexekkel.
Ha van lehetőség az egyenlő elemekre, akkor tovább is keresni kell a tömbben. Pl. név szerint keresni. (Több Kovács István) De pl. rendszám szerint, keresni nem kell többet.

Hogyan keresünk, ha az adathalmaz rendezett? Hogyan keresünk pl. egy szót a szótárban, vagy egy nevet a telefonkönyvben? Mikor derül ki a keresés sikertelensége?

Keressünk meg adott tulajdonságú elemet a **RENDEZETT** adattömbben!

Lineáris keresés strázsával

I=Kezdőindex
CIKLUS AMÍG I<= *Végsőindex* ÉS *Tömb*[I] < keresett érték
    I = I+1
CIKLUS VÉGE
HA Tömb[I]=keresett érték
    AKKOR Ki: I a megtalált elem indexe
    EGYÉBKÉNT Ki: nem találtunk megfelelő elemet
FELTÉTEL VÉGE

Ebben az esetben biztos, hogy az átlagos lépésszám <=N/2.

Keressünk meg adott tulajdonságú elemet a **RENDEZETT** adattömbben!

Logaritmikus keresés

```
alsó:= 1 :    felső:= N
Ciklus amíg alsó<=felső
    közép:= egészrész((alsó+felső)/2)
    Ha Tömb[közép] < keresett érték
        akkor alsó:= közép+1
    Elágazás vége
    Ha Tömb[közép] > keresett érték
        akkor felső:= közép-1
        egyébként alsó=felső+1;
    Elágazás vége
Ciklus vége
```

```
Ha Tömb[közép] = keresett érték
    akkor sorszám:= közép
    különben nincs a keresett érték
Elágazás vége
```

Vegyük észre a drámai javulást a végrehajtásban.
Egy 1000 elemű tömbben a lineáris keresések hatékonysága >=500 lépés.

A logaritmikus keresés hatékonysága $\log_2 N$, azaz 1000 elemből 10 lépéssel találja meg a keresettet. (És 1 000 000-ból is csak 20, miközben a lineáris módszerek 500 000 lépéssel!!)

Természetesen itt is feltétel a rendezettség a keresési értékre.

Persze joggal merülhet fel a kérdés, hogyan lesz rendezett a tömb.
Az idők folyamán rengeteg rendezési algoritmust alkottak. Az egyik leg időigényesebb tevékenység a rendezés. Nagyon fontos tehát a módszerek elemzése a hatékonyság szempontjából.

Buborék rendezés

Ez a rendezés a szomszédos elemek cserélésével alakítja ki a megfelelő sorrendet.

```
Ciklus i=n-től 2-ig
    Ciklus j=1-től i-1-ig
        Ha Tömb[j]>Tömb[j+1]
            akkor Csere(Tömb[j], Tömb[j+1])
        Felt. Vége
    Ciklus vége
Ciklus vége
```

Animáció

Beszúrásos rendezés

Ez a beilleszt egy elemet a már részben rendezett tömbbe. Először egy elemű rendezett tömbbe illeszti a 2. elemet.

```
Ciklus i=2-től N-ig
    j=i-1 : Temp=Tömb[i]
    Ciklus amíg j>0 és Tömb[j]>Temp
        Tömb[j+1]=Tömb[j] : j=j-1
    Ciklus vége
    Tömb[j+1]=Temp
Ciklus vége
```

Animáció

Rendezés minimum kiválasztással

Ez egy természetes gondolatmenet. Keressük ki a legkisebbet, ez lesz az első helyen, majd keressük ki a maradékból a legkisebbet, ez lesz a második helyen. És így tovább.

```
Ciklus i=1-től N-1 -ig
    Min=i
    Ciklus j=i+1 –től N-ig
        Ha Tömb[Min]>Tömb[j]
            akkor Min=j
        Feltétel vége
    Ciklus vége
    Csere(Tömb[Min], Tömb[i])
Ciklus vége
```

Animáció

Ezeket a rendezéseket közvetlen rendezéseknek nevezzük.

Az algoritmusok viszonylag természetesek, és közös jellemzőjük, hogy nem túl hatékonyak. A művelet igény n^2-el arányos, azaz ha a darabszám a duplájára nő, akkor rendezési idő a négyszeresére növekszik!!!

Szóval ha lehet kerüljük a rendezést, mint műveletet. Kis elemszámoknál természetesen a mai gépsebességek mellet nincs nagy jelentősége a rendezési időnek.

A három algoritmus hatékonyságának sorrendje (véletlenszerű elemek rendezésére):

1. Beszúrásos rendezés
2. Minimum kiválasztásos rendezés
3. Buborék rendezés