Vektorok - Egydimenziós tömbök

Programozási nyelvben először deklarálni kell a tömböt (**`típus név [méret]`**): megadjuk a tárolásra kerülő elemek *típusát* és a maximális elemszámot (*méret*). A címkézés módja C/C++ programozási nyelvben mindig 0-tól indul. Pszeudokódban a címkézés (számozás) 1-től n-ig szokott lenni, C/C++ programozási nyelvben 0-tól n-1-ig vannak számozva a pozíciók.

```
int n, i, v[50];
```

Nem kell mindig az összes elemmel dolgozni, erre kell az `n`, az aktuális elemszám (ezt kérdezzük meg az elején). A feldolgozáshoz szükség van ismétlő utasításokra (általában ez **`for`**), ezért kell egy ciklusváltozó, ez lesz az **`i`**.

A következő algoritmusokban ***`v`*** lesz a tömb és ***`n`*** lesz az aktuális elemszám.

Alapműveletek

1.) Tömb elemeinek a beolvasása

```
Beolvas n
Minden i←1,n-re végezd el
│   Beolvas v[i]
■
```

```
cout<< "n= ";cin>>n;
for(i=0;i<n;i++)
{
  cout<< " [ "<<i<< "]= ";
  cin>>v[i];
}
```

2.) Tömb elemeinek a kiíratása az eredeti sorrendben: a tömb elemeit illik egymástól szóközzel elválasztani vagy adott számú pozícióra kell kiíratni őket (`setw, iomanip`)

```
Minden i←1,n-re végezd el
│   Kiír v[i], ' '
■
```

```
for(i=0;i<n;i++)
  cout<<v[i]<< " ";
cout<< "\n ";
```

3.) Tömb elemeinek a kiíratása fordított sorrendben:

```
Minden i←n,1,-1 -re végezd el
│   Kiír v[i], ' '
■
```

```
for(i=n-1;i>=0;i--)
  cout<<v[i]<< " ";
cout<< "\n ";
```

4.) Tömb elemi közül a legkisebb – minimumkeresés: feltételezzük, hogy a tömb első eleme a legkisebb. Megvizsgáljuk a többi elemet is, és ha az eddigi minimumnál kisebbet találunk, akkor kicseréljük.

```
min ← v[1]
Minden i←2,n-re végezd el
│   Ha v[i] < min akkor
│   │         min ← v[i]
│   ■
■
Kiír min
```

```
min=v[0]; // vagy óriási nagy érték
for(i=1;i<n;i++)
  if (v[i]<min) min=v[i];
cout<<"a legkisebb elem = " << min;
```

5.) Tömb elemi közül a legnagyobb – maximumkeresés: feltételezzük, hogy a tömb első eleme a legnagyobb. Megvizsgáljuk a többi elemet is, és ha az eddigi maximumnál nagyobbat találunk, akkor kicseréljük.

```
max ← v[1]
Minden i←2,n-re végezd el
│   Ha v[i] > max akkor
│   │         max ← v[i]
│   ■
■
Kiír max
```

```
max=v[0]; // vagy nagyon kicsi érték
for(i=1;i<n;i++)
  if (v[i]>max) max=v[i];
cout<<"a legnagyobb elem = " << max;
```

6.) Tömb elemeinek az összege: számítsuk ki a tömb elemeinek az összegét.

```
s ← 0
Minden i←1,n-re végezd el
│      s ← s + v[i]
■
Kiír s
```

```
s=0;
for(i=0;i<n;i++)
   s=s+v[i];
cout<<"Elemek osszege = " << s;
```

7.) Darabszám: számoljuk össze mennyi páros érték van a tömb elemei között

```
db ← 0
Minden i←1,n-re végezd el
│   Ha v[i] páros akkor
│   │         db ← db + 1
│   ■
■
Kiír db
```

```
db=0;
for(i=0;i<n;i++)
  if (v[i]%2==0)
    db++;
cout<<"Paros ertekek szama: " << db;
```

8.) Kiválogatás: írd ki a tömb elemei közül azokat, amelyek 5-re vagy nullára végződnek

```
Minden i←1,n-re végezd el
│ Ha v[i] utolsó számjegye 5 vagy 0
│ │  akkor
│ │     Kiír v[i]
│ ■
■
```

```
cout<<"Utolso szj 5 vagy 0: ";
db=0;
for(i=0;i<n;i++)
  if ( (v[i]%10==5) ||
       (v[i]%10==0) )
    {
      cout << v[i] << " ";
      db++;
    }
if (db==0) cout << "Nincs ilyen";
cout << "\n";
```

9.) Lineáris keresés: keressük meg, melyik pozíción fordul elő az **x** a tömbben, vagy írjunk ki egy „Nem szerepel a tömbben” üzenetet

```
Beolvas x
p ← 0
Minden i←1,n-re végezd el
│   Ha v[i]=x akkor
│   │         p ← i
│   ■
■
Ha p=0 akkor
│   Kiír „Nem szerepel a tömbben”
│      különben Kiír p
■
```

```
cout<<"x = "; cin >> x;
p=-1;
for(i=0;i<n;i++)
   if (v[i]==x) p=i;
if (p==-1)
     cout<<"Nem szerepel! ";
    else
     cout << "A pozicioja: " << p;
```