Ohjelmoinnin perusteet R0027

5 op

Taulukot (Arrays)

Viopen luku 8

Miksi taulukoita?

```
public static void main(String[] args) {
   // esitellään muuttujat hyppytulosten taltioimista varten
   double hyppy1, hyppy2, hyppy3, hyppy4, hyppy5,
         hyppy6, hyppy7, hyppy8, hyppy9, hyppy10;

   Scanner lukija = new Scanner(System.in);

   // ensimmäisen hypyn tulos talteen
   System.out.println("Anna 1. hypyn tulos: ");
   hyppy1 = lukija.nextDouble();

   // toisen hypyn tulos talteen
   System.out.println("Anna 2. hypyn tulos: ");
   hyppy2 = lukija.nextDouble();

   // jne.
   ...
}
```

Taulukko

- Javassa tavallinen muuttuja voi sisältää yhden arvon
- Jos käytämme taulukkoa, voimme tallentaa siihen useamman arvon
- Taulukon yksittäisiin arvoihin päästään käsiksi **indeksien** avulla
- Taulukot ovat olioita

int-taulukko

- Taulukkomuuttujan määrittäminen

```
int[] munTaulukko;
```

- Taulukkomuuttujan alustaminen

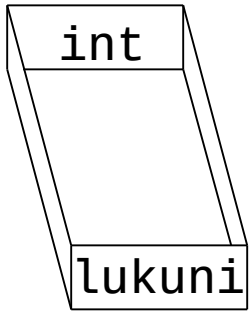
```
munTaulukko = new int[10];
```

- Taulukon solun käsitteleminen

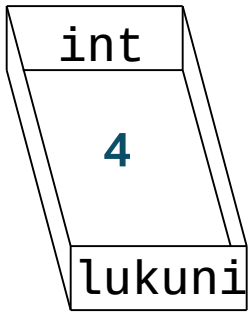
```
munTaulukko[7] = 2345;
System.out.println(munTaulukko[7]);
```

Tavallinen muuttuja tallelokerona

```
int lukuni;
```

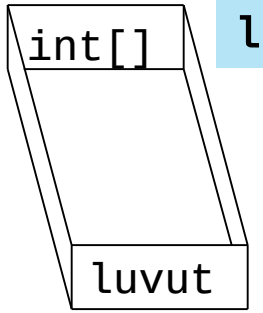



```
lukuni = 4;
```

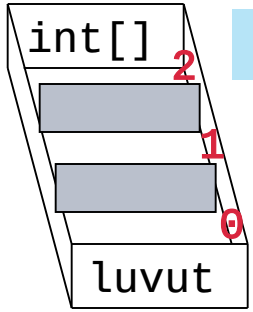


Taulukko tallelokerona

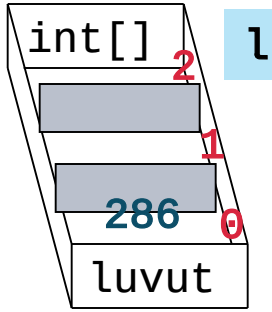
```
int[] luvut;
```



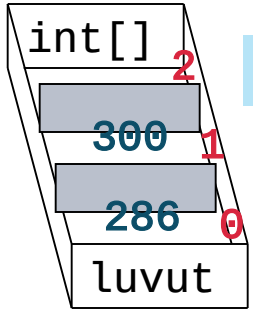

```
luvut = new int[3]
```



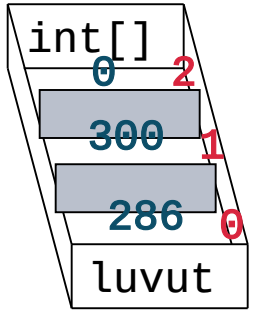

```
luvut[0] = 286;
```




```
luvut[1] = 300;
```




```
luvut[2] = 0;
```



Syntaksi

```
//tyyppi[] taulukkomuuttuja = new tyyppi[lkm];

double[] luvut = new double[3];
luvut[0] = 32.7;
luvut[1] = 22.59;
luvut[2] = 123.0;

String[] nimet = new String[2];
nimet[0] = "Pekka";
nimet[1] = "Matti";

//arvot voidaan asettaa myös heti alustuksen yhteydessä
String[] hedelmia = {"Omena", "Päärynä"};
int[] numeroita = {4,7,3,2,7,4,8,2,3,3,3,7,2};
```

Taulukon koko

- Taulukon pituuden saa näppärästi lauseella: `minunTaulukkoni.length`
- Tätä voidaan käyttää apuna esimerkiksi toistorakenteissa:

```
String[] kaupungit = {"Helsinki", "Kerava", "Kouvola"};

for(int i=0; i<kaupungit.length; i++) {
  System.out.println("Kaupunki " +i +": " +kaupungit[i]);
}
```

- Muista, että taulukon viimeinen arvo sijaitsee indeksillä (**length-1**)

Komentoriviparametrit

- Tee seuraavanlainen ohjelma, joka tulostaa päämetodin parametrina vastaanottaman taulukon ruudulle

```
public class Komentoriviparametrit {
    public static void main(String[] args) {
        for (int i = 0; i < args.length; ++i) {
            System.out.println(i+": " + args[i]);
        }
    }
}
```

- Aja tekemääsi ohjelmaa erilaisilla komentoriviparametreilla, esim.
  c:\> java Komentoriviparametrit Hauki on kala.

Merkkijono char-taulukoksi

```
String tekstini = "jotain";
char[] kirjaimeni = tekstini.toCharArray();

for(int i=0; i<kirjaimeni.length; i++) {
   System.out.println(kirjaimeni[i]);
}
```

```
j
o
t
a
i
n
```

Taulukon yksinkertainen vaihtojärjestäminen (Algoritmi)

- Ensimmäistä alkiota verrataan muihin alkioihin. Alkiot vaihdetaan, jos ne ovat väärässä järjestyksessä.
- Alkioista pienin (suurin) tulee ensimmäiseksi.
- Toista alkiota verrataan muihin alkioihin. Jälleen alkiot vaihdetaan, jos ne ovat väärässä järjestyksessä.
- Nyt taulukossa on toisena toiseksi pienin (suurin) alkio. Ensimmäisenähän oli pienin (suurin) alkio.
- Siirrytään vertaamaan kolmatta alkiota muihin jne. Puuha jatkuu kunnes toiseksi viimeinen alkio taulukosta on saatu käsiteltyä. Tämän jälkeen taulukon alkiot ovat nousevassa (laskevassa) järjestyksessä.

Taulukon yksinkertainen vaihtojärjestäminen

```
private static void vaihtojärjestä(double[] taulu) {

    // käydään läpi alkiot ensimmäisestä toiseksi viimeiseen
    for (int i = 0; i < taulu.length-1; ++i) {
        // käydään läpi i:ttä alkiota seuraavat
        for (int j=i+1; j < taulu.length; ++j) {

            // jos alkiot väärässä järjestyksessä
            if (taulu[i] < taulu[j]) {
                // i:s ja j:s alkio vaihtaa keskenään paikkaa
                double tmp = taulu[i];
                taulu[i] = taulu[j];
                taulu[j] = tmp;
            }

        }
    }

}
```

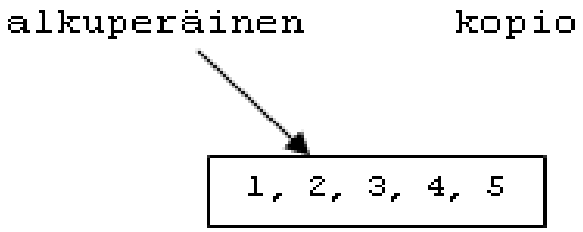
Taulukoiden kopioiminen

```
int alkuperäinen = 10, kopio;
// alkuperäinen == 10, kopio:lla ei arvoa

kopio = alkuperäinen;
// alkuperäinen == 10 ja kopio == 10

kopio++;
// alkuperäinen == 10 ja kopio == 11
```

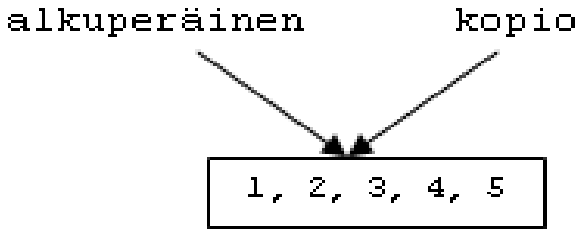
- Taulukko on olio, eikä alkeismuuttuja.
- Jos taulukon kopioi samalla tavalla kuin alkeismuuttujan, kopioituu vain viite olioon.
- Oliosta ei tule siis toista kopiota
- Molemmat viitteet osoittavat samaan olioon
- Jos toista muuttaa, toinenkin muuttuu

Taulukon kopioiminen

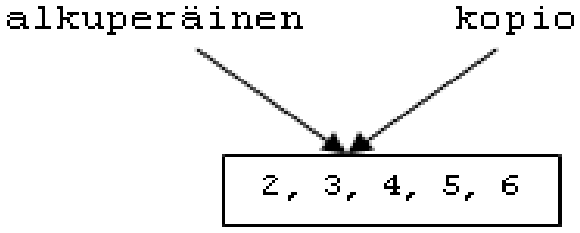
```
int[] alkuperäinen = {1, 2, 3, 4, 5};
int[] kopio;
```




```
kopio = alkuperäinen;
```



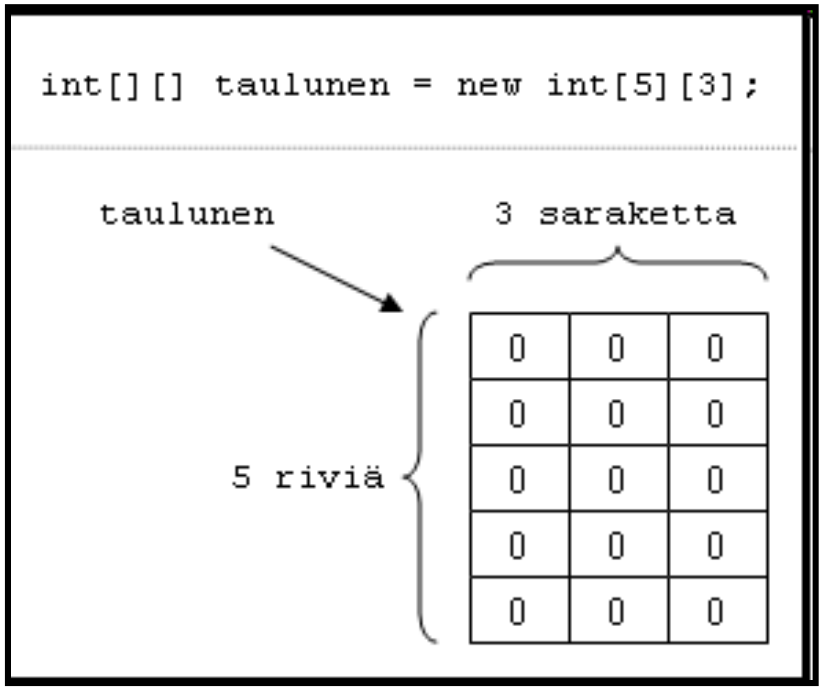

```
for (int i = 0; i < kopio.length; ++i)
    kopio[i]++;
```



Taulukon kopioiminen

```
private static int[] kopioTaulukosta(int[] ap) {
    // kopion kooksi sama kuin alkuperäisen
    int[] kopio = new int[ap.length];

    // kopiointi
    for (int i=0; i < ap.length; ++i) {
        kopio[i] = ap[i];
    }
    // palautetaan viite kopioon
    return kopio;
}
```

Moniulotteiset taulukot




```
taulunen[0][0] = 76;
taulunen[0][1] = 95;
taulunen[0][2] = 23;
taulunen[1][0] = 74;
taulunen[1][1] = 41;
taulunen[1][2] = 60;
taulunen[2][0] = 7;
taulunen[2][1] = 19;
taulunen[2][2] = 89;
taulunen[3][0] = 37;
taulunen[3][1] = 55;
taulunen[3][2] = 32;
taulunen[4][0] = 88;
taulunen[4][1] = 31;
taulunen[4][2] = 4;
```

Moniulotteiset taulukot
Rivien ja sarakkeiden pituus

- Rivien määrä
  - taulunen.length
- Sarakkeiden määrä
  - taulunen[**i**].length