JAVAUTVECKLING – LEKTION 8

Mahmud Al Hakim
mahmud.al.hakim@nackademin.se
www.alhakim.se

AGENDA

Fält (Arrayer)

Att skapa och arbeta med fält

Indexering

Jämförelse av fält

Flerdimensionella fält

VAD ÄR ETT FÄLT (ARRAY)

Inom datavetenskap är **ett fält** eller **en array** en datastruktur som består av en samling av element (av en viss storlek och ofta samma datatyp), som identifieras med ett eller flera heltaliga index och lagrade i ett sammanhängande minnesblock så att adressen till varje element enkelt kan räknas ut från dess index.

Till skillnad från en vanlig variabel som bara innehåller ett värde kan ett fält alltså innehålla ett godtyckligt antal värden.

Källa: https://sv.wikipedia.org/wiki/F%C3%A4lt_(datastruktur)

ATT SKAPA ETT FÄLT

```
typ[] namn;              // deklarera en fältvariabel
namn = new typ[n]        // skapa utrymme för fältet
```

Ett alternativt skrivsätt

```
typ[] namn = {värde0, värde1, värde2, … }
```

Längden på ett fält ges av uttrycket

```
namn.length
```

ATT SKAPA ETT FÄLT – EXEMPEL

```
double[] f1 = new double[4];
// Eller
double[] f1 = {0, 0, 0, 0};

int[] f2 = {5, 10, 20, 50, 100, 200, 500};
```

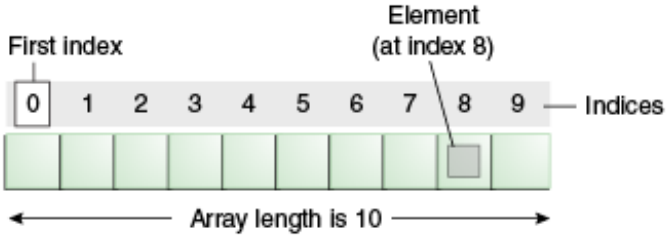
INDEXERING

Komponenterna i ett fält numreras från 0 och uppåt.

Man kan välja ut en enstaka komponent genom att indexera

`fältnamn[i]`

Indexet `i` måste ligga i intervallet 0 till `fältnamn.length-1`



Bildkälla https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html

INDEXERING – EXEMPEL

```
int[] tabell = new int[10];
for (int i=0 ; i < tabell.length ; i++)
    tabell[i] = i+1;
```

ATT SKRIVA UT FÄLT

```
for (int i=0 ; i < f1.length ; i++)
      System.out.println(f1[i]);
```

Eller

```
for (int i : f2)
    System.out.println(i);
```

ÖVNING

Skapa en metod som skriver ut komponenterna i ett fält.

```
public static void skrivUt(double[] a) {
    for (double d : a)
      System.out.println(d);
}
```

ÖVNING 11.2

Skriv ett program som först skapar ett fält med tio komponenter av typen double.

Programmet skall sedan innehålla en for-sats som ger den första komponenten värdet 1/1, den andra värdet 1/2, den tredje värdet 1/3 osv.

Sist i programmet skall komponenternas värden skrivas ut.

ÖVNING 11.2 – FACIT

```
public class FaltOvnning {
  public static void main(String[] arg) {
    double[] a = new double[10];
    for (int i=0; i<a.length; i++)
      a[i] = 1.0/(i+1);   // Obs! 1/(i+1) hade givit heltalsdivision

    for (double d : a)
      System.out.println(d);
  }
}
```

ÖVNING

Skapa en metod som beräknar medelvärdet av komponenterna i ett fält.

```
public static double medelv(double[] a) {
    double sum = 0;
    for (double x : a)
      sum = sum + x;
    return sum/a.length;
}
```

UPPGIFT 11.3

```
public static boolean sorterat(double[] f, int antal)
{
    for (int i=1 ; i < antal; i++)
      if (f[i] < f[i-1])
        return false;
    return true;
}
```

Behöver vi verkligen skicka antal?

JÄMFÖRELSE AV FÄLT

Om man gör en vanlig jämförelse jämförs referenserna till fälten:

```
g == f // lika endast om g och f pekar på samma fält
```

Vill man jämföra själva fälten kan man använda en for-sats och jämföra ett par av komponent i taget.

Alternativt. Använd standradmetoden Arrays.equals i paketet java.util:

```
Arrays.equals(f,g)
```

UPPGIFT 11.5

```
public static boolean equals(double[] a, double[] b) {
    if (a == null && b == null)
      return true;             // två tomma referenser betraktas som lika
    else if (a == null || b == null)
      return false;            // en referens är tom men inte den andra
    else if (a.length != b.length)
      return false;            // olika långa fält
    for (int i=0; i<a.length; i++)
      if (a[i] != b[i])
        return false;
    return true;
}
```

UPPGIFT 11.6 DEL 1

```
public static double sum(double[] f) {
   double sum = 0;
   for (int i=0; i<f.length; i++)
       sum += f[i];
   return sum;
 }
```

UPPGIFT 11.6 DEL 2

```
public static double kvadSum(double[] f) {
   double sum = 0;
   for (int i=0; i<f.length; i++)
       sum += f[i] * f[i];
   return sum;
 }
```

FLERDIMENSIONELLA FÄLT

```
typ[][] namn;              // deklarera en fältvariabel
namn = new typ[n][m];      // skapa utrymme för fältet
```

n är antalet rader och m är antalet kolumner.

Ett alternativt skrivsätt

```
typ[][] namn = { {värde00, värde01, värde02, … },
                 {värde10, värde11, värde12, … },
                 ...
               };
```

FLERDIMENSIONELLA FÄLT – EXEMPEL

```
String[][] names = {
    {"Mr. ", "Mrs. ", "Ms. "},
    {"Smith", "Jones"}
};
// Mr. Smith
System.out.println(names[0][0] + names[1][0]);
// Ms. Jones
System.out.println(names[0][2] + names[1][1]);
```

Exemplet är hämtat från
https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html

ÖVNING 12.1

```
int[][] tab = new int[10][10];
    for (int i=0; i<10; i++)
      for (int j=0; j<10; j++)
        tab[i][j] = (i+1)*(j+1);

    for (int i=0; i<10; i++) {
      for (int j=0; j<10; j++)
        System.out.print("\t" + tab[i][j]);
      System.out.println();
    }
```