Javaprogrammering

Del 2 - if-satser, arrayer och loopar

Repetition

- Datatyper
- Variabler
- Objektorienterad programmering
  - Klasser, objekt, metoder
- System.out.Print för att printa, Scanner för indata
- If-satser

Jämförelser mellan element

- Jämför om två värden är samma
  - `1 == 1 => true`
  - `1 == “1” => error`
  - `10 == 100 => false`
  - `“Hej”.equals(“hej”) => false`
  - `“halloj”.equals(“halloj”) => true`

- “Jämför om ett element är större/mindre än ett annat
  - `3 > 5 => false`
  - `5 > 3 => true`
  - `“Sträng” < 100 => error`
-

If-satser

För att göra olika saker beroende på jämförelser av värden i vårt program så använder vi if-satser.

Exempel:

```
Scanner reader = new Scanner(System.in);
System.out.println("Hur gammal är du? : ");
int age = reader.nextInt();

if (age > 17) {
    System.out.println("Välkommen in");
} else if (age > 14){
    System.out.println(“Du får åka moppe, men inte komma in på klubben”);
} else {
    System.out.println("Tyvärr, du är inte gammal nog");
}
```

If-satser

Ibland vill man kontrollera fler saker samtidigt i en if-sats

```
int age = 23;
String name = “John”;

if (age > 17 && name == “John”) {
    System.out.println("Du är vuxen och heter John");
} else {
    System.out.println("Du är inte en vuxen person som heter John");
}
```

Äventyrsspel

Med hjälp av if-satser kan vi skapa äventyrsspel där vi frågar användaren vad den vill göra:

```
if (answer.equals(“ja”)) {
     System.out.println("Du är nu i huset. Vill du öppna källardörren?");
     answer = scanner.nextLine();
     if (answer.equals(“ja”)) {
          System.out.println(“Ett monster attackerar dig”);
     } else {
          System.out.println(“Du överlevde. Fort ut ur huset nu!”);
     }
} else {
     System.out.println("Du gick inte in i huset, tur du överlevde!");
}
```

Uppgift - Gissningsspel

Med hjälp av Random och if-satser så kan vi göra ett spel där användaren får gissa ett slumptal, och låta användaren veta om det var rätt eller fel svarat. Försök själv!

- Skapa en klass: Gissningsspel.java
  - Skapa en main-metod
  - I main-metoden:
    - Generera ett slumptal (importera java.util.Random eller använd Math.random()) och spara undan i en variabel
    - Importera och skapa en Scanner för att kunna få input
    - Be användaren gissa på slumptalet
    - Printa ‘grattis’ om det var rätt, säg annars att det var fel gissat, och printa det rätta svaret

Arrayer []

Om vi vill spara undan många variabler på ett ställe så borde vi skapa någon form av lista. Det kan till exempel vara en kö med människor, där vi vill hälsa på en människa i taget. Om vi har allt sparat i en ‘lista’ eller Array så kan vi enkelt hålla reda på alla människor.

Det kan också vara en lista med alla spelare i ett dataspel (typ Minecraft) eller siffror för nån statistik osv.

Vi skapar en array för heltal genom att skriva:

```
int[] minFörstaArray = {1, 2, 3};
```

minFörstaArray är förstås variabelnamnet och kan väljas själv.

En array med strängar[]

En array kan innehålla vad som helst, men alla element måste vara av samma typ

```
String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
```

Fungerar!

```
String[] cars = {"Volvo", 740, "Ford", "Mazda"};
```

Fungerar ej!!!

Arrayer [] - få tag på värden

För att lägga värden i arrayen så skriver vi som sagt:

```
int[] minFörstaArray = {10, 20, 30, 40, 50};
```

Sen kan vi få tag på (och printa) varje värde i arrayen genom att skriva minFörstaArray[position]; (position börjar räkna från 0):

```
System.out.println(minFörstaArray[0]);
System.out.println(minFörstaArray[1]);
System.out.println(minFörstaArray[2]);
System.out.println(minFörstaArray[3]);
System.out.println(minFörstaArray[4]);
```

Arrayer [] - få längden av en array

```
int[] minAndraArray = {10, 20, 30, 40, 50};
```

För att få längden av en array så skriver vi:

```
minAndraArray.length;
```

Eller om vi vill printa den:

```
System.out.println(minAndraArray.length);
```

Arrayer [] - förlänga en array går ej

Går tyvärr inte i en array, om vi vill lägga till fler värden än vi har plats för så måste vi använda oss av något annat än en array eller skapa en ny större array.

Om man vill kunna förlänga en lista av värden så får man använda något som heter ArrayList istället.

Arrayer [] - ändra värden

För att lägga värden i arrayen så skriver vi:

```
int[] minFörstaArray = {10, 20, 30, 40, 50};
```

Det smidiga sen är att vi kan få tag på (och printa) varje värde i arrayen genom att skriva minFörstaArray[position] (position börjar räkna från 0):

```
minFörstaArray[0] = 100;
System.out.println(minFörstaArray[0]);
```

Arrayer [] - göra om en mening till en array

Om vi har en mening “Hej jag heter Joakim” och vill läsa varje ord för sig så kan vi helt enkelt dela upp meningen i en array med alla ord.

```
String mening = “Hej detta är en mening med flera ord”;
String[] ordarray = mening.split();
```

Arrayer [] - gå igenom hela arrayen med for-loop

Istället för att manuellt skriva `minFörstaArray[0]`, `minFörstaArray[1]`, `...`
Så måste det väl finnas ett bättre sätt? Ja! For-loopar

```
for (int siffra : minFörstaArray) {
    System.out.println(siffra);
}
```

Viktigt koncept! Används för att gå igenom arrayer och andra typer av listor

For-loopar med siffror

En for-loop kan vara användbar för andra saker. Om vi vill utföra något 100 gånger så kommer vi inte orka skriva in 100 rader för varje gång den gör det, det är onödigt. For-loopar gör att vi kan automatisera det:

Startvärde Slutvärde Gå ett steg i taget

```
for (int i = 0; i < 100; i++) {
    System.out.println(“Hej! ” + i);
}
```

Vad gör for-loopen ovanför tror ni?

For-loop som hoppar fler steg i taget

En for-loop behöver inte hoppa ett steg i taget, den behöver inte ens hoppa framlänges.

```
for (int i = 0; i < 10; i=i+2) {
    System.out.println(“Hej! ” + i);
}

for (int i = 10; i > 0; i=i-2) {
    System.out.println(“Hej! ” + i);
}
```

Vad gör for-looparna ovanför tror ni?

While-loopar

En for-loop kunde utföra något ett visst antal gånger, men tänk om vi inte vet hur många gånger vi kommer utföra något? Då kan vi använda en While-loop:

Ser ut exempelvis så här:

```
while (age < 100) {
    // Upprepa något tills age inte är under 100 längre
}
```

While-loop, mer verkligt exempel

```
Scanner reader = new Scanner(System.in);
System.out.println("Hej! Skriv in massa tal så plussar jag ihop dom.
Skriv 0 om du vill avsluta programmet. ");
int tal = reader.nextInt();
int resultat = tal;

while (tal != 0) {
    tal = reader.nextInt();
    resultat = resultat + tal
    System.out.println(“Resultat hittills: ” + resultat);
}
```

Arrayer [] - 2d-array

En array är endimensionell, bara en lista, men om vi vill representera t.ex. ett 3 i rad-bräde så behöver vi ett rutnät.

Vi kan göra 2d-arrayer!

För att skapa en 2d-array så skriver vi:

```
int[][] minFörsta2DArray = {{0, 0, 0},
                            {0, 0, 0},
                            {0, 0, 0}};
```

Om vi vill hämta det första elementet på första raden så skriver vi:

```
minFörstaArray[0][0];
```

ArrayList, en smartare array

- Arrayer var smidiga listor som gick att loopa igenom men dom hade nackdelar, går inte att lägga till element och kan inte göra så mkt
- ArrayList är smartare, den kan fyllas på, element kan raderas och den har många inbyggda metoder som inte array har
- Måste importeras (`import java.util.ArrayList;`)

```
ArrayList<String> politiker = new ArrayList<String>();
politiker.add("Olof Palme");
politiker.add("Göran Person");
politiker.add("Annie Lööf");
politiker.add("Fredrik Reinfeldt");
System.out.println(minLista);
```

ArrayList, contains

```
ArrayList<String> politiker = new ArrayList<String>();
politiker.add("Olof Palme");
politiker.add("Göran Person");
politiker.add("Annie Lööf");
politiker.add("Fredrik Reinfeldt");

if(politiker.contains(“Stefan Löfven”) {
    System.out.println(“Stefan Löfven är med i listan”);
} else {
    System.out.println(“Stefan Löfven är inte med i listan”);
}
```

Gör quizet

https://www.w3schools.com/java/exercise.asp?filename=exercise_arrays1

Sten Sax Påse - vi gör det tillsammans!

- Skapa ett nytt projekt i IntelliJ StenSaxPase och en klass StenSaxPase.java.

Uppgifter - försök själv först, sen tillsammans

- Summera (plussa ihop) alla tal från 0 till och med 1000 och printa resultatet
  - Skapa en klass (Plussare.java) där du har en for-loop i main-metoden som går igenom alla tal från 0 till 1000
  - Du kommer behöva en variabel där du kan spara resultatet under varje ‘varv’ i loopen
- Skapa en nedräknare för en raket (använd loop)
  - Programmet ska printa från 10 till 0 under 10 sekunder och sen ska den printa “LIFT OFF!”
  - Skapa en klass “Uppskjutning.java” med en main-metod där du utför uppskjutningen
  - För att pausa en sekund mellan varje siffra i uppskjutningen använd TimeUnit.SECONDS.sleep(1); (du måste importera TimeUnit)
- Skapa ett gissningsspel (Gissningsspel.java)
  - Klassen genererar ett hemligt slumptal i main-metoden
  - Användaren gissar på talet med hjälp av en Scanner
  - Programmet svarar om det hemliga talet är större eller mindre än gissningen
  - Upprepa steg b och c tills användaren gissat rätt (while-loop!)

Rövarspråket - vi gör det tillsammans