

DV017A  
Java-programmering  
Laboration 1

Jonas Sjöberg  
860224  
Högskolan i Gävle,  
Elektronikingenjörsprogrammet,  
`tel12jsg@tudent.hig.se`

Datum: Juni 2015  
Kursansvarig lärare: Atique Ullah

**Sammanfattning**

Laboration i DV017A - Inledande programmering i Java. Blandade uppgifter i grundläggande praktisk programmering i Java. Innehåller lärares instruktioner samt egna lösningar, källkod, skärmdumpar och egna kommentarer.

# Innehåll

<b>1</b>	<b>Uppgift 1</b>	<b>4</b>
1.1	Instruktioner . . . . .	4
1.2	Lösning . . . . .	4
1.2.1	Kommentar . . . . .	4
1.2.2	Källkod . . . . .	5
1.2.3	Skärmdump . . . . .	6
<b>2</b>	<b>Uppgift 2</b>	<b>6</b>
2.1	Instruktioner . . . . .	6
2.2	Lösning . . . . .	7
2.2.1	Funktion . . . . .	7
2.2.2	Kommentar . . . . .	7
2.2.3	Källkod . . . . .	7
2.2.4	Skärmdump . . . . .	8
<b>3</b>	<b>Uppgift 3</b>	<b>8</b>
3.1	Instruktioner . . . . .	8
3.2	Lösning . . . . .	9
3.2.1	Kommentar . . . . .	9
3.2.2	Källkod . . . . .	9
3.2.3	Skärmdump . . . . .	10
<b>4</b>	<b>Uppgift 4</b>	<b>11</b>
4.1	Instruktioner . . . . .	11
4.2	Lösning . . . . .	11
4.2.1	Funktion . . . . .	11
4.2.2	Kommentar . . . . .	11
4.2.3	Källkod . . . . .	11
4.2.4	Skärmdump . . . . .	12
<b>5</b>	<b>Uppgift 5</b>	<b>12</b>
5.1	Instruktioner . . . . .	12
5.2	Lösning . . . . .	13
5.2.1	Funktion . . . . .	13
5.2.2	Källkod . . . . .	13
5.2.3	Skärmdump . . . . .	15
<b>6</b>	<b>Uppgift 6</b>	<b>15</b>
6.1	Instruktioner . . . . .	15
6.2	Lösning . . . . .	15
6.2.1	Funktion . . . . .	15
6.2.2	Kommentar . . . . .	16
6.2.3	Källkod . . . . .	16
6.2.4	Skärmdump . . . . .	18

<b>7</b>	<b>Uppgift 7</b>	<b>18</b>
7.1	Instruktioner . . . . .	18
7.2	Lösning . . . . .	19
7.2.1	Funktion . . . . .	19
7.2.2	Kommentar . . . . .	19
7.2.3	Källkod . . . . .	19
7.2.4	Skärmdump . . . . .	21
<b>8</b>	<b>Uppgift 8</b>	<b>22</b>
8.1	Instruktioner . . . . .	22
8.2	Lösning . . . . .	22
8.2.1	Funktion . . . . .	22
8.2.2	Kommentar . . . . .	22
8.2.3	Källkod . . . . .	22
8.2.4	Skärmdump . . . . .	24
<b>9</b>	<b>Uppgift 9</b>	<b>24</b>
9.1	Instruktioner . . . . .	24
9.2	Lösning . . . . .	24
9.2.1	Funktion . . . . .	24
9.3	Kommentar . . . . .	24
9.3.1	Källkod . . . . .	24
9.3.2	Skärmdump . . . . .	26
<b>10</b>	<b>Uppgift 10</b>	<b>26</b>
10.1	Instruktioner . . . . .	26
10.2	Lösning . . . . .	27
10.2.1	Funktion . . . . .	27
10.2.2	Kommentar . . . . .	27
10.2.3	Källkod . . . . .	27
10.2.4	Skärmdump . . . . .	29
<b>11</b>	<b>Referenser</b>	<b>30</b>
11.1	www . . . . .	30
11.2	Literature . . . . .	30
11.3	Source files . . . . .	30

# 1 Uppgift 1

## 1.1 Instruktioner

1. I nedanstående program saknas datatyperna (där det står ...) vid variabeldeklarationerna. Din uppgift är att fylla i rätt datatyp vid respektive deklaration. Provkör sedan programmet:

```
public class Datatyper
{
    public static void main(String[] args)
    {
        ... data1 = true;
        ... data2 = 45.8F;
        ... data3 = 29;
        ... data4 = data3 < 10;
        ... data5 = 12 / 5;
        ... data6 = data3 * data5;
        ... data7 = 10 % 3;
        ... data8 = "Java programmering";
        ... data9 = 'b';
        ... data10 = (float)data5 / 4;

        System.out.println    ("Variabeln    data1: " + data1);
        System.out.println    ("Variabeln    data2: " + data2);
        System.out.println    ("Variabeln    data3: " + data3);
        System.out.println    ("Variabeln    data4: " + data4);
        System.out.println    ("Variabeln    data5: " + data5);
        System.out.println    ("Variabeln    data6: " + data6);
        System.out.println    ("Variabeln    data7: " + data7);
        System.out.println    ("Variabeln    data8: " + data8);
        System.out.println    ("Variabeln    data9: " + data9);
        System.out.println    ("Variabeln    data10: " + data10);
    }
}
```

## 1.2 Lösning

### 1.2.1 Kommentar

Det är något oklart vad som efterfrågas. I problembeskrivningen specificeras inte huruvida målvariabelns datatyp ska vara lämpad för att lagra resultatet av beräkningen på det vis att ingen data går förlorad genom trunkering eller avrundning. Eller om målvariabeln helt enkelt ska ha samma datatyp som de övriga operanderna. Jag har valt att vara konservativ i fråga om resurser och använder de datatyper som krävs för att lagra den information som ska behandlas. Om exekveringstid och/eller lagringsutrymme inte är någon faktor att beakta kunde alla variabler få vara 64-bitars floats.

Nämnvärt är att en inmatad siffra utan specificerad datatyp ges typen integer per default. Detta är fallet för `data5 = 15/5 ; 12` och 5 har typen `int` och resultatet innehåller inga decimaler. Detaljer rörande 'literals' hämtas bäst från The Java Language Specification, Java SE 8 Edition

## 1.2.2 Källkod

```

1  /*
2  *  DV017A :: Grundläggande programmering i Java
3  *  =====
4  *  Uppdaterad 2015-06-09
5  *  Jonas Sjöberg 860224-7614
6  *  Högskolan i Gävle.
7  *  <tel12jsg@student.hig.se>
8  *
9  *  Labb #1
10 *  Uppgift 1
11 */
12
13 public class Lab1Uppg01 {
14     public static void main(String[] args) {
15         boolean data1 = true;
16         float data2 = 45.8F;
17         int data3 = 29;
18         boolean data4 = data3 < 10;
19
20         /* Även om data5 skulle vara av typen double som klarar decimaltal,
21          * så är 12 och 5 båda av typen int per default och decimaldelen
22          * går förlorad. Därmed kan data5 ha typen 'int' med lika verkan. */
23         int data5 = 12 / 5;
24
25         /* Då varken data3 eller data5 är decimaltal kan 'int' också användas
26          * utan att någon information går förlorad. */
27         double data6 = data3 * data5;
28
29         int data7 = 10 % 3;
30         String data8 = "Java programmering";
31         char data9 = 'b';
32         float data10 = (float) data5 / 4;
33
34         System.out.println("Variabeln data1: " + data1
35                             + "\nVariabeln data2: " + data2
36                             + "\nVariabeln data3: " + data3
37                             + "\nVariabeln data4: " + data4
38                             + "\nVariabeln data5: " + data5
39                             + "\nVariabeln data6: " + data6
40                             + "\nVariabeln data7: " + data7
41                             + "\nVariabeln data8: " + data8
42                             + "\nVariabeln data9: " + data9
43                             + "\nVariabeln data10: " + data10);
44     }

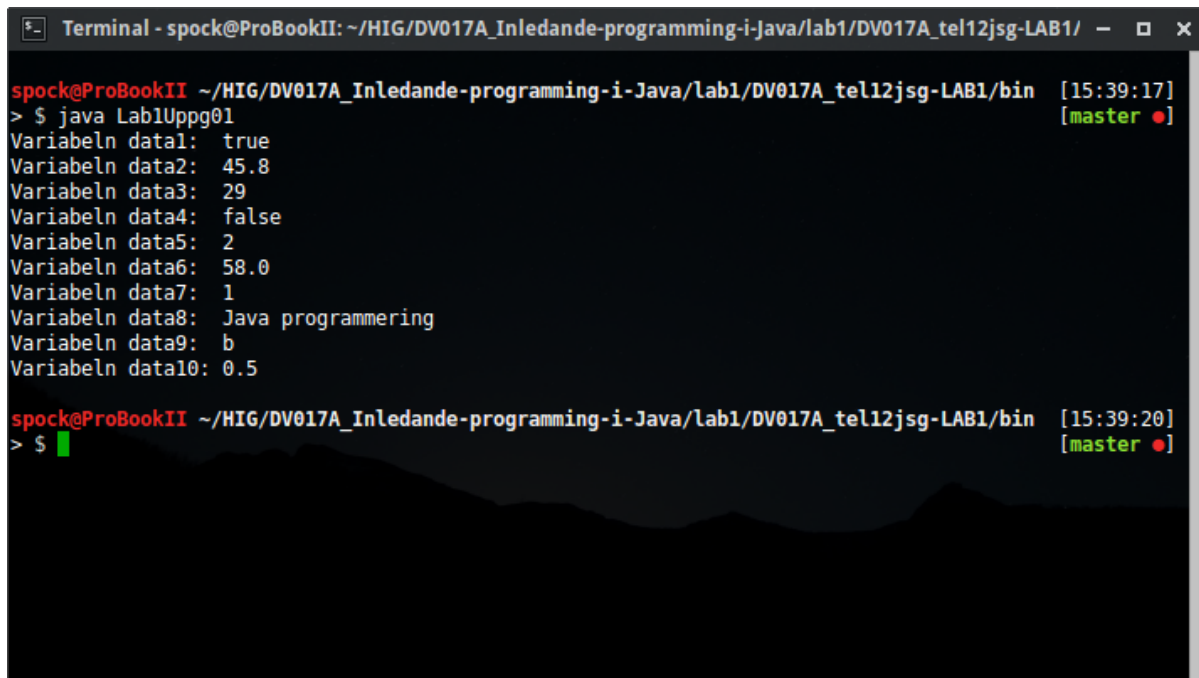
```

---

<sup>1</sup><https://docs.oracle.com/javase/specs/>

Lab1Uppg01.java

### 1.2.3 Skärmdump



```
Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/ [15:39:17]
spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [master ●]
> $ java Lab1Uppg01
Variabeln data1: true
Variabeln data2: 45.8
Variabeln data3: 29
Variabeln data4: false
Variabeln data5: 2
Variabeln data6: 58.0
Variabeln data7: 1
Variabeln data8: Java programming
Variabeln data9: b
Variabeln data10: 0.5

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:39:20]
> $
```

Figur 1: Körning av koden till Uppgift 1

## 2 Uppgift 2

### 2.1 Instruktioner

2. Skriv ett program som skriver ut summan, medelvärdet och produkten av tre heltal. De tre heltalen ska användaren skriva in från tangentbordet när programmet körs. Programmets utskrift kan t.ex se ut så här, det som skrivs in från tangentbordet är markerat med fetstil/understrykning:

```
Skriv in tre heltal.
Skriv in det första talet: *20*
Skriv in det andra talet: *30*
Skriv in det tredje talet: *25*
Summan av talen är 75.
Medelvärdet av talen är 25.
Produkten av talen är 15000.
```

## 2.2 Lösning

### 2.2.1 Funktion

Programmet använder `static final` i början av programmet för variabler som inte ska ändras under exekvering. Variabler med enbart versaler är en slags globala variabler som kan modifieras av programmeraren.

Räkneord sparas i sträng-arrays `GRUNDTAL` och `ORDNINGSTAL`, för att hämtas och skrivas ut vid exekvering.

### 2.2.2 Kommentar

Rad 33, `@SuppressWarnings(resource)` syftar till att undertrycka varningar från Eclipse och gör ingen faktisk skillnad på programmets funktion.

Programmet saknar filtrering av indata, vilket är ett stort problem då det innebär stora säkerhetsrisker och möjliga odefinierade, alternativt utforskade skeenden. Mer om detta senare..

### 2.2.3 Källkod

```
1  /*
2  *  DV017A :: Grundläggande programmering i Java
3  *  =====
4  *  Uppdaterad 2015-06-09
5  *  Jonas Sjöberg 860224-7614
6  *  Högskolan i Gävle.
7  *  <tel12jsg@student.hig.se>
8  *
9  *  Labb #1
10 *  Uppgift 2
11 */
12
13 import java.util.Scanner;
14
15 public class Lab1Uppg02 {
16
17     /* Konstant värde, typ #define i C-liknande språk med en pre-processor.
18      *  ITERATIONS är antalet tal som ska matas in. Konstant vid exekvering. */
19     private static final int ITERATIONS = 3;
20
21     /* Svenska Räkneord i string-arrays. Arrayindex överensstämmer med talet. */
22     private static final String[] GRUNDTAL = { "noll", "ett", "två", "tre",
23                                                 "fyra", "fem", "sex", "sju",
24                                                 "åtta", "nio", "tio", "osv" };
25
26     private static final String[] ORDNINGSTAL = { "nollte", "första", "andra",
27                                                  "tredje", "fjärde", "femte",
28                                                  "sjätte", "sjunde", "åttonde",
29                                                  "nionde", "tionde", "osv" };
30
31
32     public static void main(String[] args) {
```

```

33      /* Deklarera variabler och nytt Scanner-objekt för att läsa I/O. */
34      @SuppressWarnings("resource")
35      Scanner scan = new Scanner(System.in);
36      int input, sum = 0;
37      double average = 0;
38      long product = 1; /* Kan räkna med att produkten kan bli väldigt stor */
39
40      /* Fråga efter 'ITERATIONS' antal tal. Pussla ihop strängar och hämta
41       * svenska räkneord från arrayen 'GRUNDTAL'. */
42      System.out.println("Skriv in " + GRUNDTAL[ITERATIONS] + " heltal.");
43
44      /* Loopa 'ITERATIONS' gånger, fråga efter talen i sekventiell följd.
45       * Om noll heltal efterfrågas avslutas programmet direkt. */
46      for (int i = 1; i < ITERATIONS + 1; i++) {
47          System.out.print("Skriv in det " + ORDNINGSTAL[i] + " talet: ");
48
49          input = scan.nextInt();
50
51          sum += input;
52          average = sum / i;
53          product = product * input;
54      }
55
56      /* Skriv ut resultatet till stdout. */
57      System.out.println("Summan av talen är " + sum
58                          + "\nMedelvärde av talen är " + average
59                          + "\nProdukten av talen är " + product);
60  }
61 }

```

Lab1Uppg02.java

## 2.2.4 Skärmdump

# 3 Uppgift 3

## 3.1 Instruktioner

3. Skriv ett program som räknar om mil till kilometer. Användaren ska mata in antal mil som ett decimaltal. Sedan ska programmet konvertera till kilometer och skriva ut hur många kilometer det blev. Exempel på utskrift, användarens inmatning är markerat med fetstil/understrykning:

```

Program som konverterar mil till km.
Skriv in antal mil: *35.4*
Motsvarande antal km: 354

```



```
Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/ - □ ×

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:44:47]
> $ java Lab1Uppg02 [master ●▲]
Skriv in tre heltal.
Skriv in det första talet: 6
Skriv in det andra talet: 323
Skriv in det tredje talet: 1337
Summan av talen är 1666
Medelvärdet av talen är 555.0
Produkten av talen är 2591106

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:45:19]
> $ [master ●▲]
```

Figur 2: Körning av koden till Uppgift 2

## 3.2 Lösning

### 3.2.1 Kommentar

Det enda anmärkningsvärda här är hur texten skrivs ut med `System.out.format()` för att få finare kontroll över formateringen. Inkluderat i kommentarerna på raderna 2834 är delar av dokumentationen från Oracle <sup>2</sup> på de funktionerna i biblioteket (paketet) som använts.

### 3.2.2 Källkod

```
1  /*
2   * DV017A :: Grundläggande programmering i Java
3   * =====
4   * Uppdaterad 2015-06-09
5   * Jonas Sjöberg 860224-7614
6   * Högskolan i Gävle.
7   * <tel12jsg@student.hig.se>
8   *
9   * Labb #1
10  * Uppgift 3
11  */
12
13  import java.util.Scanner;
14
15  public class Lab1Uppg03 {
16
17      public static void main(String[] args) {
```

<sup>2</sup><https://docs.oracle.com/javase/tutorial/java/data/numberformat.html>

```

18      /* Skapa Scanner-objekt för att läsa I/O. Ignorera IDE:ns varningar. */
19      @SuppressWarnings("resource")
20      Scanner scan = new Scanner(System.in);
21
22      System.out.println("Program som konverterar mil till km.");
23
24      System.out.print("Skriv in antal mil: ");
25      double mil = scan.nextDouble();
26
27      double kilometer = mil * 1000;
28
29      /* Skriv ut med 'format', ekvivalent med 'printf' som i sin tur känns
30       * igen från C-liknande språk.
31       *
32       * %.0f      Float with zero places after decimal point.
33       * %n        A new line character appropriate to the platform running the
34       *            application. You should always use %n, rather than \n.
35       */
36      System.out.format("Motsvarande antal kilometer: %.0f%n", kilometer);
37  }
38  }

```

Lab1Uppg03.java

### 3.2.3 Skärmdump

```

Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/
spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:47:07]
> $ java Lab1Uppg03 [master ▲]
Program som konverterar mil till km.
Skriv in antal mil: 4.7
Motsvarande antal kilometer: 4700

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:47:19]
> $ java Lab1Uppg03 [master ▲]
Program som konverterar mil till km.
Skriv in antal mil: 2.19
Motsvarande antal kilometer: 2190

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:47:26]
> $ java Lab1Uppg03 [master ▲]
Program som konverterar mil till km.
Skriv in antal mil: 1.337
Motsvarande antal kilometer: 1337

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:47:31]
> $ [master ▲]

```

Figur 3: Körning av koden till Uppgift 1

## 4 Uppgift 4

### 4.1 Instruktioner

4. Skriv ett program som frågar efter åldern. Om den inmatade åldern är mindre än 0 så ska "Du har matat in fel ålder!" skrivas ut, annars ska "Hej, din XX åring!" skrivas ut. Istället för XX ska den ålder stå som skrevs in. Här är ett exempel på hur utskriften kan se ut, det som skrivs in från tangentbordet är markerat med fetstil/understrykning:

```
Hur gammal är du? 28
Hej, din 28 åring!
```

### 4.2 Lösning

#### 4.2.1 Funktion

För att lösa det här problemet används nästlade loopar. En yttre `do`-loop och en inre `while`-loop. Metoderna för ett objekt `scan` från klassen `Scanner` används för att undersöka den inmatade texten. Den inre `while`-loopen exekveras så länge `!scan.hasNextInt()` är sant, dvs då `scan`-objektet inte har en integer som nästa tecken i kön för att undersökas.

`hasNextInt()` hoppar över tecken som används som avskiljare och försöker sedan returnera nästa token i raden. Ör varje iteration av `while`-loopen frågas användaren om sin ålder och `scan` hoppar över den ogiltiga inmatningen med metoden `next()`.

**String next()** Finds and returns the next complete token from this scanner.

**boolean hasNextInt()** Returns true if the next token in this scanner's input can be interpreted as an int value in the default radix using the `nextInt()` method.<sup>3</sup>

Då den inre `while`-loopen avslutas sätts variabeln `age` till nästa integer som står på tur i kön med `scan.nextInt()`.

Den yttre `do`-loopen exekveras så länge den inmatade åldern `age` är mindre eller lika med noll. När det villkoret inte längre gäller antas `age` hålla en giltig ålder som skrivs ut med hjälp av `System.out.println`.

#### 4.2.2 Kommentar

Det här programmet har någorlunda skydd mot felaktig inmatning och är således relativt svårkraschat. Figur 4 visar en serie körningar med diverse märklig inmatning. Så länge inga escape-sekvenser följer med så klarar det sig från att krascha.

En lösning lik denna borde också kompletteras med undantagshantering för att gardera mot de fall då programmet faktiskt stöter på problem och kraschar till följd av oväntad inmatning.

#### 4.2.3 Källkod

```
1  /*
2  *  DV017A :: Grundläggande programmering i Java
3  *  =====
4  *  Uppdaterad 2015-06-09
5  *  Jonas Sjöberg 860224-7614
```

---

<sup>3</sup><http://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html>

```

6  * Högskolan i Gävle.
7  * <tel12jsg@student.hig.se>
8  *
9  * Labb #1
10 * Uppgift 4
11 */
12
13 import java.util.Scanner;
14
15 public class Lab1Uppg04 {
16
17     public static void main(String[] args) {
18         /* Skapa Scanner-objekt för att läsa I/O. Ignorera IDE:ns varningar. */
19         @SuppressWarnings("resource")
20         Scanner scan = new Scanner(System.in);
21
22         int age = 0;
23
24         do {
25             /* Loopa tills 'age' är nollskiljd och positiv. */
26             System.out.print("Hur gammal är du? ");
27
28             while (!scan.hasNextInt()) {
29                 /* Loopa tills nästa "token" i scan's lista går att parsea till en
30                  * int. Avgränsare mellan "tokens" är whitespace som standard,
31                  * där whitespace är mellanslag och andra tecken som motsvarar
32                  * tomt vertikalt eller horisontellt utrymme i text.
33                  */
34                 System.out.println("Hur gammal är du?");
35                 scan.next();
36             }
37
38             /* Scan måste hålla en int i sin "lista" som är OK att använda. */
39             age = scan.nextInt();
40
41         } while (age <= 0);
42
43         System.out.println("Hej, din " + age + "-åring!");
44     }
45 }

```

Lab1Uppg04.java

#### 4.2.4 Skärmdump

## 5 Uppgift 5

### 5.1 Instruktioner

5. Skriv ett program där användaren ska skriva in ett heltal. Programmet ska

```
Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/ - □ ×

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:48:24]
> $ java Lab1Uppg04 [master ▲]
Hur gammal är du? rbrrrrrppp
Hur gammal är du?
-10314
Hur gammal är du? 0000
Hur gammal är du? oj345nmadf.,
Hur gammal är du?
-.,jkhaduio
Hur gammal är du?
29
Hej, din 29-åring!

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:48:50]
> $ java Lab1Uppg04 [master ▲]
Hur gammal är du? 29
Hej, din 29-åring!

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:48:57]
> $ [master ▲]
```

Figur 4: Körning av koden till Uppgift 4

sedan skriva ut alla heltal från detta tal ner till ett. Du ska använda dig av en `while`-loop. Exempel på utskrift, användarens inmatning är markerat med fetstil/understrykning:

```
Ange det heltal som du vill räkna ner från: *6*
6 5 4 3 2 1
```

## 5.2 Lösning

### 5.2.1 Funktion

En `static final` variabel `QUERY` håller en textsträng som ska skrivas ut till användaren senare under programmets exekvering. Variabler med enbart versaler är en slags globala variabler som kan modifieras av programmeraren, likt det en gör med `#define` i C-liknande språk för att undvika att "hårdkoda" t.ex. textsträngar.

Logiken för att kontrollera korrekt inmatning är samma som den i Uppgift 4.

När variabeln `start` innehåller ett positivt heltal används en `while`-loop för att skriva ut en nedräkning på skärmen. Extra mellanslag läggs till genom att textsträngen `start` och ett mellanslag (" ") konkateneras genom summering, en funktion som särskiljer Java.

### 5.2.2 Källkod

#### Lab1Uppg05.java

```
1  /*
2  *   DV017A :: Grundläggande programmering i Java
3  *   =====
4  *   Uppdaterad 2015-06-11
5  *   Jonas Sjöberg 860224-7614
```

```

6  * Högskolan i Gävle.
7  * <tel12jsg@student.hig.se>
8  *
9  * Labb #1
10 * Uppgift 5
11 */
12
13 import java.util.Scanner;
14
15 public class Lab1Uppg05 {
16
17     private static final String QUERY = "Ange det heltal som du vill räkna ner från: ";
18
19     public static void main(String[] args) {
20         /* Skapa Scanner-objekt för att läsa I/O. Ignorera IDE:ns varningar. */
21         @SuppressWarnings("resource")
22         Scanner scan = new Scanner(System.in);
23
24         /* Att 'start' får värdet -1 gör ingen faktiskt skillnad eftersom att
25          * 'scan' garanterar att den inre while-loopen inte avslutas förrän
26          * 'start' faktiskt håller en integer och do-loopens conditional kan
27          * utvärderas säkert.
28          */
29         int start = -1;
30
31         do {
32             /* Loopa tills inmatning är ett nollskiljt och positivt heltal. */
33             System.out.print(QUERY);
34
35             while (!scan.hasNextInt()) {
36                 /* Loopa tills nästa "token" i scan's lista går att parsea till en
37                  * int. Avgränsare mellan "tokens" är whitespace som standard,
38                  * där whitespace är mellanslag och andra tecken som motsvarar
39                  * tomt vertikalt eller horisontellt utrymme i text.
40                  */
41                 System.out.print(QUERY);
42                 scan.next();
43             }
44
45             /* Scan måste hålla en int i sin "lista" som är OK att använda. */
46             start = scan.nextInt();
47
48         } while (start <= 0);
49
50         /* Räkna ner från 'start' till 1 med en **while-loop**. */
51         while (start > 0) {
52             System.out.print(start + " ");
53             start--;
54         }

```

```

55
56
57     /* Returnera lyckad exekvering oavsett hur det faktiskt gick. */
58     System.exit(0);
59 }
60 }

```

Lab1Uppg05.java

### 5.2.3 Skärmdump

```

Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/
spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:49:16]
> $ java Lab1Uppg05 [master ^]
Ange det heltal som du vill räkna ner från: -1
Ange det heltal som du vill räkna ner från: -100
Ange det heltal som du vill räkna ner från: 2.1
Ange det heltal som du vill räkna ner från: kjdfg
Ange det heltal som du vill räkna ner från: jk234jkj
Ange det heltal som du vill räkna ner från: 10
10 9 8 7 6 5 4 3 2 1 %

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:49:39]
> $ java Lab1Uppg05 [master ^]
Ange det heltal som du vill räkna ner från: 100
100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68
67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 3
4 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 %

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:49:44]
> $ [master ^]

```

Figur 5: Körning av koden till Uppgift 5

## 6 Uppgift 6

### 6.1 Instruktioner

6. Skriv två program som ger samma utskrift som den i uppgift 5, men använd istället en for-loop i ena programmet och en do-loop i det andra.

### 6.2 Lösning

#### 6.2.1 Funktion

Programmet använder återigen samma logik för att kontrollera korrekt inmatning men till skillnad från i Uppgift 4 och Uppgift 5 används en funktion `getUserInput()`.

Koden blir modulär och enklare att hantera. Funktionens åtkomstmodifierare `static` gör funktionen till en klassfunktion, alla objekt som instanstieras från klassen delar funktionen.

Funktionen hör till klassen, inte objekten. `protected` gör funktionen ”synlig” inom paketet.<sup>4</sup> Själva nedräkningen görs också inuti funktioner som anropas från `main`.

```
protected static void countdownUsingForLoop(int start)
    räknar ner från parametervärdet start med hjälp av en for-loop
```

```
protected static void countdownUsingDoLoop(int start)
    räknar ner från parameter värdet start med hjälp av en do-loop
```

### 6.2.2 Kommentar

I slutet av `main`-metoden avslutas programmet med `System.exit(0)`; där 0 indikerar lyckad exekvering och allt annat än noll nästan uteslutande är någon form av felkod, t.ex. 127 för ”command not found”<sup>5</sup>

Vid körning i Linux 3.19.0-21-generic med terminalemulatorn `zsh` version 5.0.7 skrivs märkliga tecken ut vid körning. Misstänker att detta är till följd av användningen av ’`n`’ newline-symboler som inte är plattformsspecifika. Det är rekommenderat att använda ’`%n`’ som genererar en plattformsspecifik newline-symbol.<sup>6</sup>

### 6.2.3 Källkod

```
1  /*
2  *  DV017A :: Grundläggande programmering i Java
3  *  =====
4  *  Uppdaterad 2015-06-11
5  *  Jonas Sjöberg 860224-7614
6  *  Högskolan i Gävle.
7  *  <tel12jsg@student.hig.se>
8  *
9  *  Labb #1
10 *  Uppgift 6
11 */
12
13 import java.util.Scanner;
14
15 public class Lab1Uppg06 {
16
17     private static final String QUERY = "Ange det heltal som du vill räkna ner från: ";
18
19     public static void main(String[] args) {
20         int start = getUserInput();
21
22         /* Räkna ner från 'start' till 1 med en **DO-loop**. */
23         System.out.println("\n\nRäknar ner med en \"do\"-loop ..");
24         countdownUsingDoLoop(start);
25
26         /* Räkna ner från 'start' till 1 med en **FOR-loop**. */
```

<sup>4</sup><https://docs.oracle.com/javase/tutorial/java/java00/classvars.html>

<sup>5</sup><http://www.tldp.org/LDP/abs/html/exitcodes.html>

<sup>6</sup><http://docs.oracle.com/javase/1.5.0/docs/api/java/util/Formatter.html#syntax>



```

27     System.out.println("\n\nRäknar ner med en \"for\"-loop ..");
28     countDownUsingForLoop(start);
29
30     /* Returnera lyckad exekvering oavsett hur det faktiskt gick. */
31     System.exit(0);
32 }
33
34 /**
35  * Räkna ner från 'start' till 1 med en **for-loop**.
36  * @param start startvärde att börja räkna ner ifrån
37  */
38 protected static void countDownUsingForLoop(int start)
39 {
40     for (int i = start; i > 0; i--) {
41         System.out.print(i + " ");
42     }
43 }
44
45 /**
46  * Räkna ner från 'start' till 1 med en **do-loop**.
47  * @param start startvärde att börja räkna ner ifrån
48  */
49 protected static void countDownUsingDoLoop(int start)
50 {
51     int i = start;
52     do {
53         System.out.print(i + " ");
54         i--;
55     } while (i > 0);
56 }
57
58 /**
59  * getUserInput
60  * Hämtar ett positivt heltal från användaren.
61  *
62  * @return ett positivt heltal från användaren.
63  */
64 protected static int getUserInput() {
65     /* Skapa Scanner-objekt för att läsa I/O. Ignorera IDE:ns varningar. */
66     @SuppressWarnings("resource")
67     Scanner scan = new Scanner(System.in);
68
69     /* Rutinen för att filtrera inmatningen är likadan som i Uppgift #5. */
70     int start = -1;
71
72     do {
73         System.out.print(QUERY);
74
75         while (!scan.hasNextInt()) {

```

```

76         System.out.print(QUERY);
77         scan.next();
78     }
79
80     start = scan.nextInt();
81
82     } while (start <= 0);
83
84     /* Returnera inmatat positivt heltal. */
85     return start;
86 }
87 }

```

Lab1Uppg06.java

## 6.2.4 Skärmdump

```

Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/
spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:50:38]
> $ java Lab1Uppg06 [master ^]
Ange det heltal som du vill räkna ner från: -1
Ange det heltal som du vill räkna ner från: kjhsdfg
Ange det heltal som du vill räkna ner från: -10845
Ange det heltal som du vill räkna ner från: 12.3
Ange det heltal som du vill räkna ner från: 100

Räknar ner med en "do"-loop ..
100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68
67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 3
4 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

Räknar ner med en "for"-loop ..
100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68
67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 3
4 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:50:49]
> $ [master ^]

```

Figur 6: Körning av koden till Uppgift 6

## 7 Uppgift 7

### 7.1 Instruktioner

7. Skriv ett program som låter användaren mata in värden till de tre heltalsvariablerna var1, var2 och var3. Programmet skall sedan för varje påstående a) - e) lagra värdet av påståendet i den booleska variabeln svar och därefter skriva ut värdet av svar , försett med lämplig ledtext:

- a) Talet var1 är jämnt delbart med 7.
- b) Talet var3 är inte jämnt delbart med talet var2.
- c) Talet var1 är större än minst något av talen var2 och var3.
- d) Talet var1 är större än talet var2, som i sin tur är större än talet var3.
- e) Talet var1 är större än ett av talen var2 och var3, men inte större än båda.

Tips: För att kolla om något tal är jämnt delbart med ett annat, använd modulus-operatorn %!

## 7.2 Lösning

### 7.2.1 Funktion

### 7.2.2 Kommentar

### 7.2.3 Källkod

```

1  /*
2   * DV017A :: Grundläggande programmering i Java
3   * =====
4   * Uppdaterad 2015-06-15
5   * Jonas Sjöberg 860224-7614
6   * Högskolan i Gävle.
7   * <tel12jsg@student.hig.se>
8   *
9   * Labb #1
10  * Uppgift 7
11  */
12
13  import java.util.Scanner;
14
15  public class Lab1Uppg07 {
16
17
18      private static boolean svar;
19
20      public static void main(String[] args) {
21
22          System.out.println(
23              "Var vänlig mata in heltal för variabler var1, var2 och var3;");
24          int var1 = getUserInput("var1: ");
25          int var2 = getUserInput("var2: ");
26          int var3 = getUserInput("var3: ");
27
28
29          // a) Talet var1 är jämnt delbart med 7.
30          svar = var1 % 7 == 0;
31
32          System.out.println("a) Talet " + var1 + " är jämnt delbart med 7? " + svar);
33
34

```

```

35 // b) Talet var3 är inte jämnt delbart med talet var2.
36 /* Modulus-operatören ger rest vid division, om divisionen går jämnt ut är
37    * resten noll. */
38 svar = !(var3 % var2 == 0);
39
40 System.out.println("b) Talet " + var3 + " är inte jämnt delbart med talet " + var2 + "?");
41
42
43 // c) Talet var1 är större än minst något av talen var2 och var3.
44 /* Använd OR-operatören '||'. */
45 svar = (var1 > var2) || (var1 > var3);
46
47 System.out.println("c) Talet " + var1 + " är större än minst något av talen " + var2 + " och " + var3);
48
49
50 // d) Talet var1 är större än talet var2, som i sin tur är större än talet var3.
51 /* Använder AND-operatören '&&'. */
52 svar = (var1 > var2) && (var2 > var3);
53
54 System.out.println("d) Talet " + var1 + " är större än talet " + var2 + ", som i sin tur är större än talet " + var3);
55
56
57 // e) Talet var1 är större än ett av talen var2 och var3, men inte större än båda.
58 /* Använder XOR-operatören '^'. Exklusiv-OR ger sant om A eller B är sant
59    * men inte om båda är sanna. */
60 svar = (var1 > var2) ^ (var1 > var3);
61
62 System.out.println("e) Talet " + var1 + " är större än ett av talen " + var2 + " och " + var3);
63
64
65
66 }
67
68
69 /**
70  * getUserInput
71  * Hämtar ett positivt heltal från användaren.
72  * Textsträngen 'query' skrivs ut tills dess att användaren matat in ett
73  * positivt heltal.
74  *
75  * @return ett positivt heltal från användaren.
76  */
77 protected static int getUserInput(String query) {
78     /* Skapa Scanner-objekt för att läsa I/O. Ignorera IDE:ns varningar. */
79     @SuppressWarnings("resource")
80     Scanner scan = new Scanner(System.in);
81
82     /* Rutinen för att filtrera inmatningen är likadan som i Uppgift #5. */
83     int input = -1;

```

```

84
85     do {
86         System.out.print(query);
87
88         while (!scan.hasNextInt()) {
89             System.out.print(query);
90             scan.next();
91         }
92
93         input = scan.nextInt();
94
95     } while (input <= 0);
96
97     /* Returnera inmatat positivt heltal. */
98     return input;
99 }
100
101 }

```

Lab1Uppg07.java

#### 7.2.4 Skärmdump

```

Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/
spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:51:02]
> $ java Lab1Uppg07 [master ^]
Var vänlig mata in heltal för variabler var1, var2 och var3;
var1: -1
var1: -2
var1: 3-
var1: 4-
var1: kjfg
var1: 3.1
var1: 12
var2: 34
var3: 9
a) Talet 12 är jämnt delbart med 7? false
b) Talet 9 är inte jämnt delbart med talet 34? true
c) Talet 12 är större än minst något av talen 34 och 9? true
d) Talet 12 är större än talet 34, som i sin tur är större än talet 9? false
e) Talet 12 är större än ett av talen 34 och 9, men inte större än båda? true
spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:51:39]
> $ [master ^]

```

Figur 7: Körning av koden till Uppgift 7

## 8 Uppgift 8

### 8.1 Instruktioner

8. Skriv ett program som skriver ut en viss multiplikationstabell. Den som kör programmet ska ange önskad multiplikationstabell och hur långt programmet ska räkna. Exempel på utskrift om man väljer tabell 8 och upp till 4:

```
Vilken multiplikationstabell önskas? *8*
Hur långt ska jag räkna? *4*
```

```
8 * 1 = 8
8 * 2 = 16
8 * 3 = 24
8 * 4 = 32
```

### 8.2 Lösning

#### 8.2.1 Funktion

#### 8.2.2 Kommentar

#### 8.2.3 Källkod

```
1  /*
2  *  DV017A :: Grundläggande programmering i Java
3  *  =====
4  *  Uppdaterad 2015-06-15
5  *  Jonas Sjöberg 860224-7614
6  *  Högskolan i Gävle.
7  *  <tel12jsg@student.hig.se>
8  *
9  *  Labb #1
10 *  Uppgift 8
11 */
12
13 import java.util.Scanner;
14
15 public class Lab1Uppg08 {
16
17     public static void main(String[] args) {
18
19         int multTable;
20         int countToNumber;
21
22         /* Hämta information från användaren. */
23         multTable = getUserInput("Vilken multiplikationstabell önskas? ");
24         countToNumber = getUserInput("Hur långt ska jag räkna? ");
25
26         /* Utför beräkning och skriv ut resultat.
27          * For-loopen exekverar en gång per uträkning och utskriftsrad. */
```

```

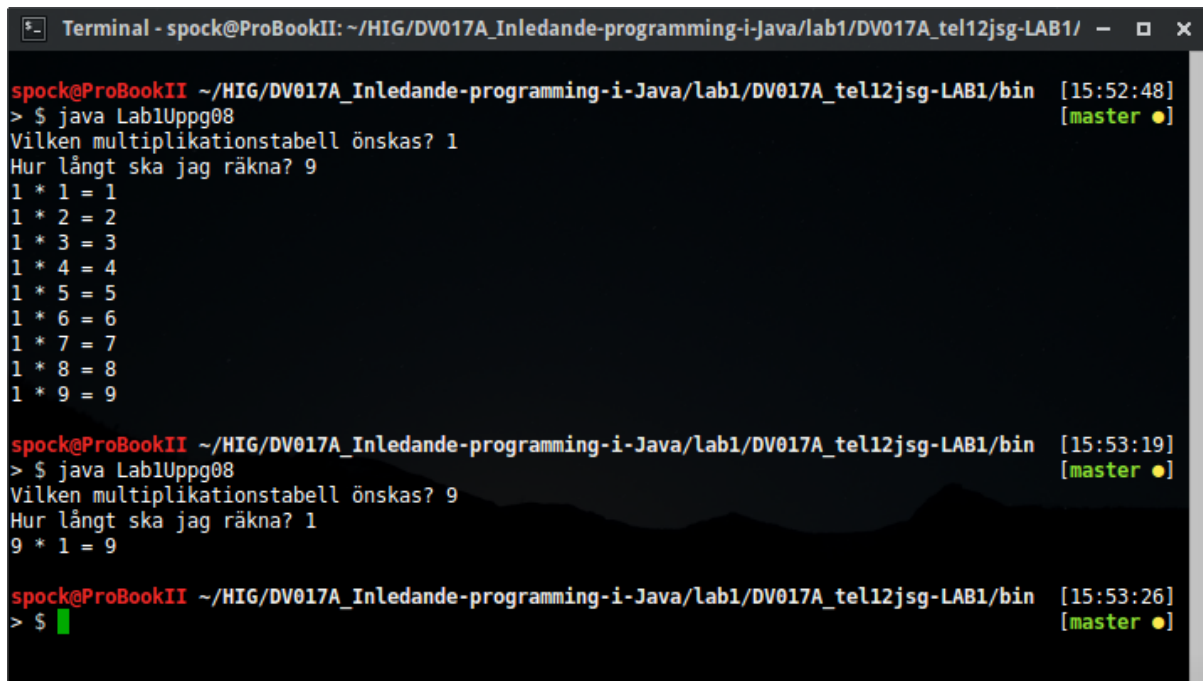
28         for (int i = 1; i <= countToNumber; i++) {
29
30             int result = multTable * i;
31             System.out.println(multTable + " * " + i + " = " + result);
32
33         }
34
35     }
36
37     /*
38     Vilken multiplikationstabell önskas? *8*
39     Hur långt ska jag räkna? *4*
40
41     8 * 1 = 8
42     8 * 2 = 16
43     8 * 3 = 24
44     8 * 4 = 32
45
46     */
47     /**
48      * getUserInput
49      * Hämtar ett positivt heltal från användaren.
50      * Textsträngen 'query' skrivs ut tills dess att användaren matat in ett
51      * positivt heltal.
52
53      * @return ett positivt heltal från användaren.
54      */
55     protected static int getUserInput(String query) {
56         @SuppressWarnings("resource")
57         Scanner scan = new Scanner(System.in);
58
59         int input = -1;
60
61         do {
62             System.out.print(query);
63
64             while (!scan.hasNextInt()) {
65                 System.out.print(query);
66                 scan.next();
67             }
68
69             input = scan.nextInt();
70
71         } while (input <= 0);
72
73         return input;
74     }
75
76

```

77  
78 }

Lab1Uppg08.java

## 8.2.4 Skärmdump



```
Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/ - □ ×

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:52:48]
> $ java Lab1Uppg08 [master ●]
Vilken multiplikationstabell önskas? 1
Hur långt ska jag räkna? 9
1 * 1 = 1
1 * 2 = 2
1 * 3 = 3
1 * 4 = 4
1 * 5 = 5
1 * 6 = 6
1 * 7 = 7
1 * 8 = 8
1 * 9 = 9

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:53:19]
> $ java Lab1Uppg08 [master ●]
Vilken multiplikationstabell önskas? 9
Hur långt ska jag räkna? 1
9 * 1 = 9

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:53:26]
> $ [master ●]
```

Figur 8: Körning av koden till Uppgift 8

## 9 Uppgift 9

### 9.1 Instruktioner

9. Skriv ett program som summerar alla jämna tal från och med 0 t.o.m 200 med hjälp av en while-sats och sedan skriver ut denna summa. Skriv sedan två program till som utför samma sak, men som istället använder sig av en do-resp en for-sats. Observera att du ska skriva de tre programmen utan att använda någon if-sats inne i iterations-satserna.

### 9.2 Lösning

#### 9.2.1 Funktion

### 9.3 Kommentar

#### 9.3.1 Källkod

```
1 /*
2  * DV017A :: Grundläggande programmering i Java
3  * =====
```



```

4  * Uppdaterad 2015-06-15
5  * Jonas Sjöberg 860224-7614
6  * Högskolan i Gävle.
7  * <tel12jsg@student.hig.se>
8  *
9  * Labb #1
10 * Uppgift 9
11 */
12
13 public class Lab1Uppg09 {
14
15     /* Tal att räkna upp till. */
16     private static final int UPPER_LIMIT = 200;
17
18
19     public static void main(String[] args) {
20
21         /* #1. Använder 'WHILE'-sats. */
22         int number = UPPER_LIMIT;
23         int sum = 0;
24
25         /* Använder 'number' som loop-räknare. Flyttar över 2 från 'number' till
26          * 'sum' per iteration. */
27         while (number > 0) {
28             sum += number;
29
30             number -= 2;
31         }
32
33         System.out.println("\n#1. Använder 'WHILE'-sats. ");
34         System.out.println("Summan av alla jämna tal från 0 till " + UPPER_LIMIT +
35                             " = " + sum);
36
37
38         /* #2. Använder 'DO'-sats. */
39         number = UPPER_LIMIT;
40         sum = 0;
41
42         /* Använder 'number' som loop-räknare. Flyttar över 2 från 'number' till
43          * 'sum' per iteration. */
44         do {
45             sum += number;
46             number -= 2;
47         } while (UPPER_LIMIT > 2 && number > 0);
48
49         System.out.println("\n#2. Använder 'DO'-sats. ");
50         System.out.println("Summan av alla jämna tal från 0 till " + UPPER_LIMIT +
51                             " = " + sum);
52

```

```

53
54     /* #3. Använder 'FOR'-sats. */
55     number = UPPER_LIMIT;
56     sum = 0;
57
58     /* Skippar initialisering inuti 'for'-loopen, tomt före första semikolon. */
59     for (; number > 0; number -= 2) {
60         sum += number;
61     }
62
63     System.out.println("\n#3. Använder 'FOR'-sats. ");
64     System.out.println("Summan av alla jämna tal från 0 till " + UPPER_LIMIT +
65                        " = " + sum);
66
67     /*
68     9. Skriv ett program som summerar alla jämna tal från och med 0 t.o.m 200 med
69     hjälp av en while-sats och sedan skriver ut denna summa. Skriv sedan två
70     program till som utför samma sak, men som istället använder sig av en do-
71     resp en for-sats. Observera att du ska skriva de tre programmen utan att
72     använda någon if-sats inne i iterations-satserna.
73
74     */
75     }
76
77 }

```

Lab1Uppg09.java

### 9.3.2 Skärmdump

## 10 Uppgift 10

### 10.1 Instruktioner

10. Skriv ett program där användaren ska ange ordningsnumret på en månad (1-12). Programmet ska sedan skriva ut månadens namn och antal dagar. Om man anger felaktigt månadsnummer (mindre än 0 eller större än 12) så ska ett felmeddelande skrivas på skärmen, och man ska sedan kunna ange ett nytt nummer. Använd switch-sats i din lösning

Programmet ska fortsätta att fråga efter månadsnummer ända tills man har matat in 0 (noll). Då ska programmet avbrytas.

```
Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/ - □ ×

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:53:48]
> $ java Lab1Uppg09 [master ●]

#1. Använder 'WHILE'-sats.
Summan av alla jämna tal från 0 till 200 = 10100

#2. Använder 'DO'-sats.
Summan av alla jämna tal från 0 till 200 = 10100

#3. Använder 'FOR'-sats.
Summan av alla jämna tal från 0 till 200 = 10100

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:53:50]
> $ [master ●]
```

Figur 9: Körning av koden till Uppgift 9

## 10.2 Lösning

### 10.2.1 Funktion

### 10.2.2 Kommentar

### 10.2.3 Källkod

```
1  /*
2   * DV017A :: Grundläggande programmering i Java
3   * =====
4   * Uppdaterad 2015-06-15
5   * Jonas Sjöberg 860224-7614
6   * Högskolan i Gävle.
7   * <tel12jsg@student.hig.se>
8   *
9   * Labb #1
10  * Uppgift 10
11  */
12
13  import java.util.Scanner;
14
15  public class Lab1Uppg10 {
16
17      /* Array håller namn för månader, textsträng i array index [0] har ett
18       * "dummy"-/placeholder-värde 'NULL' så att månad #1 hamnar på array-index #1.*/
19      private static final String[] MONTH_NAMES =
20          { "NULL", "Januari", "Februari", "Mars", "April", "Maj", "Juni", "Juli",
21            "Augusti", "September", "Oktober", "November", "December" };
22  }
```

22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70

```
/* Array av integers för antal dagar för månad. Arrayindex [0] håller ett
 * "dummy"-placeholder-värde '0' så att respektive månads ordningsnummer
 * sammanfaller med array-index.*/
private static final int DAYS_IN_MONTH[] =
    { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

/* Textsträng som skrivs ut för att prompta användaren om inmatning. */
private static final String QUERY = "\nÅnge ordningsnumret på en månad (1-12)"
    + "\n.. eller noll (0) för att avsluta: ";

public static void main(String[] args) {

    @SuppressWarnings("resource")
        Scanner scan = new Scanner(System.in);
    int input;

    /* "Huvud-loop" start. Loopa så länge användaren inte matar in '0'. */
    do {

        input = -1;

        do {

            /* Fråga användaren efter inmatning. */
            System.out.print(QUERY);

            /* Fortsätt fråga efter inmatning tills dess att scan "håller"
             * en int redo för inläsning. Använd next() för att kassera
             * gå vidare till nästa tecken, avskilt med whitespace. */
            while (!scan.hasNextInt()) {
                System.out.print(QUERY);
                scan.next();
            }

            /* Scan har hittat en int. */
            input = scan.nextInt();

            /* Skriv ut varning om inmatning är ogiltlig. */
            if (!(input >= 0) || !(input <= 12)) {
                //if (!(input >= 0 && input <= 12) {
                System.out.println("Felaktig inmatning!");
            }

            /* Fortsätt fråga om inmatning är ogiltlig. */
        } while (input < 0 || input > 12);

    } while (input != 0);

    /* Om användaren matat in '0', avsluta programmet genom att hoppa ut
```

```

71         * "huvud"-DO-loopen med 'break'. */
72         if (input == 0) {
73             break;
74         }
75
76         /* Skriv ut månadens namn och antal dagar. */
77         System.out.println(MONTH_NAMES[input] + " har "
78                             + DAYS_IN_MONTH[input] + " dagar.\n");
79
80     } while (input != 0);
81     /* "Huvud-loop" slutar här. */
82
83
84     System.out.println("\nAvslutar ..");
85     /* Returnera lyckad exekvering oavsett hur det faktiskt gick. */
86     System.exit(0);
87 }
88 }

```

Lab1Uppg10.java

## 10.2.4 Skärmdump

```

Terminal - spock@ProBookII: ~/HIG/DV017A_Inledande-programming-i-java/lab1/DV017A_tel12jsg-LAB1/ - □ ×
spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:54:33]
> $ java Lab1Uppg10 [master ●]

Ange ordningsnumret på en månad (1-12)
.. eller noll (0) för att avsluta: 2
Februari har 28 dagar.

Ange ordningsnumret på en månad (1-12)
.. eller noll (0) för att avsluta: 6
Juni har 30 dagar.

Ange ordningsnumret på en månad (1-12)
.. eller noll (0) för att avsluta: 12
December har 31 dagar.

Ange ordningsnumret på en månad (1-12)
.. eller noll (0) för att avsluta: 0

Avslutar ..

spock@ProBookII ~/HIG/DV017A_Inledande-programming-i-Java/lab1/DV017A_tel12jsg-LAB1/bin [15:54:45]
> $ [master ●]

```

Figur 10: Körning av koden till Uppgift 10

## 11 Referenser

### 11.1 www

### 11.2 Literature

### 11.3 Source files

Full source, including spice simulation files, CSV data, schematics, etc TODO: is available at [https://github.com/jonasjberg/FIX\\_URL](https://github.com/jonasjberg/FIX_URL)