HÖGSKOLAN
I HALMSTAD

Verónica Gaspes
ITE
26 oktober 2023

Inlämningsuppgifter för vecka 1

Anmärkningar

- Hur man lämnar in och vad som ska lämnas in förklaras i Blackboard och i *README* filen som följer med i zip-paketet. Denna dokument beskriver uppgifterna. Varje uppgift finns i en .java fil. I filen finns en kommentar som förklarar uppgiften på engelska.
- När du löser veckans uppgifter behöver du bara använda det vi har presenterat under första veckan:
  - Bara 2 kommandon: tilldelning och `System.out.println`. Det behövs inga andra kommandon för att lösa uppgifterna.
  - Många operationer och några metoder för att bygga upp uttryck i olika typer.
- Inför tentan måste du kunna
  - att det finns både kommandon och uttryck
  - att det finns olika typer
  - förklara vad de kommandon vi har studerat gör
  - förklara vilka typer och värden några uttryck har

  Efter inlämningsuppgifterna finns några uppgifter från gamla tentor som handlar om detta och som kan användas för att öva inför tentan och ställa frågor på Drop-in passen. Dessa lämnas inte in.

Inlämningsuppgifter - del 1

1. I filen *E1.java* finns följande kod, inklusive en kommentar som förklarar vad som behöver göras på engelska:

```
/*
This program is compiled using
   javac E1.java

and is run using
   java E1

Add code so that the program  prints one more line with the message
   Welcome to the programming course!
after the
   Hello, world!
message.

*/
public class E1{

  public static void main(String[] args) {
    System.out.println("Hello, world!");
    // Your code here!
  }

}
```

   Det du behöver göra är:

   (a) Spara filen *E1.java* i mappen där du ska arbeta.

   (b) Kompilera programmet med kommandot

   ```
   javac E1.java
   ```

   (c) Köra programmet med kommandot

   ```
   java E1
   ```

   Du borde se utskriften

   ```
   Hello, world!
   ```

   (d) Ändra i filen genom att lägga till en rad kod för att även skriva ut
   `Welcome to the programming course!`
   Det står
   `// Your code here!`
   där du ska ha din rad kod.

   (e) Spara filen innan du kompilerar och kör på samma sätt som innan. Utskriften bör nu vara:

   ```
   Hello, world!
   Welcome to the programming course!
   ```

2. I filen *E2.java* finns ett Java program. Det du behöver göra är:

   (a) Spara filen *E2.java* i mappen där du ska arbeta.

   (b) Kompilera programmet med kommandot

   ```
   javac E2.java
   ```

   (c) Köra programmet med kommandot

   ```
   java E2 vero
   ```

   Här är *vero* ett så kallat kommandoradsargument och fungerar som indata till programmet.
   Du borde se utskriften

   ```
   Welcome to the course vero!
   ```

   Du kan köra programmet med andra namn än *vero*. Gör det och se vad som händer. Du kan även testa att köra utan något namn för att se vad som händer.

   (d) Ändra i filen så att i utskriften ingår även
   `and hope you enjoy coding`
   efter namnet och före utropstecknet. Behåll bara ett kommando `System.out.println` och ändra i uttrycket som beräknar den test som skrivs ut. Det ska finnas ett mellanslag mellan namnet och ordet `and`.

   (e) Spara filen innan du kompilerar och kör på samma sätt. Utskriften bör nu vara:

   ```
   Welcome to the course vero and hope you enjoy coding!
   ```

3. I filen *E3.java* finns ett Java program. Det du behöver göra är:

   (a) Spara filen *E3.java* i mappen där du ska arbeta.

   (b) Kompilera programmet med kommandot

   ```
   javac E3.java
   ```

   (c) Köra programmet med kommandot

   ```
   java E3 10 20
   ```

   Talen 10 och 20 är kommandoradsargumenten och blir indata till programmet. Du borde se utskriften

   ```
   The value of a is 10
   and the value of b is 20
   The value of a is now 10
   and the value of b is now 20
   ```

   Du behöver veta att kommandoradsargumenten tas in i programmet som text (`String`). När indata ska vara ett heltal i programmet behöver texten konverteras till tal. Detta görs i programmet med `Integer.parseInt` (det är redan gjort som du kan se).
   Du kan köra programmet med andra tal än `10 20`. Gör det och se vad som händer. Du kan även testa att köra med bara ett tal eller inga tal alls för att se vad som händer.

   (d) Ändra i filen där det står `// Your code here` så att variablerna byter innehåll med varandra.

   (e) Spara filen innan du kompilerar och kör på samma sätt. Utskriften bör nu vara:

   ```
   The value of a is 10
   and the value of b is 20
   The value of a is now 20
   and the value of b is now 10
   ```

4. I filen *E4.java* finns ett Java program. Det du behöver göra är att spara och kompilera som i tidigare uppgifter för att sedan

   (a) Köra programmet med

   ```
   java E4 8 78
   ```

   Första talet står för vad klockan är och andra talet för hur många timmar som ska läggas till. Båda ska vara positiva heltal. Med den kod som finns i filen nu, borde se utskriften

   ```
   In 78 hours the time would be 0
   ```

   Du kan köra programmet med andra tal än `8 78`. Gör det och se vad som händer. Du kan även testa med indata som inte är tal för att se vad som händer.

   (b) Ändra i filen så att det värdet som tilldelas variabeln
   `tell_me_the_hour`
   beräknas med ett uttryck som ger rätt värde: vad klockan blir när man lägger till antalet timmar till vad klockan är. Uttrycket finns i föreläsningar om restaritmetik för matte kursen Algebra och diskret matematik.

   (c) Spara filen innan du kompilerar och kör på samma sätt som innan. Utskriften bör nu vara:

   ```
   In 78 hours the time would be 14
   ```

5. I filen *E5.java* finns ett Java program. Spara filen i rätt mapp och kompilera.

   Programmet är tänkt att beräkna addition och multiplikation i Z_n (restaritmetik modulo n).

   Indata till programmet ska vara tre positiva heltal: x, y och n. Programmet ska beräkna $x+y$ i restaritmetik modulo n och $x \cdot y$ i restaritmetik modulo n. Hur man gör detta har ni lärt er i matte kursen under läsperiod 1.

   Programmet går att köra men beräknar inte rätt svar. I programmet står det angivet var ni ska göra de ändringarna som behövs.

   Några exempel med indata och korrekt utskrift:

   ```
   java E5 3 4 5
   [3 + 4] in Z_5 is 2
   [3 * 4] in Z_5 is 2

   java E5 3 4 7
   [3 + 4] in Z_7 is 0
   [3 * 4] in Z_7 is 5
   ```

6. I filen *E6.java* finns ett Java program. Spara filen i rätt mapp och kompilera.

   Programmet använder ett kommandoradsargument som ska vara ett heltal. Programmet ska beräkna sanningsvärdet av i fall 3 är en delare till programmets indata.

   Programmet går att köra och skriver ut ett meddelande men beräknar inte rätt värde. I programmet står det angivet var ni ska göra de ändringarna som behövs.

   Här är några exempel med indata och utskrift:

```
java E6 4
[ 3 divides 4 ] is false

java E6 123
[ 3 divides 123 ] is true
```

7. I filen *E7.java* finns ett Java program. Spara filen i rätt mapp och kompilera.

   Programmet använder två kommandoradsargument som ska båda vara heltal. Programmet ska beräkna sanningsvärdet av i fall det första talet är en delare till det andra talet.

   Programmet går att köra och skriver ut ett meddelande men beräknar inte rätt värde. I programmet står det angivet var ni ska göra de ändringarna som behövs.

   Här är några exempel med indata och utskrift:

```
java E7 4 6
[ 4 divides 6 ] is false

java E7 3 123
[ 3 divides 123 ] is true
```

8. I filen *E8.java* finns ett Java program. Spara filen i rätt mapp och kompilera.

   Lägg till den kod som behövs för att

   - ta in tre kommandoradsargument som ska vara reella tal. Dessa tal står för en punkts koordinater x, y och z och
   - skriva ut avståndet från punkten till origo med koordinater $0, 0, 0$.

   Som alltid, indata tas in i programmet som text (`String`). För att göra om en text till ett flyttal används `Double.parseDouble` på samma sätt som vi tidigare använt `Integer.parseInt`.

   Här är några exempel med indata och utskrift:

```
java E8 2 3 4
The distance to 0,0,0 is 5.385164807134504

java E8 1 2 3
The distance to 0,0,0 is 3.7416573867739413
```

   Du kommer att behöva använda $\sqrt{}$. I Java är kvadratroten en funktion som heter `Math.sqrt` och som tar ett argument. Till exempel, om du vill beräkna $\sqrt{4}$ kan du skriva `Math.sqrt(4)`.

9. I filen *E9.java* finns ett Java program. Spara filen i rätt mapp och kompilera.

   Lägg till den kod som behövs för att skriva ut två heltal som ska stå för att ha kastat två tärningar. Alltså, varje gång man kör programmet ska man få se två heltal som kan vara utfallet av att ha kastat två tärningar.

   I föreläsningarna visade vi hur vi kan använda ett slumptal generator (`Math.random()`) för att producera slumptal. Tyvärr är resultatet ett flyttal i intervallet $[0, 1)$ och inte ett heltal mellan 1 och 6. Men, om

man multiplicerar med rätt värde och sedan typkonverterar till heltal kan man få ett tal mellan 1 och 6.

Här är några exempel:

```
java E9
1 3

java E9
4 1
```

10. I filen *E10.java* finns ett Java program. Spara filen i rätt mapp och kompilera.

    Lägg till kod för att skriva ut fem slumptal i intervallet $[0, 1)$ samt deras medelvärde, minsta värde och största värde. Du ska använda dig av `Math.random`, `Math.min` och `Math.max`.

    Här är ett exempel som visar hur utskriften ska se ut:

```
java E10
0.7051636132872013
0.45096438398902017
0.8292986674162087
0.7645659358856115
0.05720540849784983

average: 0.5614396018151784
min: 0.05720540849784983
max: 0.8292986674162087
```

11. I filen *E11.java* finns ett Java program. Spara filen i rätt mapp och kompilera.

    Lägg till kod för att

    - läsa in tre heltal från kommandoraden och
    - skriva ut dom i ordning, från minsta till största.

    Använd funktionerna `Math.min` och `Math.max`.

    Här är ett exempel som visar hur utskriften ska se ut:

```
java E11 1 9 5
1 5 9

java E11 5 9 1
1 5 9
```

Inlämningsuppgifter - del 2

I filen *Bonus.java* finns ett Java program. Spara i rätt mapp och kompilera.

Lägg till kod för att skriva ut de så kallade Fibonacci ord av rang 0 till 10. Dessa ord är enligt följande:
rang 0: a
rang 1: b
rang 2: ba
rang 3: bab
rang 4: babba
och i allmänt, för rang n: ta ordet med rang (n -1) och lägg ihop med ordet av rang (n-2).
Du ska använda flera kommandon för att bygga upp orden som `String` och sammanfoga orden för att skapa nya ord med gamla. Skriv ut de elva orden.

Uppgifter från gamla tentor som kan lösas efter vecka 1.

Den blåa texten är ett lösningsförslag.

- För följande tre uttrycken, ange värde och typ samt en kort förklaring.

Uttryck	Värde	Typ	Förklaring
`1 + "/ " + 2 + " = " + 1 / 2`	`"1 / 2 = 0"`	String	Se under tabellen
`Integer.parseInt("123") * Double.parseDouble("2")`	`246.0`	double	Se under tabellen
`(int)Math.random()`	`0`	int	Se under tabellen

  Förklaring gällande första raden: operationernas precedens och associativitet är sådana att uttrycket står för `(((1 + "/ ") + 2 ) + " = ") + (1 / 2)`. Operationen + mellan String och int konverterar heltalet till String och lägger ihop båda Strings. Operationen / mellan heltalen är kvoten.

  Förklaring gällande andra raden: aritmetiska operationer mellan int och double konverterar heltalet till double.

  Förklaring gällande tredje raden: `Math.random()` är ett double i intervallet [0, 1). Typkonvertering (int) behåller bara heltalsdelen (ignorerar alla decimaler).

- Vad blir utskriften av följande program:

```
public static void main(String[] cmdLn){
   System.out.println(5+3);
   System.out.println(5-3);
   System.out.println(5*3);
   System.out.println(5/3);
   System.out.println(5%3);
   System.out.println(3*5-2);
   System.out.println(3+5/2);
   System.out.println(3-5-2);
   System.out.println((3-5)-2);
   System.out.println(3-(5-2));
   System.out.println(5/3*2);
}
```

- Vad blir utskriften av följande program:

```
public static void main(String[] cmdLn){
   String ruler1 = "1";
   String ruler2 = ruler1 + " 2 " + ruler1;
   String ruler3 = ruler2 + " 3 " + ruler2;
   String ruler4 = ruler3 + " 4 " + ruler3;
   String ruler5 = ruler4 + " 5 " + ruler4;
   System.out.println(ruler1);
   System.out.println(ruler2);
   System.out.println(ruler3);
   System.out.println(ruler4);
   System.out.println(ruler5);
}
```