DA339A Laboration L8

Syfte

Den här laborationens syfte är att öva på att använda metoder som verktyg vid problemlösning och att skriva kod som innehåller metoder.

Redovisning

Laboration L8 ska inte redovisas i sig självt. Laborationen innehåller dock uppgifter som kan komma att användas och utökas i senare laborationer som i sin tur ska redovisas och i inlämningsuppgifter.

Förberedelser

Den här laborationen är direkt kopplad till det innehåll som finns i föreläsning F7 till F9.

För laboration L8 förutsätts du även behärska tidigare genomgånget innehåll på kursen.

Du behöver ladda ner filen *da339a_L8_filer.zip* och packa upp denna inför laborationen då den innehåller filer som du ska utgå ifrån för vissa av uppgifterna. I zip-filen finns en fil Template.java med ett “skelett” till ett program som du kan använda när du ska skapa egna filer till dina program. Lösningsförslag finns också på kurssidan, zip-filen *L8_Solution.zip*.

Om extra övningar

Vissa uppgifter innehåller en liten uppmaning om extra övningar. Vissa av dessa är en utmaning för de som vill ha något lite svårare att göra. Vissa uppgifter är mer en uppmaning att testa olika sätt att lösa en uppgift eller att göra utökningar av uppgiften. Finner du uppgifterna i sig själva tillräckligt utmanande så gör inte de extra övningarna med en gång. Spara då dessa som extra kod att experimentera med och övning till när du löst alla de övriga uppgifterna. Kom ihåg - programmera och skriva kod är ett hantverk och övning ger färdighet! Experimentera och lek runt och testa olika idéer i din kod när du gjort de regelrätta uppgifterna.

Uppgifter

Uppgift 1

Ta en kopia på filen *Template.java* och döp om den till ditt programnamn, ex. *MittProgram.Java*. Döp också om klassnamnet i filen.

Filen innehåller ett program med en while-loop i main(…). Din uppgift blir att skriva en program-meny med en switch-case inuti loopen. Från menyn skall du sedan anropa olika programfunktionalitet. Det gör du med en switch/case sats och metoder/metodanrop för menyvalen. Nedan visas en utskrift av *Template.java*. I själva kodfilen finns hjälpsamma kommentarer i koden. Tips! Tänk på att programmets flöde sker i *main(…).* Det finns två färdiga metoder i koden:

- ***public static void showMenu() {…},*** som är konstruerad för att skriva ut menyn på skärmen.
- ***public static int makeChoice() {…},*** som är konstruerad för att begära ett val från användaren.

```
import java.util.Scanner;
import java.security.SecureRandom;

public class Template {
  public static void main(String[] args) {

    boolean isRunning = false;
    do {
        // Här skrivs för närvarande en meny ut

    } while (isRunning);
  }

  public static void showMenu()
  {
    System.out.println("Select one:");
    System.out.println("1. Add numbers");
    System.out.println("2. Countdown");
    System.out.println("3. Compare numbers");
    System.out.println("4. Factorial");
    System.out.println("5. Randomize");
    System.out.println("6. Temperature");
    System.out.println("0. Exit");
  }

  public static int makeChoice()
  {
    Scanner input = new Scanner(System.in);
    System.out.print("Make a choice: ");
    int choice = input.nextInt();
    return choice;
  }
}
```

Här, inuti while-loopen, skall du kalla på metoder och skriva din switch-case kod.

Metoden showMenu()

Metoden makeChoice()

Nedan följer uppgifter för att skriva en programmeny med switch/case.

Uppgift 1a

Nu skall du kalla på metoder från while-loopen (menyloopen) i *main()*. Metoderna kommer att skriva ut på skärmen och vänta på input från användaren.

1. Ersätt menyutskriften genom att istället kalla på metoden *showMenu()* från while-loopen, menyloopen, i main() för att skriva ut en meny på skärmen. Du borde få en utskrift som liknar detta (samma som tidigare):

```
Select one:
1. Add numbers
2. Countdown
3. Compare numbers
4. Factorial
5. Randomize
6. Temperature
0. Exit
```

2. Kalla på metoden *makeChoice()* efter *showMenu()* och lagra ned returvärdet från metoden i en *int* variabel.
   Ex. *int val = makeChoice();*
   Du borde få en utskrift som liknar detta:

```
Select one:
1. Add numbers
2. Countdown
3. Compare numbers
4. Factorial
5. Randomize
6. Temperature
0. Exit
Make a choice:
```

3. Du kan nu mata in ett val, men du hoppar ut ur programmet direkt. Det beror på att variabeln *isRunning* är *false*. Så vill vi inte att vårt program skall fungera.
   Fundera på vilka olika sätt det kan lösas på innan du fortsätter till nästa uppgift. Går det bara att sätta *isRunning* till *true*?

Uppgift 1b

Nu skall du skriva första delen av switch-case satsen enligt nedan.

1. Efter tagit emot input från användaren, *makeChoice()*, skall du skriva din switch/case sats. ”Switcha” på valet du lagrade ned i en variabel med hjälp av *makeChoice()*
   ex.
   ```
   switch (val) {
   … // Dina case satser här.
   }
   ```
2. Skriv ***case*** för val ***0*** först, dvs för att avsluta programmet. *Tips! Använd isRunning variabeln för att avsluta menyloopen (eller för att stanna kvar).*
   Obs! *break* får inte användas för att avsluta programmet.
3. Nu borde det gå att avsluta programmet.
   Men vi tar inte hand om felaktiga val. Därför behöver du också skriva en *default case* med info till användaren när det sker. Gör detta!

Uppgift 1c

Nu är det dags att börja skriva egna metoder. Gör nya *case* för varje menyval och kalla på en metod efter ':' i case-satsen. Så här kan det se ut:

```
case 2: addNumbers();
  break;
```

Exempel bilden visar att vi kallar på en metod *addNumbers()* i vårt case 2, dvs val 2 i menyn. Nedan listas lite tankar på vad menyvalen skall/kan innehålla. Men försök att koda utan att titta så noga på varje steg utan använd den kunskap du redan har för att försöka lösa uppgiften. Om du till exempel skriver ut vart tredje tal i stället för vartannat i val 1 har inte så stor betydelse. Försök också fundera på vad som kan göras bättre när du kodar, dvs lek runt och testa i koden du skriver. Metoder går vi igenom på F9.

1. **Nedräkning**
    a. Här skall programmenyn kalla på en metod som räknar ned från heltalet 50 till heltalet 0.
    b. En for-loop skall användas.
    c. Metoden skall skriva ut vartannat tal, dvs ha decrementet '-2'
    d. Ta inspiration av de metoder som redan finns i koden och från föreläsningar om for-loopar.
    e. Signaturen skall vara:
    `public static void countdown() {…}`

2. **Addera nummer**
    a. Här skall programmenyn kalla på en metod som adderar heltal (int) i en summa med hjälp av en while-loop.
    b. Om användaren matar in '0' som värde skall loopen avslutas och summan skrivas ut på skärmen.
    c. Signaturen på metoden skall vara
    `public static void addNumbers() {…}`
    d. Du behöver en *Scanner* för input och några lokala variabler (*int*) i metoden för att hålla reda på inmatade värden och summan.
    e. Ta inspiration av de metoder som redan finns i koden och från föreläsningar för summering av tal.

3. **Jämför flera nummer**
    a. Här skall programmenyn kalla på en metod som jämför positiva flyttal och skriver ut det största på skärmen.
    b. Signaturen på metoden skall vara
    public static void compare(int num) {…}
    där num är en parameter för antal tal som skall matas in.
    c. num får ”hårdkodas” in när du kallar på metoden i *main()*. Ex. *compare(5)*;
    d. Skriv en for-loop som slumpar *num* tal och skriver ut dem på skärmen. Använd *SecureRandom*.
    e. for-loopen skall också kunna hålla reda på vilket tal som är störst genom att lagra detta i en lokal variabel *max*.
    Tips,
    ```
    if (tal > max)
    {
       max =  tal;
    }
    ```
    f. Variabeln *max* skall skrivas ut efter loopen är klar.
    g. https://docs.oracle.com/javase/8/docs/api/java/security/SecureRandom.html

4. **Fakultetuträkning**
    a. Här skall en metod kodas som räknar ut fakulteten på valfritt positivt heltal.
    b. I case satsen i main skall programmet fråga efter ett heltal som fakultet skall beräknas på och lagra detta i en variabel. Till exempel *int fac*. Tag input med en `Scanner` som vanligt.
    c. Signaturen på metoden skall vara
    public static int factorial(int fac) {…}
    d. Använd en for-loop i metoden. Fakultet: ex. 5! = 5 * 4 * 3 * 2 * 1.
    Tips! Utelämna noll i for-loopen. Fundera på varför?
    e. Metoden returnerar en int som resultat och skall skrivas ut på skärmen i *main()*. Exempelvis *return result*;, där result är en lokal variabel *int result*.
    f. Inga utskrifter får göras inne i metoden, med exempelvis **`System.out.println()`**.
    g. Om det skickas in 0 i metoden skall 1 returneras som resultat (0! = 1)
    h. Om det skickas in ett värde mindre än 0 skall metoden returnera -1 eller liknande för felaktigt resultat. Ta hand om det felaktiga resultat på ett adekvat sätt i *main()*.

5. **Randomisera + Oändlig loop**
    a. Här skall en metod kodas som randomiserar två tal i taget i en while-loop. Använd `SecureRandom`.
    b. Gör en valfri matematisk operation på talen med hjälp av Math klassen och kontrollera resultatet.
    c. Om resultat från operationen uppfyller något villkor, som du själv kommit på, så avslutas loopen. Tänk på att hitta ett villkor som inte skapar en oändlig loop.
    d. https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html
    e. https://docs.oracle.com/javase/8/docs/api/java/security/SecureRandom.html

6. **Temperaturkonvertering**
    a. Här skall en metod användas för att konvertera Fahrenheit till Celsius och tvärtom och presentera resultatet.
    b. Skriv en huvudmetod med signaturen
    public static void temperature() {…}.
    Den metoden kallas på från *main()*.
    c. I *temperature()* metoden delar du upp flödet i två nya metoder. Dessa skall kunna väljas i en liten ”mini-meny” med två val, använd if eller switch/case
        i. ***public static double fahrToCels(double t) {…}***
        ii. ***public static double celsToFahr(double t) {…}***
    d. Metoderna skall beräkna temperturkonvertering baserat på val och returnera värdet.
    e. I *temperature()* tar du hand om värdet från den valda metoden i d och presenterar resultatet.
    f. Använd tidigare lab L6 för kod om du känner för det, men anpassa för metodanrop.
        i. `C = (5/9) * (F – 32)`
        ii. `F = (9/5) * C + 32`

Extrauppgift

Hittar du förenklingar och förbättringar som går att göra i lösningsförslaget eller i egen kod? Fundera kritiskt.

- Går det till exempel att använda mer metoder?
- Går det att utnyttja returvärden mer?
- Går det att implementera fler parametrar i metodhuvuden?
- Mer?

Exempelutskrift i Git Bash vid körning av lösningsförslag

Meny: Select one

```
af6509@WC5CG1473GD7 MINGW64 ~/kurser/da339a/lab/lab8/da339a_L8_filer
$ java L8_PropSolution
Select one:
1. Countdown
2. Add numbers
3. Compare numbers
4. Factorial
5. Randomize
6. Temperature
0. Exit
Make a choice:
```

Choice 1: Countdown

```
Make a choice: 1
Value : 50
Value : 48
Value : 46
Value : 44
Value : 42
Value : 40
Value : 38
Value : 36
Value : 34
Value : 32
Value : 30
Value : 28
Value : 26
Value : 24
Value : 22
Value : 20
Value : 18
Value : 16
Value : 14
Value : 12
Value : 10
Value : 8
Value : 6
Value : 4
Value : 2
Value : 0
```

Choice 2: Add numbers

```
Select one:
1. Countdown
2. Add numbers
3. Compare numbers
4. Factorial
5. Randomize
6. Temperature
0. Exit
Make a choice: 2
Number 1: 4
Number 2: 4
Number 3: 0

The sum is 8
```

Choice 3: Compare numbers

```
Make a choice: 3
Number: 7.067184410958771
Number: 352.55642979985623
Number: 287.98500345596113
Number: 504.18901115055803
Number: 50.185406758222584
The largest number was: 504.18901115055803
```

Choice 4: Factorial

```
Make a choice: 4
Integer to calculate: 4

Result = 24
```

Choice 5: Randomize

```
Make a choice: 5
Randomizing:
8.100138389901202 powered to 8.255212185257367 is 3.1608060729123987E7
```

Choice 6: Temperature

```
Make a choice: 6

1. F to C
2. C to F

Make a choice: 2
Enter temperature: 25

Fahrenheit: 77.0

Select one:
1. Countdown
2. Add numbers
3. Compare numbers
4. Factorial
5. Randomize
6. Temperature
0. Exit
Make a choice: 6

1. F to C
2. C to F

Make a choice: 1
Enter temperature: 100

Celsius: 37.77777777777778
```

Choice 0: Exit

```
Make a choice: 0

af6509@WC5CG1473GD7 MINGW64 ~/kurser/da339a/lab/lab8/da339a_L8_filer
$
```