DA339A Laboration L2

Syfte

I laboration L2 kommer du lära dig mer om att navigera, filhantera samt kompilera och köra javaprogram i kommandotolken. I laborationen ingår också lära sig att rätta enklare fel i koden. För de allra flesta felen i L2 behöver man ingen kunskap om programspråket Java utan man klarar sig med att bara leta efter stavfel och titta på ordningen i programmet.

Redovisning

Laboration L2 ska inte redovisas.

Förberedelser

Du förväntas ha genomfört laboration L1.
VIKTIGT – Ladda ner zip-filen *testa_java.zip*, packa upp, och ersätt den gamla *testa_java* mappen (med samma namn) som ni skapade i laboration 1 med denna uppdaterade mapp. Ni kan hitta zip-filen på kurssidan vid laboration 2.

Program som används i laborationen

Utöver de program som användes i L1 kommer L2 att introducera:

- programmet Atom för att editera text
- Java Development Kit för att kunna kompilera och köra ett Javaprogram på datorn

Uppgifter

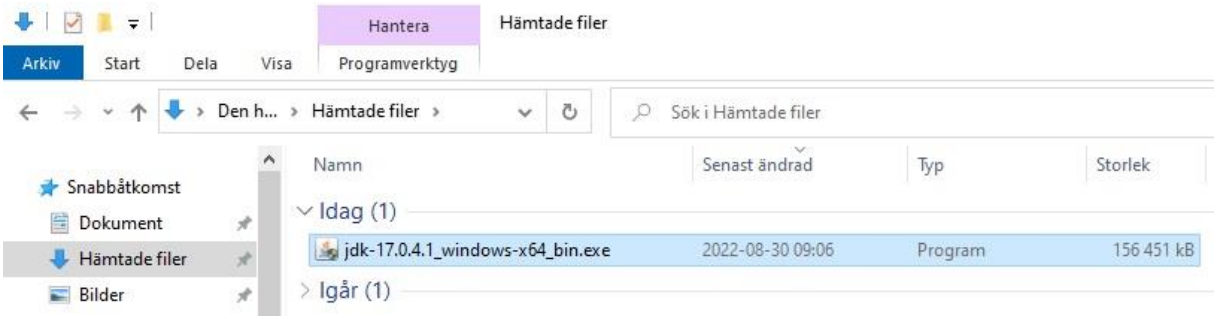
Uppgift 1 Kompilator

VIKTIGT – Om du jobbar på skolans datorer behöver du inte ladda ner OpenJDK 17 (finns redan installerat). Du kan skriva ”java -version” i Git Bash för att se vilken version av Java som finns och därefter börja med Uppgift 2.

För att vi skall kunna kompilera och köra ett Javaprogram måste Java Development Kit vara installerat på datorn. För att installera detta börja med att ladda ner OpenJDK 17 från Oracle. Var säker att du väljer ”Installer” varianten (enklare installation) och för rätt operativsystem.

https://www.oracle.com/java/technologies/javase/jdk17-archive-downloads.html

Gå till katalogen *Hämtade filer* (eller var du valde att spara filen du laddade ner om du sparade denna någon annanstans) och hitta installationsfilen (brukar vara högst upp för att det är den senaste filen du laddat ner). Dubbelklicka på filen och klicka vidare vid dialoger så installeras, sannolikt, allt enligt planerna.



Testa installationen

För att se om du installerat JDK rätt använder du ett annat program PowerShell (PC)/Terminalen (Mac)/Git Bash. Om du letar på nätet finns det ofta förklaringar som använder sig av kommandotoken cmd.exe. Det beror på att i äldre Windows-installationer finns inte alltid PowerShell med utan då får man använda kommandotoken cmd.exe som har lite andra kommandon. Men vi använder Git Bash då det har samma utseende för det vi skall göra som Terminalen på Mac eller PowerShell på PC.

För att starta Git Bash på PC skriv ”Git Bash” i sökfältet eller via *Launchpad* för Mac.

Nu kan du kan du skriva ”java -version” för att ta reda på vilken version av Java du har och om installationen gick bra. Har allt fungerat som det skall kommer det stå något i stil med:

```
af6509@WC5CG1473GD7 MINGW64 ~
$ java -version
java version "17.0.4" 2022-07-19 LTS
Java(TM) SE Runtime Environment (build 17.0.4+11-LTS-179)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.4+11-LTS-179, mixed mode, sharing)
```

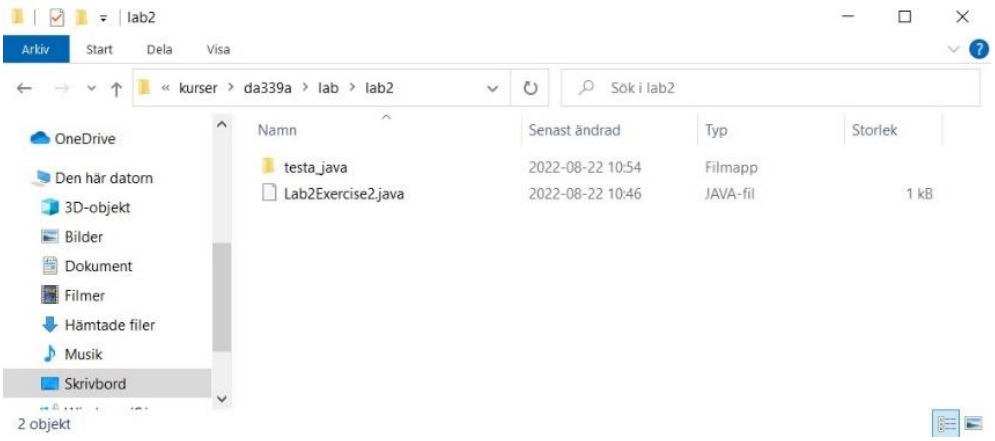
Om det fungerar kan du hoppa direkt till Uppgift 2. Fungerar det inte fråga labbhandledaren om hjälp.

Uppgift 2 Editor

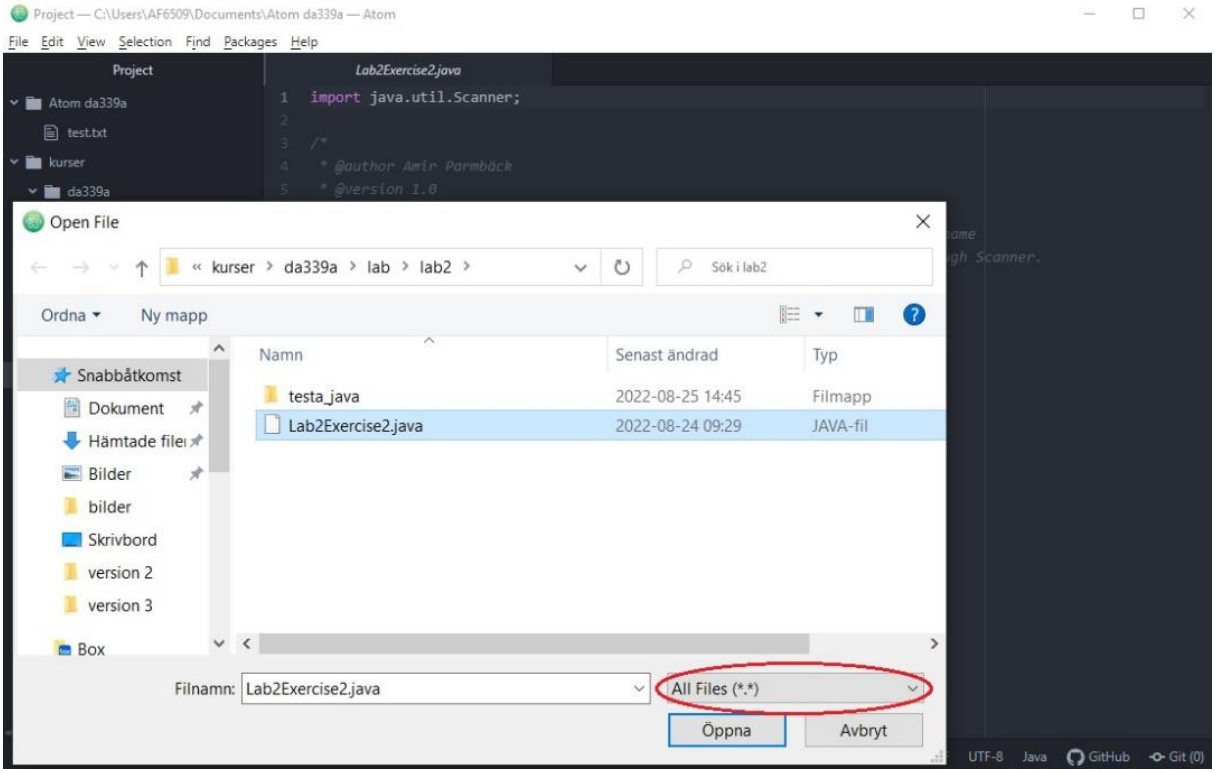
En editor är ett program i vilket man kan jobba med text eller programkod. Det är viktigt att du följer reglerna (syntaxen) för hur instruktionerna ska skrivas. Java är case-sensitive vilket betyder att Java skiljer på stor och liten bokstav (a och A behandlas som två helt olika tecken). Instruktionerna som programmet består av kallas för källkod. När du sparar instruktionerna i en fil på hårddisken (till exempel Lab2Exercise2.java som du kommer att använda nedan) kallas filen för källkodsfil.

I laborationen kommer du använda programmet Atom för att editera text som installerades under L1. Du får såklart använda andra editorer, dock rekommenderar vi att ni använder Atom.

När man programmerar skriver man ibland ett program från grunden och ibland jobbar man vidare med ett program som redan är skrivet. I denna laboration ska du jobba med ett redan färdigt program. På kurssidan hittar du programmet Lab2Exercise2.java. Hämta filen Lab2Exercise2.java och placera den i katalogen lab2 (som du skapade under laboration L1, ”Uppgift X Förberedelse för Lab2”).



Starta Atom och öppna Lab2Exercise2.java. Om du inte ser filen så kanske du måste ändra filformat till ”All files” för att Lab2Exercise2.java ska synas.



Uppgift 3

När du skrivit och sparat ett program som en textfil (tänk dig att du har skrivit Lab2Exercise2.java) är det dags att översätta programmets instruktioner till ett format som kan utföras av datorn. Denna översättning kallas för att kompilera källkoden. Det är ett speciellt program som utför kompileringen, nämligen **javac**. Programmet ingår i JDK. Programmet kallas för en kompilator. Programmet **javac** använder du från ett annat program, nämligen PowerShell (PC) / Terminalen (Mac) / Git Bash.

Nu gäller det att navigera till katalogen som innehåller Lab2Exercise2.java.

Och avsluta med RETURN (dvs. knappen med ↵ eller ”Enter”).

Du kan kontrollera att du hamnat rätt genom att skriva *pwd* följt av RETURN.
Datorn skall då svara med /Users/DittUsername (kan variera beroende vart du lagt filen).
Navigera till katalogen lab2 (eller vad din katalog nu heter) genom att skriva:

- *cd* da339a följt av RETURN
- *cd* lab följt av RETURN
- *cd* lab2 följt av RETURN

eller genom att hela sökvägen (*path* på engelska) skrivs

- *cd* da339a/lab/lab2 följt av RETURN

Har du hamnat fel och vill backa tillbaka en nivå så kan du skriva: *cd* .. (glöm ej mellanslaget mellan *cd* och ..). Då hamnar du i den katalogen som du kom ifrån. Kolla om du hamnat rätt

genom att skriva *pwd*. Du kan också skriva *ls*. Vilket är en förkortning av list så kommer du att få en lista med vad som finns i din katalog.

Kompilera Lab2Exercise2.java

Nu ska du ***kompilera*** Lab2Exercise2.java. Den aktuella katalogen ska vara lab2. Filen Lab2Exercise2.java ska finnas i lab2. Kompileringen sker genom att skriva instruktionen:

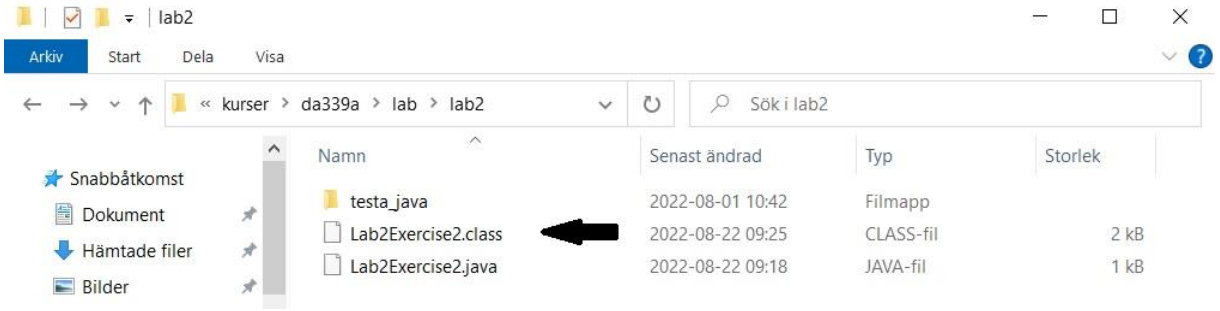
- javac Lab2Exercise2.java följt av RETURN.

Det är viktigt att du skriver källkodsfilens namn korrekt. Kom ihåg att Java är case sensitive och gör skillnad på liten och stor bokstav.

Nu startas programmet **javac.exe** och filen som ska kompileras är Lab2Exercise2.java. Resultatet av kompileringen är en fil som slutar med ”.class”, i det här fallet: *Lab2Exercise2.class*. Klassen placeras i samma katalog som källkodsfilen. Den nya filen innehåller bytekod och kallas oftast för bytekodsfil eller klassfil.

Kontrollera att Lab2Exercise2.class har skapats. Detta kan du göra genom att:

- Studera innehållet i katalogen lab2 med hjälp av File Explorer (PC) eller Findern (Mac).
- Studera innehållet i katalogen lab2 genom att skriva ls i kommandotolken och tryck på RETURN.



Exekvera programmet

Java kräver ett speciellt program, en virtuell maskin, för att programmet du skrivit ska exekveras. Detta program heter **java.exe** och ingår i JDK.

Den aktuella katalogen ska vara lab2. Exekvering sker genom att skriva instruktionen:
java Lab2Exercise2 följt av RETURN.

Det är viktigt att du skriver bytekodfilens namn korrekt och tar hänsyn till att Java är case sensitive.

Om allt är som det ska kommer detta visa sig på skärmen.

```
af6509@WC5CG1473GD7 MINGW64 ~/Desktop/kurser/da339a/lab/lab2
$ java Lab2Exercise2
Hej, ange ditt namn!
```

Skriv nu in ditt namn och avsluta med RETURN (enter), så skall det se ut så här.

```
af6509@WC5CG1473GD7 MINGW64 ~/Desktop/kurser/da339a/lab/lab2
$ java Lab2Exercise2
Hej, ange ditt namn!
seb
Ditt namn är seb
```

Uppgift 4

Nu ska du göra en mindre ändring i källkodsfilen. När du följer nedanstående instruktioner så gör du en mindre ändring i programmet som du jobbade med ovan.

1. Ändra i Lab2Exercise2.java:
   Öppna Lab2Exercise2.java i Atom om filen inte redan är öppen. Pröva att ändra texten genom att ta bort *Hej*. När du gjort ändringen ska du spara filen. Kompilatorn använder ju den sparade versionen.

```java
public class Lab2Exercise2 {
    public static void main(String args[]){
        Scanner scanner = new Scanner(System.in);
        System.out.println("Hej, ange ditt namn!");
        String name = scanner.nextLine();
        System.out.println("Ditt namn är " + name);
    }
}
```

2. Kompilera Lab2Exercise2.java:
   Gå till ***Kommandotolken*** och skriv in kompileringskommandot på nytt:
   a. javac Lab2Exercise2.java följt av RETURN.
3. Exekvera Lab2Exercise2:
   Skriv in exekveringskommandot på nytt:
   a. java Lab2Exercise2 följt av RETURN

```
af6509@WC5CG1473GD7 MINGW64 ~/Desktop/kurser/da339a/lab/lab2
$ java Lab2Exercise2
 ange ditt namn!
hej
Ditt namn är hej
```

Uppgift 5

Vi ska nu se vad som händer om man gör några vanliga fel i källkoden. Det är väldigt vanligt att ett program inte fungerar som tänkt vid första provkörningen. Om så är fallet måste man öppna sin källkodsfil och ändra i programmet. Därefter kompilera källkodsfilen på nytt för att slutligen exekvera programmet på nytt.

Här är ett exempel på vad som händer när något fel i koden när du försöker kompilera java filen.

```
af6509@WC5CG1473GD7 MINGW64 ~/Desktop/kurser/da339a/lab/lab2
$ javac Lab2Exercise2.java
Lab2Exercise2.java:18: error: reached end of file while parsing
}
 ^
1 error
```

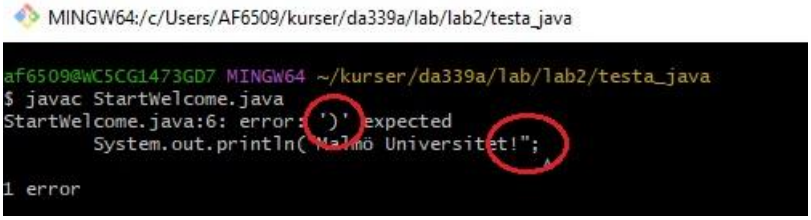
Om du försöker exekvera programmet efter en misslyckad kompilering får du ett meddelande i stil med.

```
af6509@WC5CG1473GD7 MINGW64 ~/Desktop/kurser/da339a/lab/lab2
$ java Lab2Exercise2
Error: Could not find or load main class Lab2Exercise2
Caused by: java.lang.ClassNotFoundException: Lab2Exercise2
```

De steg som man ska ta är alltså samma som i uppgift 2b:
1. Öppna källkodsfilen och gör nödvändiga ändringar. Spara sedan ändringarna.
2. Kompilera källkodsfilen. Om kompileringen inte går bra gå tillbaka till punkt 1.
3. Exekvera bytekodsfilen. Om resultatet inte är bra så gå tillbaka till punkt 1.

Uppgift 5.1 – Rätta fel i kod

Med hjälp av de tre stegen ovan ska du nu rätta ett program. Börja med att försöka kompilera (javac) java-filen *StartWelcome.java* (ligger i mappen testa_java) och se vad som händer.



I bilden ovan ser vi resultatet när vi försökte kompilera filen (notera det som är inringat). Försök rätta programmet med hjälp av bilden samt att granska alla tre kod raderna som börjar med ”System.out.println”. Glöm inte att spara ändringarna ni gör i Atom innan ni försöker kompilera filen igen. Bild på rättad kod finns sist i laboration 2 dokumentet.

Uppgift 6

I denna uppgift ska vi ska se andra sätt vi kan skriva in (input) samt visa resultat (output) på ett annat sätt än när vi använder Git Bash.

Öppna java-filen *InputString.java* (finns i testa_java mappen) i Atom och granska koden och kommentarerna som finns (gör inget om du inte förstår allt kod), därefter kompilera koden samt exekvera den. Programmet som körs ber dig att ange ditt namn i ett nytt fönster (input) och efter du gjort det kommer ännu ett nytt fönster att visa ditt namn plus ett meddelande (output).

Om du vill kan du själv testa att göra ändringar med koden du har genom att ändra på frågan (input) och svaret (output).

Uppgift 7 – LocalVariable visa att det finns olika variable typer decimal tal etc

I denna uppgift ska vi se hur man deklarera samt initierar variabel typerna *int* (heltal), *double* (decimal tal) och *long* (stora tal, kommer mer om detta senare i kursen). Kompilera och kör programmet *LocalVariable.java* (finns i testa_java) för att se att det funkar. Programmet skriver

ut tre olika tal (int, double och long), ändra *number* variabeln så att den blir 10.25 i stället och spara, därefter kompilera och kör programmet igen. Gick det att kompilera? Om inte varför?

Uppgift 8

Programmet *MathEx.java* syfte är att addera två tal, 12 och 6 (12+6). Kompilera java-filen och kör programmet, vad blir resultatet och stämmer det? Om det inte stämmer, öppna MathEx.java i Atom och försök rätta problemet. Kompilera programmet på nytt och se om du lyckats fixa problemet. Bild på rättad kod finns sist i laboration 2 dokumentet.

Svar på uppgifter

Uppgift 5.1:

StartWelcome.java

```java
public class StartWelcome1 {
    public static void main(String[] args) {

        System.out.println("Välkommen till ");
        System.out.println("Java-kurs vid");
        System.out.println("Malmö Universitet!"); // Det saknades ett ")"

    }
}
```

Uppgift 7:
Int kan endast vara heltal. Double kan vara decimaltal.

Uppgift 8:
Det ska vara ett ”+” och inte ett ”-” vid uträkningen.

MathEx.java

```java
public class MathEx {
    public void example() {
        int nbr1 = 12, nbr2 = 6; // Genom att skriva på detta sättet kan vi de

        System.out.println(nbr1 + nbr2); // Gör uträkningen
    }

    public static void main(String[] args) {
        MathEx prog = new MathEx(); // behöver ej bekymra dig om detta ännu!
        prog.example(); // anropar metoden example
    }
}
```