

Laboration 1.

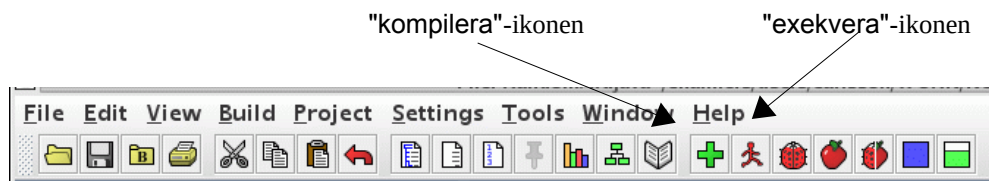
Syfte

Syftet med denna laboration är dels att göra dej bekant med de verktyg som kan vara aktuella i programmeringsarbetet, dels ge en första inblick i att skriva enkla Javaprogram och få övning i att nyttja **if**-, **while**- och **for**-satserna.

Del A: Förberedande övningar.

Uppgift 1 - Att kompilera och köra ett Java program

1. Logga in.
2. Skapa en ny mapp med namnet **ProgrammeradeSystem** (i din filarea på fileservern).
 - Klicka med höger mus -> Nytt -> Mapp.
 - Döp den nya mappen till **ProgrammeradeSystem**.
3. Skapa en ny mapp Lab1 i mappen **ProgrammeradeSystem**.
 - Gå till mappen **ProgrammeradeSystem** genom att dubbelklicka på mappen.
 - Klicka med höger mus -> Nytt -> Mapp.
 - Döp den nya mappen till **Lab1**.
4. Gå till kursens hemsida
<http://www.cse.chalmers.se/edu/year/2017/course/TDA143>
5. Lägg upp en snabbänk till kursens hemsida, det är inte sista gången du kommer att besöka denna sida.
6. Kopiera filen **Uppgift1.java**, som du hittar vid rubriken Laboration 1 under fliken Laborationer på kurshemsidan. Kopieringen av filen tillgår enligt följande:
 - Vänsterklicka på länken **Uppgift1.java** på kursens hemsida.
 - Högerklicka på filen **Uppgift1.java** och välj "Spara mål som .."
 - Ett fönster öppnas, dubbelklicka på mappen **ProgrammeradeSystem** och sedan på mappen **Lab1**.
 - Välj "Spara".
7. Kompilera programmet.
 - När du dubbelklicka **Uppgift1.java** (i mappen Lab1) skall **JGrasp** startas. Startas någon annan editor skall du kontakta handledaren.
 - Välja "kompilera"-ikonen (se figur nedan).



8. Kör programmet genom att välja "exekvera"-ikonen.
9. Studera programkoden och försök förstå varför programmet gör vad det gör.

Uppgift 2 - Kompileringsfel

När man skriver ett program kan man naturligtvis göra fel. En typ av fel är s.k. *kompileringsfel* vilka uppstår då man inte följer de språkregler som finns för det programspråk man använder. Denna typ av fel upptäcks av kompilatorn när man försöker kompilera programmet och man erhåller felmeddelanden från kompilatorn. Dessa felmeddelanden kan, särskilt för den ovane, ofta vara svåra att tyda. Denna uppgift går ut på att lära sig tyda felutskrifter från kompilatorn för att lokalisera och korrigera felen i ett program.

Inför nedanstående fel i programmet **Uppgift1.java** . Kompilera om programmet efter varje fel som införs och se vilka felutskrifter som erhålls. Återställ programmet till sitt ursprungliga utseende inför varje förändring.

- Ändra **Uppgift1** på rad 2 till **uppgift1**.
- Ta bort **void** på rad 3.
- Ta bort { på rad 3.
- Ändra **String** på rad 4 till **string**.
- Ta bort ; på rad 5.
- Sätt in ett semikolon (;) sist på rad 12.
- Ändra **==** på rad 15 till **=**.
- Ta bort } på rad 21.
- Ta bort [] på rad 3. Varför går programmet att kompilera, men inte att exekvera?

Tips: I felutskriften anger kompilatorn på vilken rad i programmet den upptäckt felet. För att få radnumren utskrivna i programmet i **jGrasp** går du in i menyn **View** och markerar **LineNumbers**.

Uppgift3 – Mer kompileringsfel

Nedanstående Javaprogram som innehåller ett antal fel finns på kursens hemsida.

```
public class TExFel {  
    public static void main(String[] arg) {  
        int tal1, tal2;  
        double tal3  
        indata = JOptionPane.showInputDialog("Ange första talet");  
        tal1 = parseInt(indata);  
        indata = JOptionPane.showInputDialog("Ange andra talet");  
        Tal2 = Integer.parseInt(indata);  
        tal3 = tal1 / tal2;  
        JOptionPane.showMessageDialog(null, "Resultatet blev" + tal3);  
    } //main  
} // ExFel
```

- 1) Kopiera programmet **ExFel.java** som finns lagrat på kursens hemsida, vid rubriken **Laboration 1** under fliken **Laborationer**, och lagra detta under namnet **ExFel.java** i ditt eget bibliotek **Lab1**.
 - Vänsterklicka på länken **ExFel.java** på kursens hemsida.
 - Högerklicka på filen **ExFel.java** och välj "Spara mål som .."
 - Ett fönster öppnas, dubbelklicka på mappen **ProgrammeradeSystem** och sedan på mappen **Lab1**.
 - Välj "Spara".
- 2) Ladda in filen **ExFel.java** i **jGrasp** trycka på **Open** i menyn **File**. Kompilera programmet. Försök förstå de felutskrifter som erhålls och försök rätta felen. Konsultera handledaren om ni får problem.

- 3) När ni lyckas kompilera programmet, kör programmet och ge värdena 2 och 5 som indata.
- Varför blir resultatet 0.0? Borde det inte vara 0.4?

Förklaring:

Värdet som skrivs ut beräknas i satsen

```
tal3 = tal1 / tal2;
```

I denna sats är båda operanderna, tal1 och tal2, i divisionen heltal, vilket i Java innebär en *heltalsdivision* som betyder att resultatet av divisionen blir ett heltal!! I vårt aktuella fall där 2 divideras med 5 blir resultatet 0. Att utskriften blir 0.0 beror på att resultatet som erhöles från divisionen lagras i variabeln tal3 som är av typen **double**, vilket innebär att heltalet 0 konverteras till det reella talet 0.0 när det lagras i variabeln.

För att erhålla ett reellt tal från en division måste minst en av operanderna vara ett reellt tal.

För att göra om heltalet heltal till ett reellt tal skriver man i Java (**double**) heltal.

Ändra i programmet så att indatavärdena 2 och 5 ger resultatet 0.4.

- Utskriften av resultatet blev inte så snygg (texten och talet skrivs ihop). Rätta till!

Uppgift 4

Kopiera filen **Triangel.java** som finns lagrat på kursens hemsida, vid rubriken **Laboration 1** under fliken **Laborationer**, och lagra detta i ditt eget bibliotek **Lab1** (under namnet **Triangel.java**). Kör programmet och försök förstå vad det gör.

Utgå sedan från programmet **Triangel** och skriv ett nytt program som via dialogrutor läser in värdena på de båda katettrarna i en rätvinklig triangel samt beräknar hypotenusan och skriver ut resultatet i en dialogruta. Döp ditt nya program till t.ex **Triangel2**.

Uppgift 5

Kopiera programmet **Cirkel.java** som finns lagrat på kursens hemsida, vid rubriken **Laboration 1** under fliken **Laborationer**, och lagra detta i ditt eget bibliotek **Lab1**. Kör programmet och försök förstå vad det gör.

Utgå från programmet **Cirkel.java** och skriv ett nytt program som förutom ytan också beräknar och skriver ut cirkelns omkrets. Döp ditt nya program till t.ex **Cirkel2.java**.

Uppgift 6

Kopiera programmet **Pris.java** som finns lagrat på kursens hemsida, vid rubriken **Laboration 1** under fliken **Laborationer**, och lagra detta i ditt eget bibliotek **Lab1**. Kör programmet och försök förstå vad det gör.

Utgå sedan från programmet **Pris** och skriva ett nytt program som ger 5 procents rabatt om totalpriset är större än 750 kronor, 10 procents rabatt om totalpriset är större än 1500 kronor och 15 procent om totalpriset är större än 3000 kronor. Ändra också utskriften på så sätt att uppgifter om bruttopris, rabatt och nettopris skrivs ut. Döp ditt nya program till t.ex **Pris2.java**.

Del B: Följande uppgifter skall redovisas för och godkännas av en handledare. Efter tredje föreläsningen skall ni kunna lösa samtliga dessa uppgifter. Sista dag för godkännande är fredag 27/1.

Uppgift 7

Skriv ett program som läser in dels den gällande växelkursen mellan Euro och svenska kronor, dels ett antal svenska kronor, samt beräknar och skriver ut hur många Euro detta motsvarar. Antalet erhållna Euro skall skrivas ut med exakt två decimaler.

Uppgift 8

Skriv ett program som läser utgångshastigheten v (m/sek) och en kastvinkel α (i grader) och sedan beräknar banhöjden h och kastlängden d enligt nedanstående formler för kast utan luftmotstånd:

$$h = \frac{v^2 \cdot \sin^2 \alpha}{2 \cdot g} \quad \text{och} \quad d = \frac{v^2 \cdot \sin 2\alpha}{g}$$

där tyngdkraftsaccelerationen $g = 9.81$ m/sek². Utskriften av banhöjd och kastlängd skall göras med ett lämpligt antal decimaler.

Tips: I Java finns en standardklass **Math**. Gå in på kursens hemsida under fliken **Java** och vidare till "specifikationen av API:n". Du kommer till en sida där alla standardklasser i Java finns beskrivna. Leta upp klassen **Math** och se efter vad det finns för "matnyttigt". Vilken enhet används i de trigonometriska funktionerna?

Testdata: En utgångshastighet på 20 m/sek och en kastvinkel på 45 grader ger en banhöjd på ca 10.19 meter och en kastlängd på ca 40.77 meter.

Uppgift 9

För att en båt skall klassas som en 12:a får inte nedanstående uttryck avvika från värdet 12 med mer än 0.05

$$\frac{2 \cdot d + \sqrt{A} + L - f}{2,37}$$

I formeln betecknar d båtens omfång, A båtens segelyta, L båtens längden och f båtens fribordshöjd. Skriv ett Javaprogram som läser in värden på d , A , L och f och avgör om båten uppfyller villkoret för att klassas som en 12:a.

Läs in samtliga indatavärden via en och samma dialogruta, dvs använd ett objekt av klassen **Scanner** för att avkoda de enskilda indatavärdena.

Testdata: Följande värden ger att båten är en 12:a
 $d = 6.3$, $A = 63.7$, $L = 8.85$ och $f = 1.02$

Obs! En instans av klassen **Scanner** anropar metoden **Local.getDefault()** för att ta reda på vilka konventioner som används för att exempelvis ange decimaltal. Den dator ni kör programmet på kan alltså ha svenska konventioner, dvs decimalkomma används istället för decimalpunkt. För att en instans av klassen **Scanner** som med säkerhet använder decimalpunkt vid inläsning av decimaltal skall ni innan ni skapar instansen av klassen **Scanner** skriva satsen

```
Locale.setDefault(Locale.UK);
```

för att få engelska konventioner.

Uppgift 10

Skriv ett program som läser ett datum på formen **yymmdd** (t.ex 991012) och skriver ut datumet på den amerikanska formen **mm/dd/yy** (t.ex 10/12/99). Använd dialogrutor för inmatning och utskrifter. Utforma programmet på så sätt att körningen upprepas tills man trycker på **Cancel**-knappen i inmatningsrutan. Du får anta att den indata som ges till programmet är ett korrekt datum på formen **yymmdd**. Uppgiften *skall* lösas med användning av heltalsdivision och rest vid heltalsdivision.

Tips: Vad blir resultaten av följande uttryck $1234/100$ och $1234\%100$?

Testdata: Resultatet av indatasträngen 010101 skall bli 01/01/01.

Uppgift 11

Den berömda matematikern Gottfried Leibniz gav följande formel för π

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \dots$$

- a) Skriv ett program som beräknar π enligt denna formel och tar med de 500 första termerna.

Metoden **Math.pow** får inte användas i lösningen!

Tips: Uttrycket är en summa. En summa är en serie termer som adderas. I vårt aktuella uttryck består termerna av en täljare och en nämnare. En term kan beräknas från den föregående termen. Nämnaren i den nya termen är 2 större än nämnaren i föregående term. Täljaren är till absolutbelopp alltid 1, men byter tecken mellan varje term (dvs är varannan gång positiv och varannan gång negativ).

- b) Skriv ett program som beräknar π enligt denna formel och tar med så många termer att den sist medtagna termen är den första termen som till sitt absolutbelopp är mindre än 0.00001.

Hur stor är noggrannheten i det värde du erhåller på π om den sist medtagna termen till absolutbelopp är mindre än 0.0001? Hur många decimaler bör du då skriva ut?

Uppgift 12

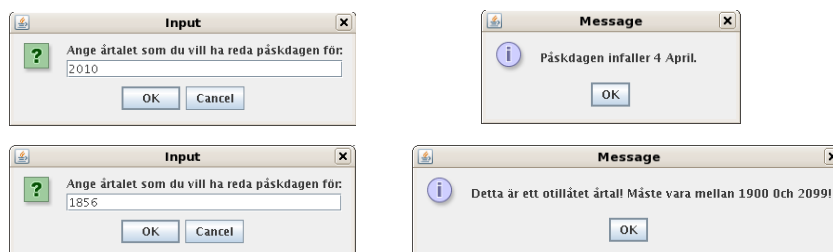
I Scientific American, Februari 1981, fanns följande algoritm för att beräkna vilket datum påskdagen infaller under åren mellan 1900 och 2099:

1. Kalla årtalet för Y. Subtrahera 1900 från Y och kalla skillnaden för N.
2. Dividera N med 19. Kalla resten (återstoden) för A.
3. Dividera (7A+1) med 19. Kalla kvoten för B.
4. Dividera (11A+4-B) med 29. Kalla resten för M.
5. Dividera N med 4. Kalla kvoten för Q.
6. Dividera (N+Q+31-M) med 7. Kalla resten för W.
7. Beräkna 25-M-W. Kalla resultatet D.
8. Om D är positivt infaller påskdagen i April och dagen är D, annars infaller påskdagen i Mars och dagen erhålls ur nedanstående tabell:

<u>D</u>	<u>Dag</u>
0	31
-1	30
..	..
-8	23
-9	22

Alla variabler i algoritmen är *heltal* och alla operationer är *heltalsoperationer*.

Skriv ett program som använda dialogrutor (enligt figuren nedan) för att upprepade gånger läsa in ett årtal, samt (med användning av ovan beskrivna algoritm) beräknar och skriver ut när påskdagen infaller för det inlästa årtalet. Exekveringen av programmet avbryts genom att användaren trycker på **Cancel**-knappen. Programmet skall kontrollera att det inlästa årtalet ligger mellan 1900 och 2099. Om felaktigt årtal ges skall en felutskrift göras.



Testdata: 2011 infaller påskdagen 24 april
2000 inföll påskdagen 23 april
1964 inföll påskdagen 29 mars
1910 inföll påskdagen 27 mars

Uppgift 13

Skriv ett program som läser in ett heltal N och genererar en multiplikationstabell med N rader och N kolumner enligt exemplet nedan:

Ange gradtal: **4**

$$1 * 1 = 1$$

$$1 * 2 = 2$$

$$1 * 3 = 3$$

$$1 * 4 = 4$$

$$2 * 2 = 4$$

$$2 * 3 = 6$$

$$2 * 4 = 8$$

$$3 * 3 = 9$$

$$3 * 4 = 12$$

$$4 * 4 = 16$$

Tips 1: Använd två nästlade **for**-satser - en yttre som styr antalet rader och en inre som styr antalet kolumner inom raderna.

Tips 2: Hur många kolumner skall den inre **for**-satsen skriva ut på den första raden? På den andra raden? På den tredje raden?