Programmering 1

Laboration 3: Logiska uttryck, beslutsfattande och iteration

Denna laboration är tänkt att både utföras och **redovisas** under de schemalagda timmar som finns anslagna för ändamålet. Ifall du inte har möjlighet att delta i laborationstillfällena, eller inte hinner slutföra laborationen under den schemalagda tiden måste du redovisa dina resultat i en **skriftlig laborationsrapport** som skickas in via Moodle. Ett obligatoriskt delkrav för att kunna bli godkänd i kursen är att **alla laborationer är godkända**.

Obs: Vissa uppgifter (markerade med en asterisk) kommer att gås igenom gemensamt under laborationstillfällena. Dessa uppgifter godkänns alltså automatiskt om du deltar i laborationstillfället.

1. ***(Mycket enkel) talsortering***

   Skriv ett program som läser in två flyttalsvärden från användaren och kontrollerar vilket av dem som är större. Några programkörningar kan t.ex. se ut som nedan:

```
Please input two values: 4.5 2.3
4.5 is larger than 2.3

Please input two values: 4.5 6.7
6.7 is larger than 4.5

Please input two values: 4.5 4.5
The values are equal.
```

   Kommentar: Eftersom flyttal är inexakta är det i regel ingen bra idé att använda sig av exakta jämförelser mellan två flyttal. I praktiken kan du ändå använda dig av denna teknik I denna uppgift, men vid den gemensamma genomgången kommer också en annan lösningsmetod att diskuteras.

2. ***Hundår***

   Det vanliga antagandet att sju människoår motsvarar ett hundår stämmer inte riktigt, eftersom en hund blir vuxen under sina första två år. En mer exakt beräkning kan göras enligt följande samband:
   - första hundåret = 1 människoår
   - andra hundåret = 6 människoår
   - varpå följande hundår = 7 människoår

   Skriv ett program som låter användaren ange antal människoår och räknar ut motsvarande antal hundår. Några programkörningar kan t.ex. se ut enligt följande:

```
How many human years? 1
Dog years: 1
```

```
How many human years? 2
Dog years: 7                    <= 1+6

How many human years? 5
Dog years: 28                   <= 1+6+7+7+7
```

3. ***While-loop, upprepade potensberäkningar***

Skriv ett program som med hjälp av en while-loop skriver ut följande resultat.

Observera att tecknet ^ brukar användas för att representera en potensberäkning (“upphöjt i”), men för att göra själva beräkningen måste du använda en av de metoder som behandlades vid föregående laboration.

```
Iteration 1: 2^1 is 2
Iteration 2: 2^2 is 4
Iteration 3: 2^3 is 8
Iteration 4: 2^4 is 16
Iteration 5: 2^5 is 32
```

Programmet får endast innehålla en **`printf`**-rad.

4. ***(*) Betygskalkylator***

Antag att en vitsordet i en kurs bestäms enligt följande:

50% av vitsordet bestäms av kurstentamen. Maxpoäng i kurstentamen är 30 poäng.
50% av vitsordet bestäms av inlämningsuppgifterna. Maxpoäng i varje inlämningsuppgift är 10 poäng.
Det finns totalt 5 inlämningsuppgifter, och varje uppgift bidrar därmed med 10% av det totala kursvitsordet.

De möjliga kursvitsorden är U (<50% totalt), G (mellan 50% och 75%) och VG (>= 75%).

Skriv ett program som låter användaren mata in kursresultat för en studerande och beräknar kursvitsordet. Använd en while-loop för att läsa in poängen från inlämningsuppgifterna. Programkörningen kan t.ex. se ut enligt nedanstående:

```
Please enter the exam result (0-30 points): 25
Please enter the result for assignment 1 (0-10 points): 6
Please enter the result for assignment 2 (0-10 points): 10
Please enter the result for assignment 3 (0-10 points): 9
Please enter the result for assignment 4 (0-10 points): 10
Please enter the result for assignment 5 (0-10 points): 8

The final grade is 84.7%, which corresponds to the VG grade
```