Linköpings universitet
Institutionen för datavetenskap (IDA)
UPP-gruppen 2015-11-01

# Satser för val och repetition

## Mål

I denna inledande laboration kommer du att öva på de grundläggande styrstrukturerna för repetition och val samt enkel in- och utmatning.

Skriv och testa momstabellprogrammet. För samtliga uppgifter gäller att snygga utskrifter ska åstadkommas med hjälp av formatdirektiv (manipulatorer) till *cout*, samt att du ska kunna motivera valet av iterationssats.

I programmet ska felhantering av användarens inmatning göras. Felhanteringen ska vara av typen rimlighetskontroll vilket innebär att användaren alltid kommer att mata in data av rätt datatyp. Detta ska göras så fort som felet kan upptäckas. Programmet ska ej avbrytas utan ny fråga ska ställas om användaren matar in ett felaktigt data! Tänk på att användaren kan mata in felaktiga data många gånger!

## Läsanvisningar

- Styrsatser (`if`, `for`, while, `do-while`, `switch`)
- Datatyper (`int`, `char`, `float`, `double`)
- Enkla funktioner (enkla argument och returvärde)
- Konstanter (`const`)
- Aritmetiska operatorer (+, -, *, /, %)
- Jämförelse och logiska operatorer (<, >, <=, >=, ==, !=, !, &&, ||)
- Typkonvertering (`stoi`, `stod`, `to_string`)
- Standard felutmatningsström (`cerr`)
- Formaterad utmatning (`setfill`, `setw`, `left`, `right`)

# Uppgift: Momstabellen

Konstruera ett program som skriver ut en momstabell. Programmet ska på terminalen fråga efter och ta som inmatning följande värden (där alla värden ska rimlighetskontrolleras!):

- Nedre samt övre gräns för prisintervallet
- Steglängd i tabellen
- Momsprocenten (uttryckt som decimaltal i intervallet 0 till 100 %)

Det som är viktigt i denna laboration är att användaren ska känna att den har stöd från programmet om den matar in felaktiga data (felaktiga i detta fall innebär att man matat in ett flyttal, men att inmatade värdet är orimligt i relation till vad programmet frågar efter). De körexempel nedan visar hur progammet ska bete sig för just dessa indata. Om man matar in andra data blir givetvis data i tabellen annorlunda, men själva utseendet ska följa den stil som syns i körexemplen. Det är givet att användaren ALDRIG kommer att mata in något data som är annat än flyttal och dessutom kommer dessa flyttal alltid att ligga i intervallet [-100.00, +100000.00]. Användaren kommer heller aldrig att mata in mer än 2 decimaler.

*KRAV*: Du ska skriva en enkel funktion för att beräkna momsen och en enkel funktion för att beräkna priset inklusive moms. Båda funktionerna ska ta priset exklusive moms och momssatsen som parametrar och returnera momsen respektive priset inklusive moms.

*KRAV*: Du får bara använda variabler av heltalstyp samt datatypen *float*. Du får alltså *inte* använda datatypen double. Detta är för att du tydligare ska se de problem flyttal kan ge (som du råkar ut för även med datatypen double om du visar fler decimaler).

*KRAV*: Du får *inte* använda typkonvertering

Du ska själv avgöra vad som är rimliga indata för respektive inmatning. Tanken är att användaren ska köpa saker från t.ex. en affär och att affären aldrig kommer att acceptera att den ska betala köparen något.

Ett lämpligt tillvägagångssätt när man programmerar är att man börjar med antagandet att användaren är "snäll" och matar in korrekta och rimliga värden till programmet. Detta gör att man dels slipper hantera felaktiga indata (d.v.s. inga felkontroller behövs i programmet) och dels att man snabbt får fram ett program som "löser" uppgiften.

När man fått sitt program att fungera givet att användaren matar in korrekta data går man vidare till steg två. Detta innebär att man lägger till felhantering av indata. Användaren ska givetvis få sina felmeddelanden så fort det överhuvudtaget är möjligt för att det ska vara bra. Felmeddelanden ska givetvis vara informativa och vänliga så att användaren känner att den är väl omhändertagen.

*TIPS*: Försök bryta ut felhanteringen i funktioner för att öka läsligheten i din kod.

Följ ovanstående tips även i resterande laborationer så tjänar du en massa tid.

Här följer två programkörningar givet just dessa förutsättningar. Programmet ska ge resultat enligt nedan vid körning (användarens inmatningar markeras nedan genom att texten är kursiverad och det

som programmet skriver ut är i normalt teckensnitt):

**Körexempel 1**

```
INMATNINGSDEL
============
Mata in första pris: 10.00
Mata in sista pris: 15.00
Mata in steglängd: 0.5
Mata in momsprocent: 10.00

MOMSTABELLEN
============
        Pris              Moms       Pris med moms
--------------------------------------------------
       10.00              1.00               11.00
       10.50              1.05               11.55
       11.00              1.10               12.10
           .                 .                  .
           .                 .                  .
       15.00              1.50               16.50
```

**Körexempel 2**

```
INMATNINGSDEL
============
Mata in första pris: 10.00
Mata in sista pris: 12.00
Mata in steglängd: 0.3
Mata in momsprocent: 20.00

MOMSTABELLEN
============
        Pris              Moms       Pris med moms
--------------------------------------------------
       10.00              2.00               12.00
       10.30              2.06               12.36
       10.60              2.12               12.72
           .                 .                  .
           .                 .                  .
       11.80              2.36               14.16
```

OBS! Sista “Pris utan moms”-värdet i andra exemplet är 11.80 och inte 12.00 (inte heller 12.10)! Detta beror på att det inte går jämnt upp med steglängden givet start- och slutpris. Inmatat “sista pris” ska då inte komma med i tabellen.

Här följer ett körexempel där användaren börjar med att mata in ett felaktigt data. I denna uppgift är det meningen att du ska fundera ut fler fall med orimliga indata(kombinationer) och ge vettiga felmeddelanden för dem. Programmet ska köras tills användaren matar in korrekt data.

### Körexempel 3

```
INMATNINGSDEL
============
Mata in första pris: -10.00
FEL: Första pris måste vara minst 0 (noll) kronor
Mata in första pris: 100.00
Mata in sista pris: 101.00
Mata in steglängd: -10.3
FEL: steglängd måste vara minst 0.01
Mata in steglängd: 0.1
Mata in momsprocent: 12.00

MOMSTABELLEN
============
        Pris             Moms       Pris med moms
-------------------------------------------------
      100.00            12.00              112.00
      100.10            12.01              112.11
      100.20            12.02              112.22
           .                .                   .
           .                .                   .
      101.00            12.12              113.12
```

Lämpliga testdata (det finns fler, men detta är en uppsättning att börja med) kan vara (kombinera lite olika för att se om ditt program verkar fungera):

```
Första pris: -1  0  10   100
Sista pris:  -1  0  1    11    12   15   101  100000
Steg:        -1  0  0.1  0.25  0.3  0.5  1    10
Momsprocent: -1  0  1    20    50   100  101
```

*TIPS*: När man testar sitt program är det viktigt att man är noggrann. Man bör skriva upp vilka testfall man vill ha och vad man förväntar sig för resultat. Dessutom bör man testa alla testfall igen om man rättar sitt program när man hittat fel. En bra testare ser “utanför begränsningarna”. Det innebär att man ser användaren som en person som inte kanske har förstått hur den ska mata in saker

och på det viset kan det bli många roliga testfall.

*OBS*: Denna uppgift begränsas till att vi håller oss till vissa indata när vi testar era program. Inga värden utanför intervallet [-100.00, 100000.00] kommer att användas och aldrig fler än 2 decimaler.