Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

TDT4102 Prosedyre- og objektorientert programmering Vår 2016

Øving 2

Frist: 2016-01-29

Mål for denne øvingen:

- lære grunnleggende C++ og prosedyreorientert programmering
- lære hvordan programmer lese inn data fra brukeren og skrive tekst ut på skjermen
- lære hvordan funksjoner kan ta inn data som parametre og returnere verdier

Generelle krav:

- Bruk de eksakte navn og spesifikasjoner gitt i oppgaven.
- Det er valgfritt om du vil bruke en IDE (Visual Studio, Xcode), men koden må være enkel å lese, kompilere og kjøre.

Anbefalt lesestoff:

- Kapittel 1, 2 og 3, Absolute C++ (Walter Savitch)

1 Funksjoner og «Input/Output» (25%)

For at programmer ikke skal bli uoversiktlige deler man dem opp i funksjoner, som inneholder gjenbrukbar kode. Denne koden kan bli kjørt fra andre steder i programmet ved hjelp av et funksjonskall. (Alle programmer har minst en funksjon. Når programmet startes vil operativsystemet kjøre funksjonen som heter `main`.)

I det følgende eksempelprogrammet har vi to funksjoner i tillegg til `main`-funksjonen. Den ene, `add`, tar inn to argumenter og legger de sammen, for deretter å *returnere* verdien til det stedet i programmet der funksjonen ble kjørt. Utskriften av resultatet til skjerm skjer der funksjonen ble kjørt fra, dvs. main-funksjonen.

Den andre, `inputIntegersAndPrintProduct`, har ingen parametre, men leser inn data fra brukeren og skriver ut til skjerm. Den returnerer ingenting, og har derfor returtypen `void`.

```cpp
#include <iostream>

// En funksjon som har paramtre og returverdi
// Sagt på en annen måte: En funksjon som tar inn argumenter og
// returnerer en verdi
int add(int a, int b) {
    return a + b;
}

// En funksjon som leser inn input fra bruker og skriver ut til skjerm
void inputIntegersAndPrintProduct() {
    int x = 0;
    int y = 0;
    std::cout << "Skriv inn to heltall: ";
    std::cin >> x;
    std::cin >> y;

    int product = x * y;

    std::cout << x << " * " << y << " = " << product << std::endl;
}

int main() {
    int sumOfOneAndTwo = add(1, 2);
    std::cout << "1 + 2 = " << sumOfOneAndTwo << std::endl;

    // Funksjonskall kan settes rett inn der man vil ha verdien:
    std::cout << "2 + 2 = " << add(2, 2) << std::endl;

    inputIntegersAndPrintProduct();
}
```

I dette faget er det viktig å forstå forskjellen mellom konseptene «lese inn fra brukeren» og parametre/argumenter i funksjoner, samt mellom konseptene å «skrive ut til skjerm» og det å returnere verdier fra funksjoner. Hensikten til denne oppgaven er at vi skal bli sikrere på dette.

a) Skriv en funksjon som heter `inputAndPrintInteger` som lar brukeren skrive inn et heltall og skriver det ut til skjerm. Test funksjonen ved å kjøre den fra `main`, slik:

```
int main() {
    inputAndPrintInteger();
}
```

Eksempel på resultat av å kjøre programmet:

```
Skriv inn et tall: 42
Du skrev: 42
```

b) Skriv en funksjon som heter `inputInteger` som lar brukeren skrive inn et heltall og *returnerer* dette. Test funksjonen ved å kjøre den fra `main`, slik:

```
int main() {
    int number = inputInteger();
    std::cout << "Du skrev: " << number;
}
```

Eksempel på resultat av å kjøre programmet:

```
Skriv inn et tall: 123
Du skrev: 123
```

c) Skriv en funksjon `inputIntegersAndPrintSum` som ved å bruke en av funksjonene du nå har skrevet, leser inn to heltall fra brukeren og skriver ut kun summen til skjermen. Eksempel på resultat av å kjøre programmet:

```
Skriv inn et tall: 3
Skriv inn et tall: 4
Summen av tallene: 7
```

d) Skriv en kommentar i koden din som forklarer hvilken funksjon av `inputAndPrintInteger` og `inputInteger` du brukte for å lage `inputIntegersAndPrintSum`, og hvorfor.

e) Skriv en funksjon som heter `isOdd`, som tar inn et heltall og returnerer en boolsk verdi. Funksjonen skal returnere `true` dersom argumentet er et oddetall og `false` ellers. Den skal *ikke* lese noe inn fra brukeren eller skrive ut noe til skjermen.

En effektiv måte å teste funksjonen på ved hjelp av en for-løkke kan se slik ut:

```
int main() {


    for (int i = 10; i < 15; i++) {
        if (isOdd(i)) {
            std::cout << i << " er et oddetall." << std::endl;
        } else {
            std::cout << i << " er et partall." << std::endl;
        }
    }
}
```

Eksempel på resultat av å kjøre programmet:

```
10 er et partall.
11 er et oddetall.
12 er et partall.
13 er et oddetall.
14 er et partall.
```

f) Skriv en funksjon som heter `printHumanReadableTime` som tar inn antall sekunder som en parameter, og skriver dette ut til skjerm i et menneskevennlig format. For eksempel vil 10000 sekunder skrives ut som «2 timer, 46 minutter og 40 sekunder».

Programmet ditt trenger ikke være grammatisk korrekt, du kan med andre ord skrive ut «1 minutter». (Hvis du ønsker å gjøre programmet ditt grammatisk korrekt, kan du skrive ut entallbøyningen av *time*, *minutt* og *sekund* etterfulgt av «r» eller «er» *hvis* (if) antallet ikke er 1.)

Hint: Du kan bruke modulo-operatoren (%) for å forenkle koden din. Modulo gir resten om man dividerer venstre side av utrykket på høyre. For eksempel vil 72 % 60 gi 12.

2 Løkker (15%)

a) Lag en funksjon som heter `inputIntegersUsingLoopAndPrintSum`, som tar utgangspunkt i koden i funksjonen `inputIntegersAndPrintSum` fra oppgave 1. Den nye funksjonen skal la brukeren velge hvor mange tall som skal summeres, enten ved å angi antallet tall først, eller ved å slutte når brukeren skriver inn tallet 0.

b) Den forrige deloppgaven kan løses enten ved at brukeren angir antall tall som skal summeres i forkant, eller at man fortsetter å lese inn tall helt til brukeren skriver inn 0. **Skriv en kommentar i koden din som forklarer hvilken type løkke som er best egnet for hver av de to alternativene, og hvorfor.**

c) Skriv funksjonen `inputDouble` som skal gjøre det samme som `inputInteger` fra oppgave 1, men istedenfor å lese inn et heltall, skal denne funksjonen lese inn og returnere et desimaltall.

d) Skriv en funksjon som konverterer NOK til Euro.

La brukeren gi beløpet som skal konverteres som et positivt desimaltall. Hvis brukeren gir et negativt tall skal programmet spørre etter et nytt tall. Skriv ut det vekslede beløpet med to desimaler.

Bruk 1 EUR = 9,64 NOK.

Gjenbruk tidligere skrevne funksjoner så langt som mulig, og test funksjonen fra `main()`.

e) Skriv en kommentar i koden din som forklarer hvorfor vi ikke bør bruke `inputInteger` i forrige oppgave, men heller `inputDouble`, legg spesielt merke til hva i oppgaveteksten som legger føring på denne bruken. Kommenter også på returtypen til funksjonen du skrev i forrige oppgave.

3 Menysystem (10%)

a) Hittil har vi testet funksjonene vi har skrevet på en ganske usystematisk måte, ved å prøve dem ut i `main`. Dette skal vi nå rydde opp i.

Skriv om `main` slik at brukeren kan velge i en meny mellom funksjonene fra foregående oppgaver, eksempel:

```
Velg funksjon:
0) Avslutt
1) Summer to tall
2) Summer flere tall
3) Konverter NOK til EURO.
Angi valg (0-3):
```

Hvis brukeren f.eks. velger 2, skal funksjonen som lar brukeren angi tall for summering kjøre, når denne er ferdig, skal menyen kjøre på nytt, programmet skal ikke avslutte før brukeren selv velger dette ved å angi menyvalget for «Avslutt».

b) **Skriv en funksjon som skriver ut en gangetabell på skjermen (cout). La brukeren gi både bredde og høyde på tabellen.** Velg selv navn for funksjonen.

Hint: Løkke i løkke.

Legg denne funksjonen til i testmenyen

4 Bruk av funksjoner i funksjoner, og røtter (25%)

I funksjonene under skal du tolke hva som skal være argumenter til funksjonen, og lære å bruke funksjoner i funksjoner. Generelt heretter i faget, skal funksjoner returnere verdier, og ikke skrive ut noe til skjerm, med mindre annet er spesifisert.

a) **Skriv en funksjon `discriminant` som regner ut**

$$b^2 - 4ac$$

og returnerer verdien (ingen utskrift til skjerm).

b) **Skriv en funksjon `printRealRoots` som finner de reelle røttene til andregradsligningen**

$$ax^2 + bx + c = 0$$

ved å bruke formelen

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

der a, b og c er gitt som argumenter til funksjonen. Gjenbruk funksjonene fra de forrige deloppgavene, og skriv ut løsningene til skjerm.

Hint: Bruk en funksjon fra standardbiblioteket til å finne kvadratroten. Slå gjerne opp på http://www.cplusplus.com/ for å finne ut hvilke funksjoner som er tilgjengelige.

Ligningen har 2, 1, eller 0 reelle løsninger (vi ser bort fra imaginære løsninger). Dette bestemmes ved at:

2 løsninger dersom	$b^2 - 4ac > 0$
1 løsning dersom	$b^2 - 4ac = 0$
ingen løsninger dersom	$b^2 - 4ac < 0$

Du kan anta at $a \neq 0$.

c) **Lag en funksjon `solveQuadraticEquation` som lar brukeren skrive inn 3 desimaltall og bruk `printRealRoots` til å regne ut røttene til andregradsuttrykket gitt ved disse tallene.**

d) Legg til `solveQuadraticEquation` i testmenyen i `main()`.

e) Bruk testmenyen til å finne røttene til disse andegradsligningene. Sjekk at programmet fungerer med tall som gir 0, 1, og 2 løsninger.

$$x^2 + 2x + 4 = 0$$
$$4x^2 + 4x + 1 = 0$$
$$8x^2 + 4x - 1 = 0$$

NB: Generelt i øvingsopplegget bør dere gjøre noe tilsvarende dette for å teste at koden deres fungerer som den skal.

5 Flere løkker (25%)

a) Skriv en `calculateLoanPayments` funksjon som regner ut årlige innbetalinger av et lån over 10 år. La brukeren spesifisere lånebeløp og rente. La programmet for hver innbetaling skrive ut (`cout`) størrelsen på innbetalingen og det gjenstående lånet. Utskriften skal være i et lettleselig format.

Bruk følgende formel for å regne ut innbetalingene:

$$\text{Innbetaling} = \frac{\text{TotaltLån}}{\text{AntallAvdrag}} + \frac{\text{Rente}}{100} * \text{GjenståendeLån}$$

Du trenger kun regne med én innbetaling i året, og denne skjer ved slutten av året. Renter skal ikke legges til eller trekkes fra det gjenstående lånebeløpet, men kun regnes med i innbetalingen.

Legg denne funksjonen til i testmenyen.

b) Utvid funksjonen fra forrige deloppgave til å skrive ut oversikten over innbetalinger som en pen og pyntelig tabell. Dersom vi velger en rente på 5%, et totalt lån på 100 000 kroner og 10 avdrag, vil de første linjene i oversikten se slik ut:

```
År          Innbetaling          Gjenstående Lån
1           15000                90000
2           14500                80000
```

Hint: Bruk \t som tabulator i utskriften.