Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

TDT4102 Prosedyre- og objektorientert programmering Vår 2016

Øving 0 for Windows

Frist: Ingen (frivillig øving)

Mål for denne øvingen:

- Bli kjent med programmeringsverktøy
- Lage et første program kun med teksteditor og kompilator
- Lage et første program med Visual Studio

Denne øvingen er mer en veiledning i hvordan å lage et program. Målet er å veilede deg gjennom prosessen å lage et første program. Hvis du gjennomfører denne øvingen og blir fortrolig med skriving, kompilering og kjøring av egne program vil du komme fortere i gang med de obligatoriske øvingene. I denne øvingen går første del ut på å kode ved hjelp av en vanlig teksteditor og kompilere fra kommandolinjen.

NB: Denne øvingen er utformet på basis av Visual Studio 2015. Dersom du bruker en tidligere versjon av programmet garanteres det ikke at instruksene stemmer nøyaktig. Det anbefales sterkt å oppgradere til Visual Studio 2015.

Aktuelle kapitler i boka:

- Kap. 1 (1.1 og deler av kap 1.3) i Absolute C++ (Walter Savitch)

Oppgave 1 – Bli kjent med kompilatoren

Skriving av kildekode og kompilering kan i prinsippet gjøres med enkle verktøy. En helt vanlig teksteditor[1] er alt du trenger for å skrive kode, og kompilering kan gjøres ved å kjøre kompilatoren fra kommandolinja.

For å kompilere kode trenger du altså en kompilator. Her antar vi at du bruker Microsofts kompilator `cl`, som følger med Visual Studio. Selv om NTNU-studenter har tilgang til betalutgaver av Visual Studio 2015 gjennom MSDNAA/Dreamspark, som kan være nyttig for senere bruk, er gratisutgaven Visual Studio 2015 Community tilstrekkelig for vårt bruk og det er den versjonen som vil bli brukt til demonstrasjoner i forelesninger og øvingsforelesninger. Visual Studio tar ofte lang tid (opptil flere timer) å installere, så installer programmet på forhånd og ikke i en forelesning eller øvingstimen til studassen din.

I denne oppgaven kan du f.eks. bruke kildekoden som er vist nedenfor dette avsnittet eller du kan skrive av andre eksempler i boken. Kopier og lim det som er mellom strekene.

```
// Dette er et helt enkelt program som du kan kopiere og bruke i
// denne oevingen.
#include <iostream>

int main() {
        std::cout << "Hello World!" << std::endl;
        return 0;
}
```

Oppgave 1.1 - Lagring av egen kildekode

1. Start en vanlig teksteditor. Kopier og lim inn eksemplet over. Opprett en ny katalog i hjemmemappen din og lagre filen din der, for eksempel med navnet `HelloWorld.cpp`. Sjekk mappen hvor filen din er lagret, og forsikre deg om at den er der og har riktig navn. Det er vanlig konvensjon at kildekodefiler for C++ har filtypenavnet (delen av filnavnet etter punktum) `cpp`. Ved å bruke riktig filtypenavn oppnår du at mange verktøy automatisk skjønner at filen inneholder C++-kode. Hvis du bruker et annet filtypenavn kan det hende at du ikke får kompilert koden.

[1] Her kan du blant annet bruke Notepad, som følger med Windows. Da Notepad er en ganske begrenset teksteditor, lønner det seg gjerne å laste ned en mer avansert teksteditor, for eksempel Notepad++. Med Notepad++ får du både syntax highlighting og mye annen funksjonalitet.

Oppgave 1.2 - Kompilering fra kommandolinjen på Windows

1. For å kompilere fra kommandolinjen trenger vi først å starte opp et kommandolinjevindu. Dersom du åpner kommandolinjen på vanlig måte med `cmd.exe`, vil man hver gang man ønsker å kjøre C++-kompilatoren `cl` måtte spesifisere hvilken mappe denne ligger i. Det er tungvindt.

2. I stedet skal vi bruke «Developer Command Prompt for VS2015». For å starte denne, gå inn på start-menyen (Windows 7/10) eller start-skjermen (Windows 8) og begynn å skrive «Developer Command Prompt». Du skal få opp programmet i listen – start dette. Dersom du bruker Windows 7 eller 10 finner du også «Developer Command Prompt» i «Visual Studio 2015»-mappen i startmenyen.

3. Vi skal nå navigere oss fram til mappen der du lagret `HelloWorld.cpp`. For å navigere mellom mapper i Windows-kommandolinjen bruker man kommandoen `cd` (**c**hange **d**irectory).

   Når du først starter «Developer Command Prompt» vil du antakelig befinne deg i mappen som Visual Studio 2015 er installert i, for eksempel `C:\Program Files\Visual Studio 2015`. Dersom mappen du lagret `HelloWorld.cpp` i ligger på en annen disk enn Visual Studio, må du først bytte til den. Det gjør du ved å skrive stasjonsbokstaven etterfulgt av kolon, f.eks. `D:`. Hvis den ligger på samme disk, kan du nå gå «nedover» i mappehierarkiet ved å skrive kommandoen `cd ..` (to punktum) helt til du havner i *roten* av disken, f.eks. `C:\`.

4. Gå nå «oppover» i mappehierarkiet til du kommer til mappen der du lagret `HelloWorld.cpp`, ved å igjen bruke `cd`, nå med navnet på mappen du vil gå inn i. Hvis `HelloWorld.cpp` ligger inni hjemmemappen din, vil du for eksempel først skrive `cd Users`, så `cd brukernavn`, og så videre. For å gå inn i mapper med mellomrom i navnet, kan du bruke hermetegn rundt mappenavnet. For å se hvilke filer og mapper som ligger i mappen du befinner deg i, kan du bruke kommandoen `dir`.

5. Når du er kommet til mappen der `HelloWorld.cpp` ligger, skal vi kompilere denne til en programfil. For å gjøre dette skriver du

   ```
   cl HelloWorld.cpp
   ```

6. Sjekk nå hva som ligger i mappen, enten i Windows-utforskeren eller med `dir`, og se hvilke filer som er produsert. `obj`-filen er en midlertidig fil du ikke trenger å bry deg om, mens `exe`-filen er det endelige programmet ditt. Kjør programmet du har laget ved å skrive `HelloWorld.exe`, og sjekk utskriften.

7. Rediger teksten i `HelloWorld.cpp` slik at programmet skriver ut noe annet (husk å ha med hermetegnene rundt teksten).

8. Kompiler og kjør på nytt.

Oppgave 1.3 - Hva skjer hvis koden min er feil?

En god del av tiden du bruker på å gjøre øvinger vil bestå i å finne ut hva som er feil i koden din. En type feil er syntaktiske feil som resulterer i kode som ikke vil kompilere. Hvis du

prøver å kompilere kode som ikke er riktig skrevet vil kompilatoren gi deg feilmelding(er) som inneholder informasjon om hva som kan være galt. Noen ganger er dette forståelig informasjon, andre ganger kan det være vanskeligere å skjønne feilmeldingene.

Du skal nå med vilje «ødelegge» koden din ved å endre på småting, og deretter observere hva slags feilmeldinger du får når du kompilerer. Du kan for eksempel gjøre følgende:

1. Fjern semikolonet på slutten av linjen

   ```
   std::cout << "Hello World!" << std::endl;
   ```

2. Lagre filen, kompiler og sjekk feilmeldingen du nå får. Forstår du feilmeldingen?
3. Husk å rette opp filen igjen før du går videre.
4. Introduser andre feil og les feilemeldingene som kompilatoren gir. Du kan f.eks. prøve å slette en av krøllparantesene, skrive `cout` som `cut` osv.
5. Les feilmeldingene du får og rett opp slik at filen din blir riktig igjen etterpå.

Noen tips til å forstå feilmeldinger fra kompilatoren:

- En feilmelding fra kompilatoren `cl` er vanligvis på formen

  ```
  kildekode.cpp(4): error C2653: 'st': is not a class or namespace name
  ```

  Første del, `kildekode.cpp(4)`, inneholder filnavnet til filen der feilen befinner seg, samt et tall, som er linjenummeret der kompilatoren tror feilen befinner seg.

- Kompilatoren har ikke alltid rett i på hvilken linje feilen ligger. Ofte er det snakk om «følgefeil» fra en tidligere linje. Når du får opp mange feilmeldinger på én gang, lønner det seg ofte å se på de første feilmeldingene.

- Etter linjenummeret står det hvilken *type* feilmelding det er snakk om. Dette kan enten være «error» eller «warning». En «error» vil avbryte kompileringen uten at kompilatoren produserer en programfil. Dersom du får en «warning» vil kompileringen fortsette, men disse indikerer at koden din gjør noe du antakelig ikke vil at den skal gjøre. Husk at selv om koden din kompilerer, er det ikke sikkert at det resulterende programmet *virker* slik du vil! Får du en «warning» når du kompilerer, bør du ta en ekstra kikk på koden din før du programmerer videre.

- `cl` gir også ut noen advarsler som ikke har noe med koden din å gjøre, men derimot hvilke innstillinger som er valgt for kompilatoren. Disse advarslene er kjennetegnet ved at de begynner med filbanen til Visual Studio-mappen, og kan ignoreres.

- Deretter følger en feilkode (som f.eks. kan googles) samt beskrivelsen av feilen, ideelt uttrykt på en forståelig og ikke altfor generell måte. For mindre feilmeldinger som er grunnet i syntaksfeil og andre småfeil vil beskrivelsen vanligvis være konsis og forståelig, men når man begynner å bruke mer avanserte deler av C++ kan den være vag eller til og med direkte misledende. Da kan det være lurere å gå grundig gjennom koden i nærheten av der feilen oppstod og se om man ser en feil.

Feilmeldinger fra C++-kompilatoren kan være vanskelig å forstå, særlig når de er misvisende. Ikke sitt og dra deg i håret av den grunn! Både studassen din, emnets diskusjonsforum på It's learning og nettressurser som stackoverflow.com er uvurderlige hjelpemidler når det ser aller mest håpløst ut.

Oppgave 2 - Programmering med Visual Studio 2015

I denne oppgaven antas det at du allerede har installert Visual Studio 2015, som beskrevet i Oppgave 1.

Oppgave 2.1 - Start opp Visual Studio

1. Start Visual Studio 2015 fra startmenyen/startskjermen.
2. Velg **New project** og opprett et prosjekt av typen **Visual C++** → **General** → **Empty Project**. Kall prosjektet f.eks. «HelloWorld», og lagre prosjektet i en egnet mappe.
3. I sidelinjen til høyre vil du se et mappetre, blant annet med en mappe kalt «Source Files». Høyreklikk på denne mappen og velg **Add** → **New Item**. Velg deretter **C++ File (.cpp)**. Kall den for eksempel `HelloWorld.cpp`, og trykk **Add**.
4. Du vil nå få opp den tomme filen du nettopp lagde. Kopier inn kildekoden fra Oppgave 1 og lagre.
5. Kompilering i VS gjøres ved hjelp av menyvalget **Build** → **Build Solution** (eller ved å trykke `Ctrl+Shift+B`). Prøv å finne `HelloWorld.exe` (filnavnet avhenger av hva du kalte prosjektet) som nå ble opprettet (tips: let i mappen du lagret prosjektet ditt i).
6. Kjør programmet ditt ved å trykke `Ctrl+F5`, eller ved å gå inn i menyen **Debug** og velge **Start Without Debugging**. Hva skjer? Jo, programmet ditt blir kjørt, men det blir umiddelbart ferdig og avsluttes, slik at du ikke fikk se teksten du skrev ut! Dette hadde ikke vært et problem om programmet ditt forventet input fra brukeren, men her er det eneste som skjer at en tekststreng skrives til skjermen.
7. For å fikse dette er vi nødt til å endre litt i innstillingene til prosjektet du opprettet. I mappetreet til høyre ser du øverst en «Solution», og under det navnet på prosjektet ditt (f.eks. «HelloWorld», står med fet skrift), som inneholder alle mappene og filene i prosjektet. Høyreklikk på prosjektet og velg **Properties**. I vinduet som kommer opp går du inn i **Configuration Properties** → **Linker** → **System**. Klikk nå på linjen der det står «SubSystem» og velg «Console» fra dropdown-menyen. Klikk **OK** for å lagre innstillingene.
8. Kjør programmet ditt på nytt ved å trykke `Ctrl+F5`. Hvis du får opp et spørsmål om prosjektet skal bygges på nytt, trykk **Yes**. Du ser nå at programvinduet blir værende, og du kan se teksten som ble skrevet ut. Gratulerer!

Oppgave 2.2 - Kompileringsfeil i Visual Studio

- Prøv å introdusere de samme feilene som du testet i Oppgave 1.3, og se hvor feilmeldingene blir listet ut.
- Prøv å dobbeltklikke på linjer i feilmeldingen og sjekk om Visual Studio merker linjene i koden din der den tror feilen ligger.

Oppgave 3 - Kompilering på UNIX (for interesserte)

UNIX og UNIX-lignende operativsystemer som Linux og OS X kjører på en stor andel av verdens servere og personlige datamaskiner. Blant annet har NTNU Linux-servere tilgjengelig for alle studenter. Vi skal nå prøve å logge på denne serveren, skrive og kompilere et program, og kjøre dette programmet. Programfilene du lager når du kompilerer på Windows vil selvsagt ikke kjøre på UNIX (og omvendt), men selve koden kan kompileres på UNIX slik at du får et tilsvarende UNIX-program.

1. Logg deg på NTNUs Linux-server for studenter ved hjelp av SSH. På Windows er den mest brukte SSH-klienten PuTTY[2]. Etter at du har åpnet PuTTY, koble deg til serveren `login.stud.ntnu.no`. Skriv deretter inn ditt vanlige NTNU-brukernavn og -passord, begge etterfulgt av `Enter`. Merk at det av sikkerhetshensyn ikke vil vises tekst når du skriver inn passordet ditt.
2. Du er nå logget inn på NTNUs studentserver. Dersom du er kjent med bruk av terminalen i Windows, fungerer denne på en lignende måte men med en del ulikheter.
3. Dersom du vil lære å bruke en UNIX-terminal, er det enklest å google noe lignende «unix terminal tutorial». Her er likevel en kort introduksjon til de viktigste kommandoene:
   - Når du nettopp har logget inn befinner du deg i hjemmemappen din, som på denne serveren er NTNU-hjemmeområdet ditt. Skriv kommandoen `ls` (**ls**t) for å se hvilke filer som ligger i mappen du befinner deg i.
   - Dersom du ikke ønsker å gjøre alt i hjemmemappen din, må du flytte deg til en annen mappe. Dette gjør du med kommandoen `cd` (**c**hange **d**irectory). Hvis du for eksempel har en mappe på hjemmeområdet ditt som heter «Dokumenter», kan du skrive `cd Dokumenter` for å gå inn i den. Du befinner deg nå i «Dokumenter»-mappen din. Skulle du ønske å gå videre til en undermappe, gjentar du `cd`-kommandoen, denne gangen med navnet til undermappen, for å gå videre dit.
   - Ønsker du å sjekke hvilken mappe du befinner deg i, kan du gjøre dette med kommandoen `pwd`. Dersom du skriver kommandoen `cd` uten noe etter på (det vil si uten noen *argumenter*) vil du bli returnert til hjemmemappen din. Det er også verdt å nevne at kommandoen `cd ..` forflytter deg til mappen *under* i mappetreet.
   - Dersom du ønsker å *opprette* en mappe, gjør du dette med kommandoen `mkdir`. Skriv for eksempel `mkdir kode` for å opprette mappen «kode» som en undermappe av mappen du befinner deg i.

[2] PuTTY kan lastes ned fra http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

4. Lag deg en ny fil ved navn `HelloWorld.cpp` på et egnet sted, for eksempel med teksteditoren `nano`. For å lage en ny fil ved hjelp av `nano` (eller redigere en eksisterende) skriver du

   ```
   nano filnavn
   ```

   Når du er kommet inn i `nano` bruker du hurtigtastene som vises på bunnen av skjermen. Tegnet `^` betyr «ctrl», så `Ctrl+O` (`^O`) utfører kommandoen «WriteOut», som er et annet ord for å lagre.

5. Alternativt: Hvis du i Oppgave 1 lagret `HelloWorld.cpp` på NTNU-hjemmeområdet ditt, kan du i stedet navigere deg fram til den ved hjelp av `cd` og gjenbruke den.

6. NTNU-serveren har to C++-kompilatorer installert: `clang++` og `g++`. Hvilken av disse du bruker er vilkårlig, men `clang++` er kjent for å gi mer forståelige feilmeldinger. Kompiler `HelloWorld.cpp` ved å skrive

   ```
   clang++ HelloWorld.cpp
   ```

   eller

   ```
   g++ HelloWorld.cpp
   ```

7. Sjekk hva programfilen som ble laget heter og start denne med kommandoen `./filnavn` (merk punktumet først i kommandoen).

8. Standard filnavn ved kompilering på UNIX er `a.out`. Du kan gi programmene du kompilerer andre filnavn ved å bruke argumentet `-o EgetFilnavn`, f.eks.

   ```
   g++ -o HelloWorld HelloWorld.cpp
   ```

9. Prøv også her å introdusere feil i koden og se på feilmeldingene du får når du kompilerer filen.