Forkurs i informatikk - Torsdag (Java)

Dagens formål:
- Få mer trening i bruk av terminalen og hvordan man kan bruke den til å se på andre typer filer; nemlig Java-filer! Vi skal også se på hvordan vi kan skrive flere instruksjoner etter hverandre - som blir et eget program!
- Kjøre/kompilere/endre programmer i Java.

Motivasjon: I dag sikter øvelsene på å gjøre overgangen til programmering utover i semesteret enklere ved at dere blir kjent med hvordan Java-filer ser ut og hvordan vi skriver enkle programmer.

Hjelp: dersom det er noen ord eller uttrykk du ikke forstår står det en del forklart her. Mye brukte kommandoer ligger her.

Ikke nøl med å spørre om hjelp underveis!

Programmeringsspråk og kildekode

Under dagens forelesning ble det snakket om hvordan alt er filer, og om hva kildekode er. Vi skal derfor i dag kikke på kildekode skrevet i programmeringsspråket Java, og lage enkle programmer selv.

Kildekode som er skrevet i Java-språket har filendelsen "*.java*", som forteller datamaskinen om hvordan denne filen skal tolkes.

Aller først - forsikre deg om at du har forkursmappen som ble lastet ned fra tirsdagen ved å navigere deg til den i Terminalen (hint: **ls** og **cd**). Hvis du ikke har den, se øvelser Dag 1 for hvordan dette skal gjøres. Når du er inne i mappen, bruk **tree**-programmet for å se hvilke filer som ligger der. Som du ser har du mange forskjellige filer:

Pythonfiler (.py)
Javafiler (.java)
Tekstfiler (.txt)

I dag skal vi titte nærmere på Java-filene. Åpne derfor kildekode-filen "*Dag2.java"* med **cat**-kommandoen for å se nærmere på den. Hva tror du Java-programmet gjør?

Fra kildekode til maskinkode, til kjøring

- også kalt "å kompilere"

Teksten, eller kildekoden, som ligger i Dag2.java er det man trenger i Java for å kunne skrive ut til tekst til Terminal-vinduet. Datamaskinen forstår likevel ikke det som står skrevet i filen - og vi trenger derfor å gjøre om teksten til en type kode som kun datamaskinen forstår, nemlig maskinkode. For å kunne gjøre det trenger vi et program til for Terminalen - **javac** (**java c**ompile). Dette programmet tar en kildekode-fil skrevet i Java som argument. Prosessen å gjøre om

kildekode-filer i Java til maskinforståelig kode kalles for kompilering, eller "*å kompilere*". Vi skal nå kompilere vår første fil, ved å sende med filnavnet til filen vi ønsker å kompilere til **javac**:

```
[bruker@maskin ~/MineFiler/Java] $ javac Dag2.java
```

Når kommandoen ovenfor er kjørt får vi en ny fil, som inneholder kode som kun datamaskinen forstår. Denne filen heter "*Dag2.class*". Vi skal ikke tenke på hva som ligger inne i denne filen, men hvis du ønsker kan du prøve å åpne den med **less**-programmet.

Nå som vi har kompilert java-filen vår, skal vi gå videre med å kjøre programmet vårt. Dette gjør vi med programmet **java**, som tar inn navnet på filen (uten filendelsen) som argument:

```
[bruker@maskin ~/MineFiler/Java] $ java Dag2
```

Nå har programmet (forhåpentligvis) kjørt ferdig. Gjorde programmet det du trodde?

Endringer i kildekode

Nå som vi har kompilert og kjørt programmet, skal vi nå gjøre endringer i selve *kildekoden*. Åpne derfor Atom/Emacs med en av følgende kommandoer:

```
[bruker@maskin ~/MineFiler/Java] $ atom &
```

ELLER

```
[bruker@maskin ~/MineFiler/Java] $ emacs &
```

Som du ser når vi åpner editoren, har vi fått fargelagt koden vår fordi den har en .java-filendelse. Fargene er for å kunne skille de forskjellige elementer i koden, slik at den blir mer lettlest for oss programmerere. Prøv nå å endre på programmet slik at den skriver ut "Hello world!" istedet. Når du har gjort endringene, husk å lagre. Kompiler deretter programmet på nytt med **javac** og kjør det med **java**. Sammenlign utskriften til Terminalen med forrige gang du kjørte programmet. Ble det riktig? Kan du få programmet til å skrive det ut to ganger?

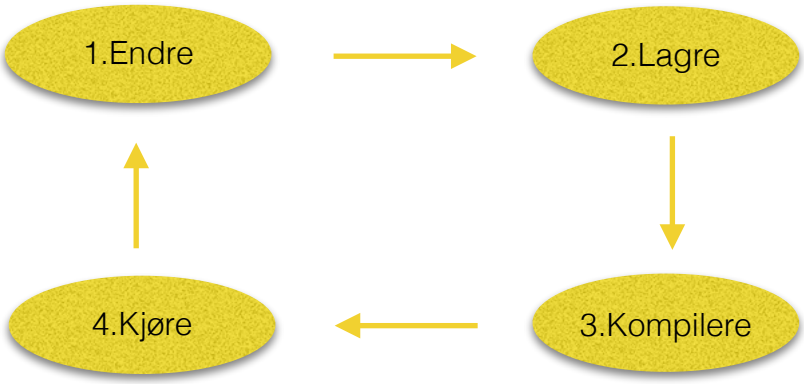
To ting som er fint å huske på når vi jobber med Java-programmer

- Husk å lagre hver gang man gjør endringer i filen
- Alltid husk å kompilere (med **javac**) før du kjører (med **java**)

Tekstvariablen - *String*

Som det ble snakket om på forelesning kan vi i dataprogrammene våre også ha variabler - lagringsplasser for ting som tekst og tall. Aller først skal vi se på et program som ligner på det fra sist øvelse, men som bruker variabler for lagring av tekst i utskriften. Åpne nå "*ProgOppgave2.java"* i Atom/Emacs, og se etter hva som er forskjellig fra forrige program. Hva tror du programmet skriver ut? Kompiler og kjør programmet for å se om det stemmer overens med det du trodde.

Endre nå programmet slik at String-variablen navn inneholder ditt eget navn. Test deretter at det fungerer ved å *lagre, kompilere og kjøre.*



Programmeringsprosessen i Java

Andre variabler og if

Åpne nå "*ProgOppgave3.java*" og se på koden. Her har vi tatt med en annen type variabel - int. Denne variabelen kan holde på ett heltall (dvs. *ikke* kommatall).

Kompiler og kjør programmet. Hva blir utskriften? Nå skal du endre på programmet slik at utskriften blir noe annet uten å endre på noe annet enn tallet. Kompiler og kjør. Ble det riktig?

Et eget program

Til denne øvelsen skal du lage et program selv basert på det du har lært i de tidligere øvelsene. For øvelsen er du gitt *skjelettet* for et Java-program. Skjelettet ligger i filen "*Skjelett.java*". Åpne denne i en editor og lag et program. Pass på å teste underveis.

Spesifikasjon av programmet:
- Lag en variabel for å ta vare på alder og legg inn en eksempel-alder
- Lag en variabel for å ta vare på et navn og legg inn et eksempel-navn
- Lag tester i programmet ditt som sjekker om en person er myndig eller ikke (dvs. er over 18 år gammel), og som skriver ut til terminalen:

Hvis han ikke er myndig: "Kåre er ikke myndig, fordi han er X år gammel"
Hvis han er myndig: "Kåre er myndig, han er jo X år gammel!"

Et program for morroskyld

Helt til slutt skal vi leke oss litt med det siste programmet som ligger i mappen - "*Morro.java*". Åpne derfor filen i editoren du liker best. Hvis du vil gjette - hva kommer til å bli skrevet ut til Terminalen?

Kompiler og kjør programmet. Hva skjer?

For deg som har lyst til å leke litt:

- Prøv å få programmet til å skrive ut alle tall mellom 0 og 1000000 (mill)
- Prøv å få programmet til å skrive ut alle partall mellom 0 og 1000000 (mill)
- Prøv å få programmet til å skrive ut hele nigangen (dvs. 9*1, 9*2, 9*3, 9*4). Et hint her er at du kan bruke gangetegn inne i System.out.println!

Oppgaver fullført

Bra jobbet, du er nå ferdig med alle forkursets øvelser! Husk at du ikke trenger å pugge kommandoene vi har vært borti i dag, og at dette mest er ment som øvelser som skal hjelpe deg på vei. Til neste uke er det fint om du husker på:

- Programmet **javac** som kompilerer kildekode i Java
- Programmet **java** som kjører Java-programmer

Synes du dette var gøy? Ta gjerne kontakt med et orakel på rommet ditt for litt mer avanserte øvelser. **Takk for at du har deltatt på forkurset - vi gleder oss til å programmere med deg utover i semesteret!**

PSST! Se opp for Fagutvalgets kurs og bli gjerne med oss! Mer info på fui.ifi.uio.no