

# INF-1100 Obligatorisk oppgave 1

## Innledning

Fokus på denne oppgaven er å vise at en har fått installert/bruke programvaren/verktøyene man trenger for å gjennomføre dette emnet.

## Oppgave1 Sette opp miljø og verktøy/programvare'

jeg har lastet ned linux mint sammen med vs code og github desktop, jeg har satte opp gcc kompilatoren

### Windows:

- Installer Visual Studio Code (eller annen IDE om man foretrekker et annet verktøy)
- Installer WSL 2 utvidelse
- Installer Windows Subsystem for Linux (velg Ubuntu om du er usikker på hvilken distro av linux du vil kjøre, og husk passordet du setter)
- Installere GCC (kompilator), GDB (debugger) og make (byggesystem)
  - o I linux-shellet (WSL) kjør kommandoen:  
*sudo apt install build-essential gdb*

To linker som kan være til hjelp:

[Installere WSL](#)

[Koble VSCode med WSL](#)

### MacOS:

- Installer Visual Studio Code (eller annen IDE om man foretrekker et annet verktør)
- Sjekk om Clang er installert, hvis ikke installer clang i terminalen med kommandoen:

*xcode-select -install*

Link som kan være til hjelp:

[Using Clang in Visual Studio Code](#)

### **Linux:**

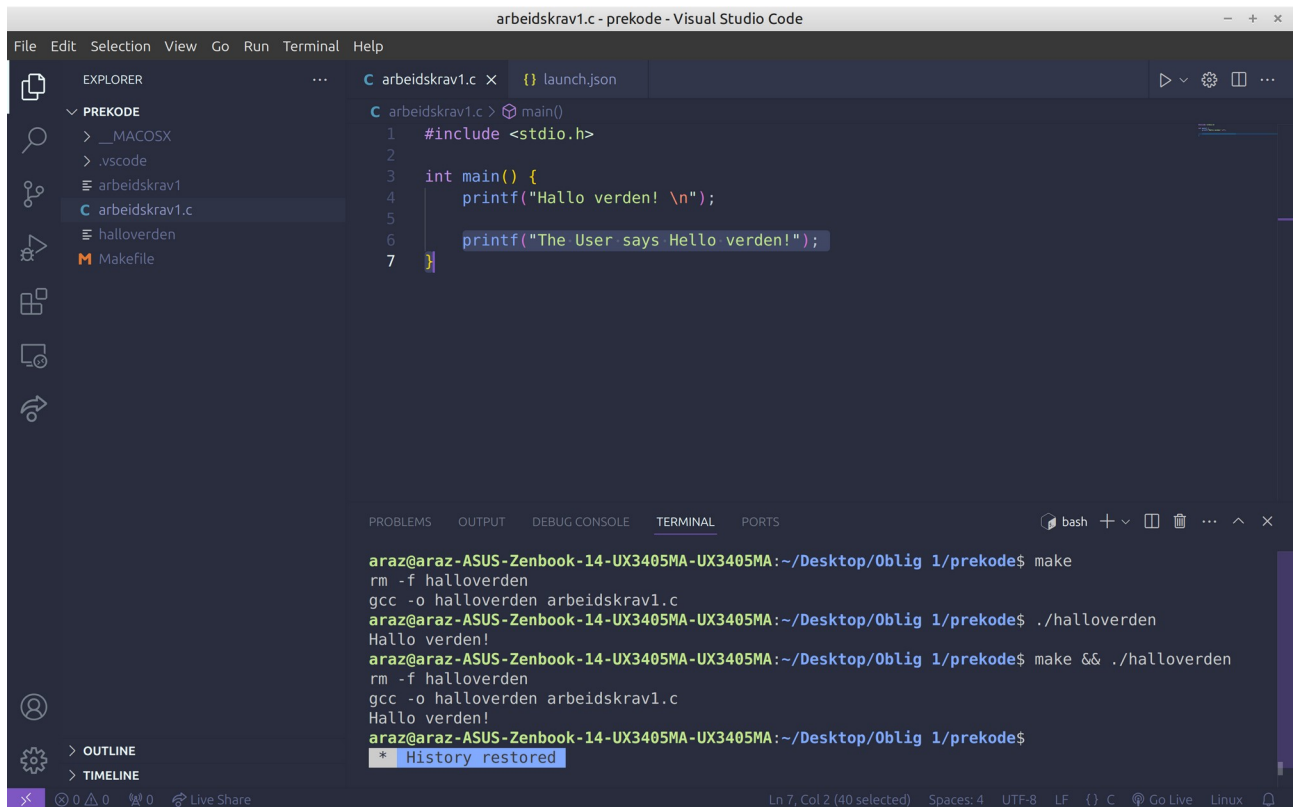
- Installer Visual Studio Code (eller annen IDE om man foretrekker et annet verktøy)
- Installere GCC (kompilator), GDB (debugger) og make (byggesystem)
  - o I terminalen kjør kommandoen:  
*sudo apt install build-essential gdb*

### **Oppgave 2 Kjøre make og programmet som make lager**

Last ned precode.zip, pakk ut innholdet, kompilér programmet med make og kjør programmet som er blitt laget.

```
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ make
rm -f halloverden
gcc -o halloverden arbeidskrav1.c
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ ./halloverden
Hallo verden!
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$
```

linux er satt opp på pcen sammen med visual studio code og GCC kompilatoren, I følgende skjermbilde får du se at make er runnet og genererer "rm -f halloverden" og "gcc -o halloverden arbeidskrav" dette skjønner jeg ikke hvorfor eller hva den gjør. Men jeg lærte også at hvis jeg bruker && mellom to kommandoer så kan jeg runne begge to!



The screenshot shows the Visual Studio Code interface with a C program named `arbeidskrav1.c` open in the editor. The program includes `<stdio.h>` and has a `main` function that prints two lines: "Hallo verden! \n" and "The User says Hello verden!". The Explorer sidebar on the left shows the project structure with files `__MACOSX`, `.vscode`, `arbeidskrav1`, `arbeidskrav1.c`, `halloverden`, and `Makefile`. The Terminal at the bottom shows the execution of the program using `make` and `./halloverden`, resulting in the output "Hallo verden!".

```
arbeidskrav1.c - prekode - Visual Studio Code
File Edit Selection View Go Run Terminal Help

C arbeidskrav1.c X {} launch.json

C arbeidskrav1.c > main()
1 #include <stdio.h>
2
3
4 int main() {
5     printf("Hallo verden! \n");
6     printf("The User says Hello verden!");
7 }
```

```
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ make
rm -f halloverden
gcc -o halloverden arbeidskrav1.c
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ ./halloverden
Hallo verden!
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ make && ./halloverden
rm -f halloverden
gcc -o halloverden arbeidskrav1.c
Hallo verden!
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$
* History restored
```

### Oppgave 3

Eksperimenter med å fjerne/legge til forskjellige deler på *halloverden* programmet og deretter kompilere.

Følgende er generellt kompilert

The screenshot shows the Visual Studio Code interface with a C file named `arbeidskrav1.c` open. The code is as follows:

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Hallo verden! \n");
5
6     printf("The User says Hello verden!");
7
8 }
9
```

The terminal at the bottom shows the following commands and output:

```
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ make && ./halloverden
rm -f halloverden
gcc -o halloverden arbeidskrav1.c
Hallo verden!
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$
* History restored
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ gcc arbeidskrav1.c
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ ./a.out
Hallo verden!
The User says Hello verden!araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$
```

her mangler det semi kolon, error koden forklarer akkurat det med «expected ';' before '}' så nevner den følgende linje og hvor feila er. Samt også akkurat hvor i setninga, og hvor i fila, på linje 6:42 som betyr linje 6 og kolonne 42

The screenshot shows the same Visual Studio Code interface, but now with a compilation error. The code is identical to the previous one. The terminal shows the following output:

```
Hallo verden!
The User says Hello verden!araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$ gcc arbeidskrav1.c
arbeidskrav1.c: In function 'main':
arbeidskrav1.c:6:42: error: expected ';' before '}' token
6 |     printf("The User says Hello verden!")
  |                                         ^
.....
9 | }
  | ~
araz@araz-ASUS-Zenbook-14-UX3405MA-UX3405MA:~/Desktop/Oblig 1/prekode$
```

## Oppgave 4

Det er 29 bokstaver i det norske alfabetet.

- a) Hvor mange bits trenger vi for å gi hver bokstav et unikt mønster?
  - a.  $5=2^5=32$  fordi man har 29 bokstaver og minimum bits blir da 32, og da har vi svaret vårt, vi treng 5 bits.
- b) Hvor mange bits trenger vi for å skille mellom stor og liten bokstav for alle bokstavene?
  - a.  $6=2^6=64$  fordi man treng  $29*2$  fordi at det er 29 vanlige bokstaver så store/småe liksom, da finner vi minimum tall for de.

## Oppgave 5

Beregne følgende. Skriv resultatet binært

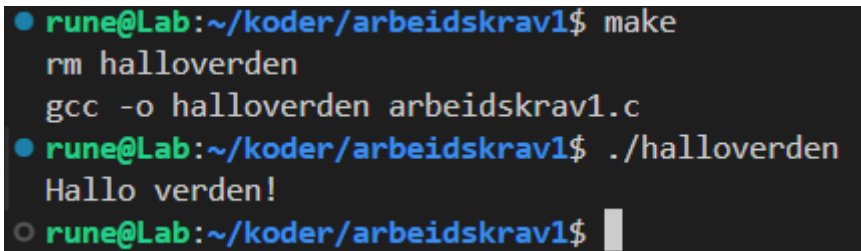
- a)  $00011001 \text{ AND } 10111100 =$ 
  - a. **00011000** dette er slikt fordi at jeg har tallen over hverandre i en slags måte at de står rett ovenfor hverandre, i denne måten er det satt opp riktig sted slik som b). det er slikt at AND betyr at , hvis det er 1 og 0 da får vi 0, mens om vi har 0 og 0 har vi 0, mens etter hvis vi har 0 og 1 får vi 0 mens om vi har 1 og 1 har vi 1 altså, det må være slikt at begge «switch» ene må være på men dersom «switch-en» er av og på er den kun av,
- b)  $00011001 \text{ OR } 10111100 =$ 
  - a. **10111101**, dette er slikt fordi at jeg har tallen over hverandre i en slags måte at de står rett ovenfor hverandre, i denne måten er det satt opp riktig sted slik som a). Her er det lignende som forrige, bare at OR betyr at den ene eller den andre må være på for at hele greia er «på».

Hva skal leveres inn:

For oppgave 1 og 2:

Et skjermbilde av at du har kjørt make filen og kjørt programmet som make lager.

Ala som skjermbildet under:

A terminal window with a dark background and light-colored text. The prompt is 'rune@Lab:~/koder/arbeidskrav1\$'. The user enters 'make', which outputs 'rm halloverden' and 'gcc -o halloverden arbeidskrav1.c'. Then the user enters './halloverden', which outputs 'Hallo verden!'. The prompt returns.

```
● rune@Lab:~/koder/arbeidskrav1$ make
rm halloverden
gcc -o halloverden arbeidskrav1.c
● rune@Lab:~/koder/arbeidskrav1$ ./halloverden
Hallo verden!
○ rune@Lab:~/koder/arbeidskrav1$
```

### For oppgave 3:

Beskriv hvilke typer feilbeskjeder du får. hva tror du de betyr?

### For Oppgave 4 og 5

Svar på a) og b) med en kort forklaring på hvorfor du har fått det svaret du har fått.