

Mmake  
Systemnära programmering 5DV088  
- 7,5hp

**Namn och email:**

Gustaf Söderlund - et14gsd@cs.umu.se

## Innehåll

1	Kompilering och körning	1
2	Användarhandledning	2
3	Diskussion och reflektion	3

# 1 Kompilering och körning

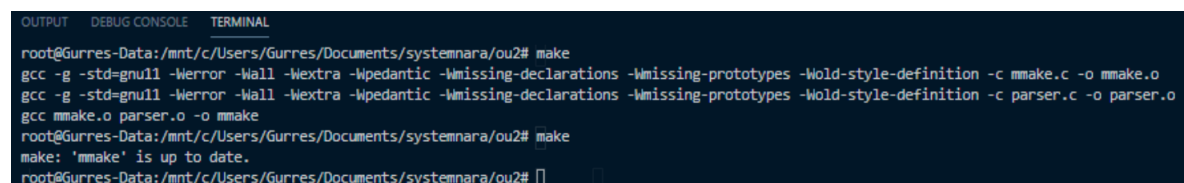
Programmet har som krav att kunna kompilera med hjälp av unix-verktyget make. För att kompilera programmet med verktyget make skapades en makefil. Den skapta makefilen innehåller de nödvändiga flaggor och filer som ska kompileras vid körning. I den här makefilen används följande filer mmake, mmake.o och parser.o. Makefilen för laborationen finns beskriven i figur 1.

A screenshot of a text editor showing a Makefile. The file is titled 'Makefile' in the top left corner. The content is as follows:

```
1 # Compiler
2 CC=gcc
3
4 # Compiler flags
5 CFLAGS= -g -std=gnu11 -Werror -Wall -Wextra -Wpedantic -Wmissing-declarations -Wmissing-prototypes -Wold-style-definition
6
7 # Force remove
8 RM=rm -f
9
10 # Main program linking
11 mmake: mmake.o parser.o
12 | $(CC) mmake.o parser.o -o mmake
13
14 # Separate file compilation
15 mmake.o: mmake.c parser.h
16 | $(CC) $(CFLAGS) -c mmake.c -o mmake.o
17
18 parser.o: parser.c parser.h
19 | $(CC) $(CFLAGS) -c parser.c -o parser.o
20
21
22 # Clean workspace from built files
23 clean:
24 | $(RM) mmake mmake.o parser.o
```

Figur 1: Representerar hur makefilen är skriven.

För att kompilera makefilen behöver filerna som är nödvändiga för programmet finnas i en och samma mapp på datorn. När filerna är samlad startar man en terminal och skriver kommandot make. När kommandot make är inskrivet ska flaggorna och filerna som angetts i makefilen kompileras vilket man kan se i figur 2. För att kontrollera om programmet har kompilerats skriver man kommandot make igen. Som man ser i figur 2 skrivs texten "make: 'mmake' is up to date." ut i terminalens fönster, där 'mmake' representerar den fil som är huvudfilen.

A screenshot of a terminal window with three tabs: 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL'. The 'TERMINAL' tab is active. The terminal shows the following commands and output:

```
root@Gurres-Data:/mnt/c/Users/Gurres/Documents/systemnara/ou2# make
gcc -g -std=gnu11 -Werror -Wall -Wextra -Wpedantic -Wmissing-declarations -Wmissing-prototypes -Wold-style-definition -c mmake.c -o mmake.o
gcc -g -std=gnu11 -Werror -Wall -Wextra -Wpedantic -Wmissing-declarations -Wmissing-prototypes -Wold-style-definition -c parser.c -o parser.o
gcc mmake.o parser.o -o mmake
root@Gurres-Data:/mnt/c/Users/Gurres/Documents/systemnara/ou2# make
make: 'mmake' is up to date.
root@Gurres-Data:/mnt/c/Users/Gurres/Documents/systemnara/ou2#
```

Figur 2: Representerar en lyckad körning med verktyget make.

## 2 Användarhandledning

För att använda programmet mmake behöver man skapa en makefil. Makefilen som programmet ska läsa in måste bestå av en eller flera regler som följer ett visst format.

- En regel består av två rader. På den första raden står target specificerat. Där target representerar huvudprogrammet som ska byggas. Det target som står längst upp i filen kallas default-target.
- På samma rad som target efter ett kolontecken står en lista av prerequisites som representerar objektfilerna som target behöver för att byggas.
- Efter att prerequisites har angetts går man vidare till nästa rad och gör ett "tab hopp". Här skrivs ett kommando som exekveras för att bygga target. Kommandot har samma namn som programmet som ska exekveras. Kommandot innehåller en lista av argument till programmet.

Första raden i filen får ej bestå av ett blanksteg. Där ska istället default target vara skrivet. Blanksteg används istället för att separera prerequisites och argument. Nyrad används för att avsluta listan med prerequisites och argument. Figur 3 beskriver hur formatet för en regel kan skrivas.

```
target : [PREREQUISITE...]  
        program [ARGUMENT...]
```

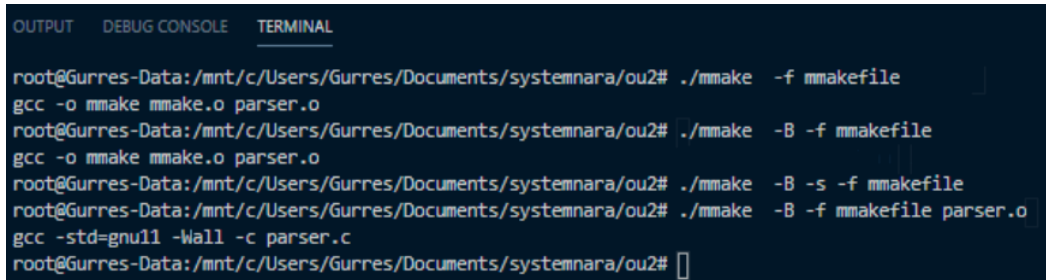
Figur 3: Filformat

I programmets finns det tre olika flaggor som programmet stödjer.

- Flaggan -f använder man för att ange en specifik makefile vid körning.
- Flaggan -B tvingar ombyggnation av makefilen oavsett om filerna är uppdaterad eller inte.
- Flaggan -S skriver inte ut någon output till terminalen vid körning.

När makefilen är skapad är programmet redo för att köras. De nödvändiga filerna läggs sedan i en gemensam mapp på datorn. Starta terminalen och navigera till rätt mapp, kompilera programmet enligt instruktionerna som finns beskriven under rubriken kompilering och körning. Använd sedan programmets följande format i terminalen: `./mmake [-f MAKEFILE] [-B] [-s] [TARGET]`

I figur 4 visas det hur de olika flaggorna används tillsammans med det givna format som tidigare var beskrivet.

A terminal window with a dark background and light text. At the top, there are three tabs: 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL', with 'TERMINAL' being the active tab. The terminal shows a series of commands entered at a prompt 'root@Gurres-Data: /mnt/c/Users/Gurres/Documents/systemnara/ou2#'. The commands are: 1. './mmake -f mmakefile' 2. 'gcc -o mmake mmake.o parser.o' 3. './mmake -B -f mmakefile' 4. 'gcc -o mmake mmake.o parser.o' 5. './mmake -B -s -f mmakefile' 6. './mmake -B -f mmakefile parser.o' 7. 'gcc -std=gnu11 -Wall -c parser.c' 8. The prompt is shown again at the end.

```
root@Gurres-Data: /mnt/c/Users/Gurres/Documents/systemnara/ou2# ./mmake -f mmakefile
gcc -o mmake mmake.o parser.o
root@Gurres-Data: /mnt/c/Users/Gurres/Documents/systemnara/ou2# ./mmake -B -f mmakefile
gcc -o mmake mmake.o parser.o
root@Gurres-Data: /mnt/c/Users/Gurres/Documents/systemnara/ou2# ./mmake -B -s -f mmakefile
root@Gurres-Data: /mnt/c/Users/Gurres/Documents/systemnara/ou2# ./mmake -B -f mmakefile parser.o
gcc -std=gnu11 -Wall -c parser.c
root@Gurres-Data: /mnt/c/Users/Gurres/Documents/systemnara/ou2#
```

Figur 4: Exempel på körningar med olika flaggorna i terminal.

### 3 Diskussion och reflektion

Det är andra gången jag läser denna kurs och de problem jag stött på under detta år med labben var att rätta mina misstag från förra året med test 11(som nu är grönt tillskillnad från förra året). Jag hade lite problem med att förstå hur man traverserade makefilen i rätt ordning. Jag kände också att det är svårare jobba med funktioner rekursivt. Då jag ofta tappar bort mig själv. Uppgiften var annars intressant och jag känner att jag lärt mig en del efter att ha gjort den.