

# DAT320

# Operating Systems

Fall 2014

Introduction to C



---

University of  
Stavanger



University of  
Stavanger

# Today

- History of C
- Hello World
- Toolchain



University of  
Stavanger

# History

- Initially developed by Dennis Ritchie between 1969 and 1973 at AT&T Bell Labs
- Closely related to development of UNIX
- Inspired by ALGOL



University of  
Stavanger

# History

- C was derived from B
- B from BCPL
- Rumors say that the name B comes from the name of the inventors wife, Barbara



University of  
Stavanger

# First program

```
int main( void )  
{  
    return 0;  
}
```

```
$gcc first.c -o first
```

```
$ ./first
```

```
$
```



University of  
Stavanger

# First program

- Every full C program begins inside a function called "main"



University of  
Stavanger

# Parts of a C-program

```
int main( void )  
{  
    return 0;  
}
```



University of  
Stavanger

# Hello world

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    /* The hello world program*/
```

```
    printf("hello, world\n");
```

```
}
```

```
mortenm@badne7:~$ gcc hello.c -o hello
```

```
mortenm@badne7:~$ ./hello
```

```
hello, world
```





- The first line of the program `#include <stdio.h>` is a preprocessor command, which tells a C compiler to include `stdio.h` file before going to actual compilation.
- The next line `int main(void)` is the main function where program execution begins.
- The next line `/*...*/` will be ignored by the compiler and it has been put to add additional comments in the program. So such lines are called comments in the program.
- The next line `printf(...)` is another function available in C which causes the message "Hello,World!" to be displayed on the screen.
- The next line **`return 0;`** terminates `main()` function and returns the value 0 to the OS.

# The preprocessor, #

- Executed **before** compiler
- Include other files:  
`#include<...>, #include "..."`
- Simple macros:  
`#define BUFFER_SIZE 100`
- Simple control structures:  
`#if`  
`#endif`



University of  
Stavanger

# Functions in C

```
return_type function_name( parameter list )  
{  
    body of the function  
}
```



University of  
Stavanger

# Functions in C

```
#include <stdio.h>
int my_add(int, int);

int main(void)
{
    int sum;
    sum = my_add( 3,5);
    printf("hello, world\n");
    printf("%d + %d = %d\n", 3,5, sum );
}

int my_add( int a, int b)
{
    return a+b;
}
```



University of  
Stavanger

# Guide

[http://www.tutorialspoint.com/cprogramming/c\\_quick\\_guide.htm](http://www.tutorialspoint.com/cprogramming/c_quick_guide.htm)



University of  
Stavanger

# Make

*“Tool to help automate build of software”*

target: dependencies

[tab] system command



University of  
Stavanger

# Make

*“Tool to help automate build of software”*

target: dependencies

[tab] system command



University of  
Stavanger

# Make

**Makefile:**

```
hello: hello.c
    gcc hello.c -o hello
```

```
mortenm@badne7:~/OpSys2014$ make
gcc hello.c -o hello
mortenm@badne7:~/OpSys2014$
.
.
mortenm@badne7:~/OpSys2014$ make
make: `hello' is up to date.
mortenm@badne7:~/OpSys2014$
```



# Behind the scene of gcc.

gcc is actually a collection of tools:

- powerpc-linux-gnu-addr2line.exe

- powerpc-linux-gnu-ar.exe

- powerpc-linux-gnu-as.exe

- powerpc-linux-gnu-c++.exe

- powerpc-linux-gnu-c++filt.exe

- powerpc-linux-gnu-cpp.exe

- powerpc-linux-gnu-g++.exe

- powerpc-linux-gnu-gcc4.3.2.exe

- powerpc-linux-gnu-gcc.exe

- powerpc-linux-gnu-gcov.exe

- powerpc-linux-gnu-gdb.exe

- powerpc-linux-gnu-gprof.exe

- powerpc-linux-gnu-ld.exe

- powerpc-linux-gnu-nm.exe

- powerpc-linux-gnu-objcopy.exe

- powerpc-linux-gnu-objdump.exe

- powerpc-linux-gnu-ranlib.exe

- powerpc-linux-gnu-readelf.exe

- powerpc-linux-gnu-size.exe

- powerpc-linux-gnu-strings.exe

- powerpc-linux-gnu-strip.exe



University of  
Stavanger

