Hello World!

PHYS2G03

McMaster University

A basic program: Say hello!

```
[wadsley@phys-ugrad ~]$ printf 'hello world!\n'
hello world!
```

printf is a very simple program available in unix – it just prints whatever text you give it back to the terminal

Note: I had to use the apostrophes '*hello world|n*' to make the \n newline character work

A basic program: Say hello!

```
[wadsley@phys-ugrad ~]$ which printf
/usr/bin/printf
[wadsley@phys-ugrad ~]$ ls -l /usr/bin/printf
-rwxr-xr-x 1 root root 49768 Aug 20  2019 /usr/bin/printf*
[wadsley@phys-ugrad ~]$ cat /usr/bin/printf
ELF>?@@?@8     @@@@@@?88@8@@@????
    ?????`??`?h??`?TT@T@DDP?td??@?@??Q?tdR?td????`??`XX/li
    b64/ld-linux-x86-64.so.2GNU
    GNU??<?`j???@_L?d?aXzSt?8z????H??R0??&@
```

printf lives in /usr/bin/printf and is about 50kB.

If you look inside you see garbage characters. This is binary machine code that runs directly on the CPU.

printf is a compiled program

A basic program: Say hello!

Today we are going to create a compiled program that does something very similar:

the classic *hello world program*

Compilers takes human readable *source code* in normal text (in a language like C++) and convert them into *machine code* that the CPU can understand and run

The C++ Compiler: c++

By default a compiler produces a runnable program in one step:

c++ myprog.cpp –o myprog

myprog.cpp is a source text file with c++ code in it
myprog is a new runnable program it just made
-o chooses the next word as the program name

If you forget –o the default program name is *a.out*

The C++ Compiler: c++

By default a compiler produces a runnable program in one step:

Source code in

c++ myprog.cpp –o myprog

machine code out

myprog.cpp is a source text file with c++ code in it
myprog is a new runnable program it just made
-o chooses the next word as the program name

If you forget –o the default program name is *a.out*

hello.cpp
A basic C++ source file

Open ▾ | hello.cpp ~/hello | Save

```
1 #include <iostream>
2
3 int main()
4 {
5   std::cout << "Hello World!\n";
6 }
```

C++ ▾ Tab Width: 4 ▾ Ln 6, Col 2 ▾ INS

gedit editor window

hello.cpp
A basic C++ source file

Open ▾ | hello.cpp ~/hello | Save

```
1 #include <iostream>
2
3 int main()
4 {
5   std::cout << "Hello World!\n";
6 }
```

C++ ▾ Tab Width: 4 ▾ Ln 6, Col 2 ▾ INS

gedit editor window

Create the hello program

Log on to phys-ugrad

Make an hello directory

cd to it

Start an editor:

Type in the program

Save it

(Don't close gedit!)

Note: it has to be exactly like the previous page to work!

```
$ mkdir hello
$ cd hello
$ gedit hello.cpp &
```

$ is your prompt, something like:
[wadsley@phys-ugrad ~/hello]$
Do not type the $

Create the hello program

Look at it with more

And ls –l

It is just human readable text and only 69 bytes (characters) long

A computer CPU cannot run this

```
$ more hello.cpp
#include <iostream>

int main()
{
  std::cout << "Hello World!\n";
}
$ ls –l hello.cpp
-rw-r-xr-- 1 wadsley wadsley 69 Sep 14 23:00 hello.cpp*
```

Create the hello program

Try to

Compile the program

If it works… you will

have a new file: hello

If there is any typo it will not compile. Compilers are picky!

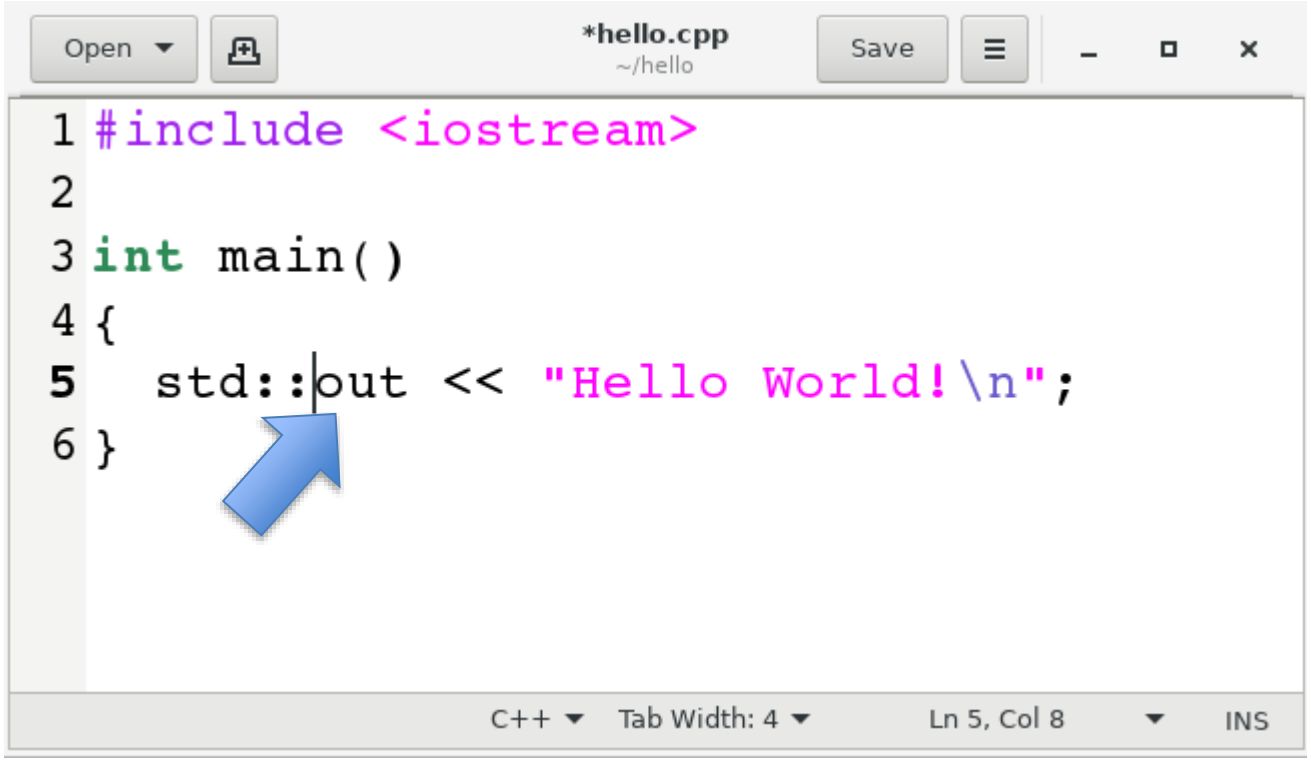
Fix it in gedit (or break it if you want)

```
$ c++ hello.cpp –o hello
$ hello
Hello world!

$ c++ hello.cpp -o hello
hello.cpp: In function 'int main()':
hello.cpp:5:3: error: 'out' is not a
    member of 'std'
  std::out << "Hello World!\n";
  ^
```

Typed in **std::out** not **std::cout** – arg!

hello.cpp with bug



Bugs are annoying. The compiler messages are tricky to interpret at first. You can fix your bug or…

Create the hello program

There is a correct copy of hello.cpp in

/home/2G03/hello

You can copy this over if you want instead

```
$ cp /home/2G03/hello/hello.cpp .
cp: overwrite './hello.cpp’? y
$ c++ hello.cpp –o hello
$ hello
Hello World!
```

the hello program

Look with ls –l

There is a new file, your program: hello

Hello is 8800 bytes of binary *machine code*

```
$ ls -l
-rw-r--r-- 1 wadsley wadsley   69 Sep 14 23:21 hello.cpp
-rwxrwxr-x 1 wadsley wadsley 8800 Sep 14 23:13 hello*
```

the hello program

Look with ls –l

Hello is executable so looks different with ls

```
[wadsley@phys-ugrad ~/hello]$ ls -l
-rw-r--r-- 1 wadsley wadsley   69 Sep 14 23:21 hello.cpp
-rwxrwxr-x 1 wadsley wadsley 8800 Sep 14 23:13 hello*
```

x means executable program

* also means executable program

The C++ Compiler: c++

Compilers do two jobs: **Compiling** and **Linking**

Compiling makes machine code
c++ hello.cpp –c Will just compile
That makes the file hello.o *.o* means *object file*
The .o files are not saved unless you specifically ask to compile only
This cannot be run as a program (not executable)

Linking is to put the machine code from hello.o into a program
The final program is hello *executable program*

Programs need extra code to talk to the OS and get any extra code they need at runtime. This is added automatically.

Your hello program does not actually contain absolutely all the code to print to the terminal. It asks to use shared printing code when it runs.
We will discuss that more under linking and libraries.

the hello program

hello.o is 2496 bytes of binary *machine code*

hello is your program: 8888 bytes of binary *machine code.*

This is actually pretty small because it uses a lot of shared code elsewhere when it runs

```
$  ls -l
-rw-r--r-- 1 wadsley wadsley   69 Sep 14 23:21 hello.cpp
-rw-rw-r-- 1 wadsley wadsley 2496 Sep 10  2019 hello.o
-rwxrwxr-x 1 wadsley wadsley 8800 Sep 14 23:13 hello*
```

hello actually running

```
ttop - 21:35:07 up 13 days, 11:07,  9 users,  load average: 0.45, 0.54, 0.24
Tasks: 209 total,   1 running, 207 sleeping,   1 stopped,   0 zombie
Cpu(s):  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   8030404k total,  7877680k used,   152724k free,   554044k buffers
Swap: 16795948k total,       16k used, 16795932k free,  6285524k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
15734 wadsley   15   0 70972 2152 1224 S  0.0  0.0   0:00.13 tcsh
27160 wadsley   21   0 13484  892  740 T  0.0  0.0   0:00.03 hello
```

Max memory use 13484000 bytes!

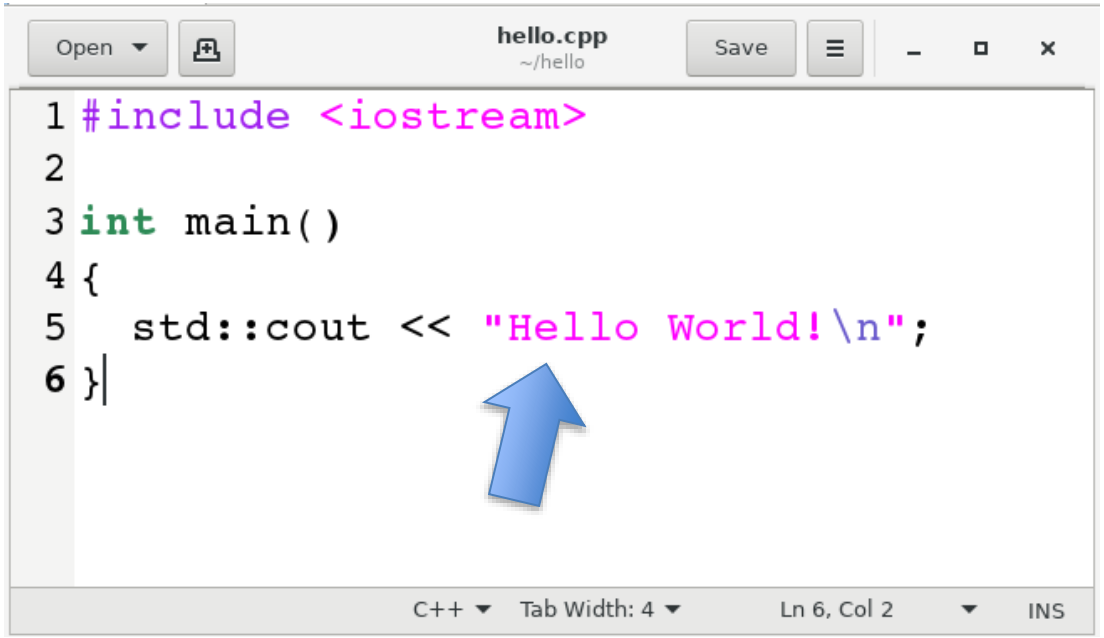
Current memory

Mostly shared code e.g. libraries

I used the **top** program – this is an interactive way to look at what's running on phys-ugrad

Play around with hello.cpp

You can change
the text
“Hello World!\n”
to say anything
you want
\n means new line



You can also change the program name, e.g.
c++ hello.cpp –o myprogram

gedit editor window

hello.cpp
A basic C++ source file

```
#include <iostream>


int main()
{
  std::cout << "Hello World!\n";
}
```

Anatomy of hello.cpp

```
#include <iostream>        header defines std::cout

int main()                 Main function
{
  std::cout << "Hello World!\n";    Output text to std output
                                    (the terminal)
}
```

The curly brackets (also called braces) { } enclose the source code that belongs to something – in this case the Main program

Anatomy of a source file with a main program C++ (or C)

```
#include <header>
```

Header tells compiler where to find things not explicitly defined here

```
int main(arguments)
```

Main function → every program needs one

Start here when program runs

```
{
   actual code;
   more code;
   …

   return 0;
}
```

actually do something

return to the operating system

Anatomy of a source file with a main program C++ (or C)

```
#include <header>  Header tells compiler where to find
                   things not explicitly defined here
int main(arguments)      Main function
                         Start here when program runs
{
   actual code;      actually do something
   more code;

   return 0;         return to the OS
}
```

Why does it look like this?

C was designed to write Unix commands like ls

The idea is that Unix

- Starts the program at the main function
- gives it the command line arguments
  e.g. ls file1.pdf the first argument is “file1.pdf”
- the program runs
- returns 0 if successful and some other integer if it failed