

C Coding Tools (Development Environment)

Dr. Hyun Lim

h.lim@lboro.ac.uk

*Institute for Digital Technologies
Loughborough University London*

- I suppose you don't ask the lecturer how to use your coding tools after today, e.g. *Visual Studio*, especially during the lectures. Because it is not in the main scope of this module and we need to study and discuss much more important and valuable topics, such as pointers, data structures.

- Refer to Resources on *LEARN*

The screenshot displays the LEARN system interface for the 20LLP109 - Digital Application Development module. The interface is divided into several sections:

- Navigation:** Includes links for My Modules, Latest News, Profile, Browse Modules, and Browse by Programme. There is a search bar for modules.
- Administration:** Includes links for Module administration and Activity results.
- Reading List:** Lists recommended books and chapters, including "Learn Python programming" by Romano, Fabrizio (2018) and "Python programming: an introduction to computer science" by Zelle, John M. (2017).
- Module Code:** Displays the module code 20LLP109.
- Module Noticeboard:** Contains three main sections: General Information (with a photo of a bookshelf), Lecture Notes (with a photo of a hand holding a book), and Coursework (with a photo of a hand holding a book).
- Tips:** A section titled "3. Find x" with a diagram of a right-angled triangle and a circle.
- Resources:** A section with a photo of a bookshelf and the word "Resources" in large letters.

A blue arrow points from the "Resources" section to the "Resources" text in the list item above.

Installation of C Compilers (free)

- Visual Studio (standard, professional):

<https://visualstudio.microsoft.com/downloads/>

- NewbieIDE (Windows, very simple and easy)

Download from *Resources* on *LEARN*

[[Installation version](#)] or [[Portable \(non-installation\) version](#)]

- Code::blocks (Windows, currently not for Mac OS)

Download [codeblocks-17.12mingw-setup](#) or

[codeblocks-17.12mingw-nosetup](#) at

<http://www.codeblocks.org/downloads/26>

- Online C compilers, such as

➤ <https://www.programiz.com/c-programming/online-compiler/>

- Refer to [*Resources*] on *LEARN*

main function

start/ end

- Any C program must have a *main()* function.

```
int main()
```

```
{
```

```
.....
```

```
return 0;
```

```
}
```

// **New** standard: C99 (ISO)

```
void main()
```

```
{
```

```
.....
```

```
}
```

// Old standard: C89 (ANSI-C)

"Hello World!" in C



input/ output

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

```
int main( )
```

```
{
```

```
    printf( "Hello World!" );
```

```
    getchar();
```

```
}
```

```
// C99 (ISO)
```

You don't need to type:

//.....

Commenting

Comment / uncomment the line(s)

// (the current line)

/* (line 1)
(line 2)

.....

(many lines)

***/**


Statement and Compound Statement (Block)

- Statement
 - Expressions can be made into a statement by a suffixing semicolon, e.g.
- Compound Statement (Block)
 - The compound statement is enclosed within braces { } and treated as a single entity, e.g.

```
y=x+1;
```

```
{  
    int x=1, y;  
    y=10+x;  
}
```

Why we use header files in C



```
#include <stdio.h>

int main( )
{
    printf( "Hello World!" );
}
```

You might be wondering why you need to `#include xxxx.h` files and why you would want to have multiple `.c` files for a program.

(1) It **speeds up compile** time.

(2) It keeps your code more **organised**:

If you separate concepts into specific files, it's easier to find the code you are looking for when you want to make modifications.

(3) It allows you **to separate interface from implementation**.

Difference between using angle brackets < > and quotes " "

```
#include <filename>
#include "filename"
```

- For `#include <filename>` the preprocessor searches in a directory pre-designated by the compiler, to include **standard library header** files.
- For `#include "filename"` the preprocessor searches for the included file **in the same directory**, normally to include header files that you defined.
- Set the PATH of a file: e.g., `#define PATH "C:\\aaa\\my_header.h"`

Exercise

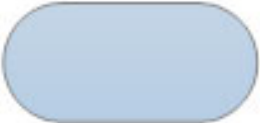

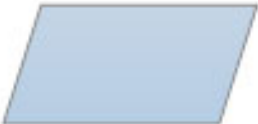
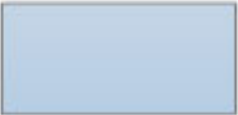

printf in C

```
#include <stdio.h>

int main()
{
    printf("Book information.\n");
    // \n means "go to the next line"
    printf("\nNo.[1] \n");
    printf("Author: John \n");
    printf("Title: C Programming \n");
    printf("Page number: 189");
    return 0;
}
```

- **C** > empty project (for Visual Studio) > .txt saved as **.c**
> include `stdio.h` (for *printf*) and write your codes

Flowchart Symbols

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision