${\bf C}$ Basics - Hello world program

CSC100 Introduction to programming in C/C++ Spring 2023

Marcus Birkenkrahe

Time-stamp: <2023-01-27 Fri 17:24>

Contents

1	README	2
2	Program structure	2
3	"What a Tangled Web We Weave"	3
4	Hello World Version 1 (medium)	4
5	Hello World Version 2 (short)	4
6	Hello World Version 3 (long)	5
7	Compiler workflow	6
8	Shell execution	6
9	Syntax highlighting in Emacs	7
10	Comments	8
11	Let's practice!	9
12	Summary	10
13	Code summary	10
14	Glossary	10
15	References	11

1 README

- This script summarizes and adds to the treatment by King (2008), chapter 2, C Fundamentals see also slides (GDrive).
- There is a separate Org-mode file available for practice. Download 2_hello_practice.org from GitHub: tinyurl.com/yckuhh2f
- Open a command line terminal and change (cd) to the Downloads directory
- Open the file in Emacs with emacs -nw --file 2_hello_practice.org
- When you leave class without having completed the file, save a copy to GDrive as a backup and/or to work on it from home
- When you've completed the file, upload it to Canvas where you'll find a "Class practice" assignment named 2_hello_practice.org.

2 Program structure

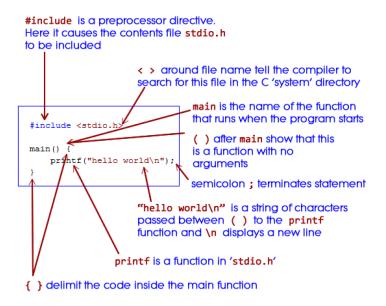


Figure 1: main function structure (Collingbourne, 2017)

- All C program statements must be included in a main function
- The main function has a body delimited by {...}
- There can be pre-processor directives #include or #define.

3 "What a Tangled Web We Weave..."

"Oh, what a tangled web we weave, when first we practice to deceive!" (Sir Walter Scott, 1808)

In this section, we're once again running code blocks from within Orgmode - with a few new *literate programming* features:

- To distinguish (and reference) code blocks, we will name them (#+NAME:). The name can can then be referenced anywhere
- To turn the code block into a source code C file (.c), we will add a :tangle FILENAME statement to the header
- To create the tangled (source code) file from a block, use the keys C-c
 C-v t (org-babel-tangle) 1
- To create the tangled (source code) from a file (all blocks), use the keys C-c C-v f (org-babel-tangle-file)
- Since source code files should have comments, we add the header argument :comments both: now, the most recent org block is used as a comment
- The workflow of "tangling" and "weaving" looks like this: Learn more about extracting source code from Org files.

¹To tangle only the currently selected block, use org-babel-tangle with a prefix argument: C-u C-c C-v t or C-u M-x org-bable-tangle.

²In our case, instead of weaving TEX files (.tex) to print, we weave Markdown files (.md), or WORD (*.odt) files, or we dispense with the weaving altogether because Orgmode files (equivalent of the *.w or "web" files) look fine on GitHub. GitHub.

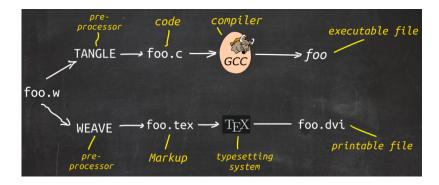


Figure 2: A file is woven into a document or tangled into a source file

4 Hello World Version 1 (medium)

```
#include <stdio.h>
int main(void)
{
   printf("Hello world\n");
   return 0;
}
```

Hello world

What happens in this code block:

- A header file (stdio.h) is included for input/output
- A function (main) without arguments (void) is defined
- The function returns integer data (int)
- A string ("...") is printed out
- A new-line is added at the end (\n)
- If successful, the program returns the value 0

5 Hello World Version 2 (short)

The program could also have been written much simpler:

• In this code block, the function main is missing the void argument, and the int (indicating the type of variable returned - an integer).

```
#include <stdio.h>
main()
{
   printf("Hello world\n");
}
```

Hello world

- It is the job of the *compiler*, gcc, which acts behind the scenes as it were, to resolve issues like "missing int" or "missing return"
- If you *tangle* the code block and compile the source file hello2.c in a shell, you get a warning:

```
$ gcc hello2.c
hello2.c:2:1: warning: return type defaults to 'int' [-Wimplicit-int]
main()
```

6 Hello World Version 3 (long)

The program could also have been written more complicated:

- int argc is an integer, or single number the number of arguments that were passed to main
- char **argv (or char *argv[]) is a *pointer* that refers to an *array* of characters a more complicated data structure

```
#include <stdio.h>
int main(int argc, char **argv)
{
   printf("hello world\n");
   return 0;
}
hello world
```

7 Compiler workflow

The machine cannot process a C source file like hello.c without help. It must:

Preprocess	The preprocessor acts on lines beginning with #
Compile	The compiler translates instructions into object code
\overline{Link}	The linker combines object code and functions like printf()
Run	The final *.exe program is a binary (machine) program
\overline{Debug}	The debugger controls rule violations along the way

I compiled the hello.c program on a Linux box - the executable is called hello.out. The other binary is hello.exe compiled on Windows. Compare the two executables - what do you notice?

```
-rwxrwxrwx 1 marcus marcus 48432 Dec 29 08:38 hello.exe
-rwxrwxrwx 1 marcus marcus 16696 Dec 29 12:28 hello.out
```

Question: are these executables portable?³

8 Shell execution

- You can also save the code in a C source code file hello.c
- Instead of Emacs, you could use notepad on Windows or nano on Linux
- You can compile the source files on the command line terminal or in the Emacs shell. Here is the workflow:
- The eshell is an Emacs Lisp simulation of a Linux shell (bash)
- On Windows, PowerShell works as well as the CMD shell:

³Executables are the result of compilation for a specific computer architecture and OS. The .exe program was compiled for Windows, the .out program was compiled for Linux. They will only run on these OS.

COMMAND	ACTION
C-x C-f hello.c	Create C file hello.c
	Copy block or write code anew in hello.c
C-x C-s	Save hello.c
M-x eshell	start a Linux shell in an Emacs buffer
gcc hello.c -o hello	compile program and create executable
ls -l hello*	list files - you should see hello, hello.c
/.hello	execute program

```
Windows PowerShell
PS C:\Users\birkenkrahe\Documents\GitHub\cc\src> gcc
PS C:\Users\birkenkrahe\Documents\GitHub\cc\src> ls
    Directory: C:\Users\birkenkrahe\Documents\GitHub\cc\src
                       LastWriteTime
                                                 Length Name
                                                  54062 a.exe
                1/27/2023
                              9:29 PM
               12/15/2022
12/15/2022
                              3:40 PM
                                                   8072 hello
                              3:40 PM
                                                    91 hello.c
                              9:17 PM
9:30 PM
9:19 PM
                1/27/2023
1/27/2023
1/27/2023
1/27/2023
                                                  83 hello1.c
54062 hello1.exe
                                                     62 hello2.c
                                                    556 hello3.c
                              9:17 PM
PS C:\Users\birkenkrahe\Documents\GitHub\cc\src> ./hello1
Hello world
PS C:\Users\birkenkrahe\Documents\GitHub\cc\src>
```

9 Syntax highlighting in Emacs

 Notice the slight syntax highlighting difference to an online REPL repl.it ⁴:

⁴replit.com is an online Read-Eval-Print-Loop (REPL) that looks like a Linux installation (in fact, it is a so-called Docker container, an emulated, customized Linux installation). When registering (for free) you can use many different programming languages - here is a link to my container.

```
#include <stdio.h>

int main(void) {

printf("Hello World\n");

return 0;

}
```

```
#include <stdio.h>
int main(void)
{
   printf("Hello world\n");
   return 0;
}
```

- There is no highlighting standard you should experiment with different themes⁵.
- Display line numbers with display-line-numbers-mode, and highlight lines with hl-line-mode ⁶ you can toggle these, and you can go through the minibuffer history with M-x M-p and M-n:

```
#include <stdio.h>

int main(void)

frintf("To C, or not to C: that is the question.\n");

return 0;

}
```

10 Comments

Forgetting to terminate a *comment* may cause the compiler to ignore part of your program - but both syntax highlighting and auto-indent in the editor will tip you off:

⁵You can find different themes for GNU Emacs here, and install them using M-x package-list-packages. To see the differences, enter M-x custom-themes and pick another theme now. You can save it automatically for future sessions.

⁶If you always want to have line numbers and highlight the line under the cursor, put these lines in your .emacs file:

```
printf("My "); /* forgot to close this comment ...
  printf("cat ");
  printf("has "); /* so it ends here */
printf("fleas");

My fleas
  Let's fix this:

printf("My "); /* forgot to close this comment */
printf("cat ");
printf("has "); /* so it ends here */
printf("fleas");
```

11 Let's practice!

My cat has fleas

Save the practice file as 1_hello_practice.org and complete it:

- 1. understand and change syntax highlighting
- 2. understanding and using comments in C



12 Summary

- C programs must be compiled and linked
- Programs consist of directives, functions, and statements
- C directives begin with a hash mark (#)
- C statements end with a semicolon (;)
- C functions begin and end with parentheses { and }
- C programs should be readable
- Input and output has to be formatted correctly

13 Code summary

CODE	EXPLANATION
#include	directive to include other programs
stdio.h	${\rm standard\ input/output\ header\ file}$
<pre>main(int argc, char **argv)</pre>	main function with two arguments
return	statement (successful completion)
void	empty argument - no value
printf	printing function
\n	escape character (new-line)
/* */ //	comments
main(void)	main function without argument

14 Glossary

CONCEPT	EXPLANATION
Compiler	translates source code to object code
Linker	translates object code to machine code
Syntax	language rules
Debugger	checks syntax
Directive	starts with #, one line only, no delimiter
Preprocessor	processes directives
Statement	command to be executed, e.g. return
$\operatorname{Delimiter}$	ends a statement (in C: semicolon - ;)
Function	a rule to compute something with arguments

15 References

- Collingbourne (2019). The Little Book of C (Rev. 1.2). Dark Neon.
- King (2008). C Programming A Modern Approach. Norton. Online: knking.com.