Table of Contents

1 Practice: C Programming Toolchain

- This directory contains practice material for the course.
- This practice session deals with your C programming toolchain. Since we're working on Linux, you have many small, powerful UNIX commands available to you.
- What you'll learn:
  - Finding out where you are on the computer
  - Climbing around the file system
  - Listing files
  - Downloading a file from the web
  - Compiling a simple C source file
  - Running an executable file
- Whenever you see this symbol: `* [ ]`, you have something to do.
- When you're in Emacs, put the cursor on the same line as the symbol and press `C-c C-c` then the symbol changes to `* [X]`
- `[ ]` Let's try this! Press `C-c C-c` a few times.

2 Finding out where you are on the computer

- To find out where you are, enter `pwd` ("print working directory") after the terminal prompt `$`. In the example below, the answer is `/home/pi`, which is the home directory of the user `pi`.

```
~$ pwd
/home/pi
```

- The answer to the shell command `pwd` is stored in the variable `$PWD`. To check this, enter the command `echo $PWD` at the prompt.

```
 ~$ echo $PWD
/home/pi
```

- You can actually see this via the symbol ~ before the prompt, which stands for another variable, `$HOME`.
- `[ ]`

  Print the variable `$HOME`.

```
echo $HOME
```

- The shell, or `bash(1)`, is a program that is capable of scripting, or programming both interactively (on the console/terminal/command line), or via stored scripts.

3 Climbing around the file system

- The computer has a list of locations stored where it checks for available programs - like an index - called the `$PATH`.
- `[ ]`

  Print the `$PATH` on your computer

  ```
  echo $PATH
  ```

- If it's not in the `PATH` list, the computer won't know it as a command. This includes executable files that you create yourself
- To change directory, use `cd`. E.g. `cd ~` or `cd $HOME` will get you to your home directory.
- `[ ]`

  Go to your home directory and print the working directory. You can execute both commands in one go whe putting a `;` between them.

  ```
  cd ~ ; pwd
  ```

  ```
  /home/pi
  ```

- `[ ]`

  Make another directory and go there (you could put all of these commands on one line separating them with semi-colons.

  ```
  mkdir -v test
  cd test
  pwd
  ```

  (The `-v` flag means "verbose" - say what you just did).

- To go to any place, you can enter `cd` followed by the location (or explicit path), e.g. `/home/pi` or `/home/pi/test`, or by a relative path using `..`
- For example `cd ..` brings you back home from `test`, while `cd .` does nothing, and `cd ../..` gets you one directory up from `pi`.
- `[ ]` Follow this path using `cd`:
  - Go back up one directory from `/test`
  - Go back up one directory from `/pi`
  - Go back up one directory from ~ (or `/home`)
  - Go back down to `/test`

4 Listing files

- The command `ls` lists all files in the current directory
- `ls` also works with file paths, e.g. this listing of the directory `$HOME`:

```
ls /home/pi
```

- `[ ]`

  This command has many useful options: try them yourself

COMMAND	LISTING	WHAT?
ls -l	long listing	file owners, size, time, permissions
ls -a	with hidden files	includes configuration files
ls -t	time-ordered	files sort by time of modification

- The example output for `ls -l` shows one file called `README.org` with permissions for the owner (`pi`) and his group (also called `pi`) of size 6832 byte, last saved on May 14 at 18:38.

```
-rw-r--r-- 1 pi pi 6382 May 14 18:38 README.org
```

5 Getting a source file from the web

- The C source file `hello.c` is stored online at this place in GitHub: tinyurl.com/mrxne2t3 (long URL)
- `[ ]`

  To download it to your `Downloads` directory, open a terminal and enter the following command:

```
wget tinyurl.com/crazylyons -O hello.c -o log
```

  - `wget` copies content from the web[1]
  - `-O hello.c` writes the content into a file `hello.c`
  - `-o log` writes messages into a file `log`[2]
- `[ ]`

  Check that you got the right content by entering `cat hello.c` at the command line. You should see this:

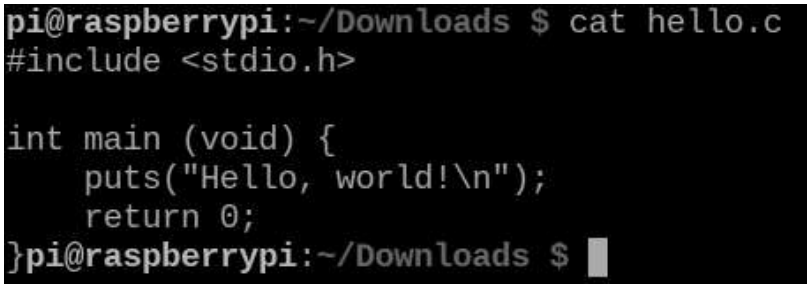

Figure 1: cat hello.c

- The `cat` command views a file. To find out more about this or any other command, you can type `man cat` at the command line. This is the UNIX help system of manual pages.
- `[ ]` Take a look at `log`.

6 Compiling a simple C source file

- You now have a fully formed C source file in your fingers.
- `[ ]`

  To compile the file, enter the following command on the command line - you shouldn't get any messages:

  ```
  cc hello.c -o hello
  ```

  This means "use `cc` to compile the file `hello.c` and put the result into `hello`".

- `[ ]`

  Now check the file listing with `ls -l`. You should see a new file, `hello`. Its name is highlighted, and you can see that it is 100 times larger and, most importantly, *executable*:

Figure 2: compile and list results

7 Running an executable file

- `[ ]` Run the executable file on the command line with the command `./hello`.
- `[ ]` To run the file, the computer needs a path to the file. Try entering the name of the file only.

- The path to the file can be explicit or implicit. Here is the explicit path solution:

Figure 3: run executable file (explicit path)

- The implicit path uses `./` to indicate the current directory:

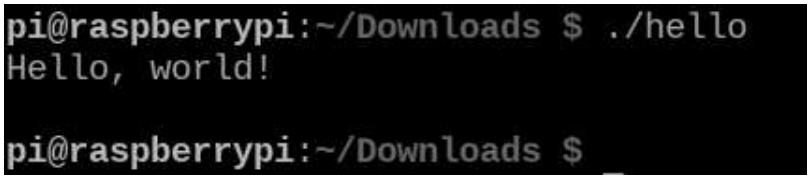

Figure 4: run executable file (implicit path)

8 Summary

- To find out where you are, use `pwd`
- Linux has environment variables like $PWD, $HOME, $PS1
- To move around use `cd` with explicit/implicit paths
- To list file information, use `ls` (with/out options)
- To get files from the web, use `wget` (or `curl`)
- To compile a C source code file, use `cc` (which is an alias for GCC, the GNU C Compiler Suite, or `c99` (a compiler standard)

9 Glossary

TERM	MEANING	WHAT?
UNIX	Operating system (OS)	Enables your computing
Linux	Operating system (OS)	Enables your computing
$PWD	Print working directory	Where you are in the file system
$HOME	Hhome directory (~)	Where your files are
Shell	Connection to the OS	E.g. `bash`, scripting language
Prompt	Command line symbol	Enter shell commands after it
echo	Displays its arguments	E.g. `echo hello` prints `hello`
variable	Memory location	Can be declared/re/defined
PATH	Environment variable	Where computer looks for pgms
mkdir	Make a new directory	E.g. `mkdir -v test`
ls	List files	E.g. `ls -l` for a long listing
wget	Download web content	Needs URL only
man	UNIX manual pages	E.g. `man wget`
log	Message file	Log download process
cat	Viewing command	E.g. `cat hello.c`
cc	C compiler (GCC)	Includes flags like `-o`

TERM	MEANING	WHAT?
compile	Make executable file	E.g. `cc hello.c`
executable	Binary (machine) file	Run e.g. with `./hello`

Footnotes:

[1] Notice that you did not need a browser to do this. `wget` is actually much smarter and faster than any browser download program. Its manual page (`man wget`) is highly readable.

[2] If we did not store the log in a file, it would simply be printed to the screen. An alternative is to redirect all messages to nowhere by replacing `-o log` with `&>/dev/null`.

Author: Marcus Birkenkrahe
Created: 2022-05-25 Wed 17:38