University of Reading
Department of Computer Science
Pat Parslow, Rhian Taylor, Neyma Siddiqi

Exercise Week 1
CS1PC20 / Autumn 2021
120 Minutes
Discussion date: Week 2

---

1. The exercises are formative – they are there to learn – but also assessed as part of your portfolio of work submitted for course works.
2. Exercises will be discussed on the Week indicated on the header. Additional (non-discussed) training tasks may be provided, and will be shown in red.
3. Make sure to schedule some time each week to keep up with the progress. It is not necessary to complete every task perfectly. If you are struggling, then reach out for help on Teams.

Exercise Introduction

Before attempting the exercises in this document please ensure that you have read and understood the key topics covered in Tutorial 1.

Remember: Tasks count towards your final grade and are also a tool to help you test your own knowledge. You should attempt them as simply learning the theory is not enough, practical application is an invaluable tool in helping you to learn.

Use this exercise to familiarize yourself with Linux and the Bash, there should be enough time to get started.

Contents

Task 1: Bash Exercises (120 min)

You may want to view the video of me going through the exercise commands before you do it – I was not recording the outcomes, so it is done more quickly than I would expect you to, so the video is only 16 minutes long. I left off the last 4 commands – see if you can work out what they do before you try them. You may get some additional hints about what the commands do from the video!

We don't expect for you to be writing long and complex bash scripts straight away. We also appreciate that moving to a CLI (Command Line Interface), while powerful, can be challenging especially when you are accustomed to interacting with primarily graphical interfaces on Operating Systems like Windows.

Below we have included several Bash commands of varying complexity that also cover more advanced concepts of the Bash. **We want you to experiment with them, try running them and observe the result! Think about what you expect their output to be before you run them.** Perhaps make your own changes to the commands. Use this as an opportunity to learn about the shell and increase your comfort in writing your own scripts.

We would like you to produce a markdown text file providing a short descriptions of your findings, i.e. all you need to provide is a brief description of what the command does (if possible write what you expected) and a short explanation of how it works where appropriate. Marks do not depend on you having the right 'prediction' – so avoid the temptation to run it and then record what happened as what you think will happen!

The first 7 commands in the list are experimenting with different ways of joining two commands together.

You do not need to type in the $ at the beginning of each line (in fact, you need to not type it in…)

1. $ mkdir -p $HOME/portfolio/week1 ; cd $HOME/portfolio/week1
2. $ cd ~
3. $ rm -r portfolio
4. $ mkdir -p $HOME/portfolio/week1 & cd $HOME/portfolio/week1
5. $ cd ~
6. $ rm -r portfolio
7. $ mkdir -p $HOME/portfolio/week1 && cd $HOME/portfolio/week1
8. $ echo "Hello World"
9. $ echo Hello, World
10. $ echo Hello, world; Foo bar
11. $ echo Hello, world!
12. $ echo "line one";echo "line two"
13. $ echo "Hello, world > readme"
14. $ echo "Hello, world" > readme
15. $ cat readme
16. $ example="Hello, World"
17. $ echo $example
18. $ echo '$example'
19. $ echo "$example"
20. $ echo "Please enter your name."; read example
21. $ echo "Hello $example"
22. $ three=1+1+1;echo $three
23. $ bc (hint: check what the prompt says … also, "quit" is your friend)
24. $ echo 1+1+1 | bc
25. $ let three=1+1+1;echo $three

```
$ echo date
$ cal
$ which cal
$ /bin/cal
$ $(which cal)
$ ‘which cal‘
$ echo "The date is $(date)"
$ seq 0 9
$ seq 0 9 | wc -l
$ seq 0 9 > sequence
$ wc -l < sequence
$ for I in $(seq 1 9) ; do echo $I ; done
$ (echo -n 0 ; for I in $(seq 1 9) ; do echo -n +$I ; done ; echo) | bc
$ echo -e ‘#include <stdio.h>\nint main(void) \n{\n  printf(“Hello World\\n”);\n  return 0;\n}’ > hello.c
$ cat hello.c
$ gcc hello.c -o hello
$ ./hello
```

Hints

- The pipe symbol |, allows to chain the execution of multiple programs; it feeds the output of the program left of the pipe as input into the program right of the pipe.
- Use the up/down arrows to browse the history of the shell to find recently used commands
- Use the left/right arrows (Pos1/End keys) to change the position of the text cursor
- Use the tab key to autocomplete commands when you’re halfway through typing them - it saves time!
- The middle mouse key copies previously marked text into the shell
- Bash has certain special characters such as ;!<> that can cause unexpected behaviour when included outside of quotes in a command.
- Use the commands
  $ help and
  $ man
  to find out what a specific command does. You should also try
  <command> --help

Portfolio (directory: portfolio/week1/introductory-bash.md)

portfolio/week1/introductory-bash.md

An enumerated list that provides for each command a prose sentence what the commands did. Where your expectation differs from observed behaviour, include a sentence describing what surprised you. See if you can explain what the commands did (briefly!)

Further Reading

- Shell scripting (also available as PDF) https://www.shellscript.sh/
- Shebang Wikipedia: https://en.wikipedia.org/wiki/Shebang_(Unix)
- Commands: https://maker.pro/linux/tutorial/basic-linux-commands-for-beginners
- Command line structure: https://www2.cs.duke.edu/csl/docs/unix_course/intro-14.html
- https://en.wikipedia.org/wiki/Filesystem_Hierarchy_Standard

Task 1: Bash Research (120 min)

This is a more difficult **optional** task which can be done instead of Task 1

We would like for you to produce a short (about 1 page, at least 300 words) report on the core concepts behind bash and shown them in practice. Your report should give brief explanations and examples of the following concepts:

- Special characters and escaping
- Redirection
- Piping
- Variables
- Arithmetic
- Common Commands that you find useful beyond our examples (ls, cd, mkdir, ...)

You should include examples from the alternative (easy) task.

In your report you should consider circumstances in which easily avoidable errors could be encountered and subsequently avoided in your commands (for example, unescaped special characters such as ; ! # &)

Portfolio (directory: portfolio/week1/introductory-bash.md)

portfolio/week1/introductory-bash.md

About 300 words describing the concepts relevant to Task1.

Further Reading

- https://learnxinyminutes.com/docs/bash/ - Usage examples of many common bash commands.
- https://www.gnu.org/software/bash/manual/html_node/Quoting.html - Nice guide to how quotes work in Bash
- https://jvns.ca/blog/2017/03/26/bash-quirks/ - Article going over basics and some common quirks in the language which can trip up beginners.
- https://www.geeksforgeeks.org/piping-in-unix-or-linux/ - Covers piping and its application.
- Advanced Bash-Scripting Guide: I/O Redirection: https://www.tldp.org/LDP/abs/html/io-redirection.html