**CS2106 Introduction to Operating Systems**

**Lab 2 - Shell Scripting and Process Programming**

**Answer Book**

Please read the instructions in the main lab sheet before completing this document. Submission deadline is **Sunday 20 February 2022, 11.59 pm**. The folder will stay open slightly after this, but once the folder closes, **absolutely no submissions will be allowed.**

**Submission checklist:** A ZIP file called AxxxxxxY.zip, where AxxxxxxY is the student ID of the student submitting. The ZIP file should contain:

* This file, appropriately renamed to the submitter’s student ID.
* grade.sh
* lab2p2f.c

|  |  |
| --- | --- |
| **Student 1** | |
| Name: |  |
| Student ID (AxxxxxxY): | A0200521E |
| Group (Bxx): | B19 |
| **Student 2** | |
| Name: |  |
| Student ID (AxxxxxxY): |  |
| Group (Bxx): |  |

**Part 1 – Bash Scripting**

**Question 1.1 (1 mark)**

The shebang, #!/bin/bash when used in scripts is used to instruct the operating system to use bash as a command interpreter.

**Question 1.2 (1 mark)**

Added the $ symbol ahead of the expression to make it a arithmetic evaluation

**Question 1.3 (1 mark)**

#!/bin/bash

echo "Hello $(whoami), today is $(date +%A), $(date +%d) $(date +%B) $(date +%Y), and the time is $(date +%T)"

**Question 1.4 (1 mark)**

$# - Number of parameters

$1 – The first parameter

$2 – The second parameter

$@ - All the input arguments

$- Process ID (PID) in bash

**Question 1.5 (1 mark)**

Exit(i) returns i to the shell which is stored into the variable ?

**Question 1.6 (1 mark)**

& Causes Bash to execute the command asynchronously

; Causes Bash to execute the command sequentially

(For grader only) Part 1 total: \_\_\_\_\_\_\_\_\_\_\_ / 6

**Part 2 – Playing with POSIX Calls**

**Question 2.1 (1 mark)**

Yes as the for loop in slow for the parent is executing at the same time as the child’s slow for loop, and the child’s slow function call is done after the parent’s slow function call.

**Question 2.2 (1 mark)**

The parent’s parent is the bash shell

**Question 2.3 (1 mark)**

ac is the number of arguments inclusive of the file be run

av is an array containing the command line arguments

vp contains information about the OS such as the environment is being run in, the home directory, the user, ssh connection if any

**Question 2.4 (1 mark)**

char \*args[] = {"cat", "file.txt", NULL};

execvp("cat", args);

stored the arguments of cat command into the array and pass the array as an argument to execvp.

**Question 2.5 (1 mark)**

dup2 makes two file descriptors equivalent.

We have to use dup2 as we used more primitive operations to open file.txt and talk.out

**Question 2.6 (1 mark)**

Closing the unused pipes helps avoid error while passing messages as the pipes are meant to be unidirectional communication channels.

**Question 2.7 (1 mark)**

First create a fork which executes slow 5 and dup2 the STDOUT to the write end of the pipe, then in the parent process reads from the read end of the pipe and is directed to the STDIN which is then read when talk is executed and finally we redirect the STDOUT which direct the output of the talk to the results.out file.

(For grader only)

Part 2 total: \_\_\_\_\_\_\_\_\_\_\_ / 6

**REPORT TOTAL: \_\_\_\_\_\_\_\_\_\_\_\_ / 13**