Scripting Languages: Assignment 2.1- Portfolio Task Brief

Overview

In this assignment you are required to write a script that demonstrates the extent to which you have understood the *shell script* concepts and practices addressed in Modules 1 to 4 inclusive. This assignment is worth **20%** (*20 marks*) of your unit grade and its completion will help you build skills required for the assignments to follow.

GENERAL REQUIREMENTS

- Your script will be marked on a standard Ubuntu Linux for desktop installation using the **bash** shell.
- Refrain from using *non-core* commands, tools and utilities in your bash shell scripts. Non-standard bash commands, tools and utilities will **not** be downloaded and installed by your tutor.
- Ensure each script you write is *fully self-contained* and is not configured to be dependent on external files, libraries or resources to run.
- Do **not** use the *trap* command in any of your scripts.
- All scripts you submit must contain your full name and student number at the top as comments.

IMPORTANT NOTES:

- This is an **individual** assignment only and must **not** be completed in collaboration with other students.
- You may **not** work with others, acquire code from others, or provide code to others.
- Further, you may **not** post any of the assignment's specifications to a code-development community of any kind seeking solutions or advice.
- **Nor** may you copy and paste, or otherwise reproduce, any shell script verbatim from sources external to the unit materials into your own scripts. You may examine (read) how external sources have approached particular shell scripting tasks with a view to generating ideas about how you might approach a particular task yourself as long as their code does **not** become your code.
- Where it is found that any of these restrictions have been ignored, academic misconduct proceedings may be initiated.
- Please read the checklist below and watch the associated video **BEFORE** submitting your assignment

ACADEMIC INTEGRITY TICK-BEFORE-SUBMIT CHECKLIST

PLAGIARISM

✓I have not copy and pasted from external sources without appropriate citation
✓My in-text and end-text citations follow APA 7 guidelines
✓I have not used my own or other student's previous assignment work

COLLUSION

✓I have not worked with any other students on this assignment unless permitted
✓My assignment is not based on or derived from the work of any other students
✓I have not shown or provided other student(s) with my assignment at any point

CONTRACT CHEATING

✓I have not asked or paid someone to do this assignment for me
✓I have not used any content from a "study notes" or "tutoring" service / website
✓I have not had a friend or family member assist me with this assignment

IF YOU ARE UNSURE ABOUT ANY OF THE ABOVE, DO NOT SUBMIT YOUR ASSIGNMENT BEFORE SPEAKING WITH YOUR UNIT COORDINATOR OR ECU LEARNING ADVISOR

Watch this video before submitting your assignment

Task – Get Prime Count and Sum (20 marks)

Write a script that:

- **Counts** all *prime numbers* that exist within a user-provided range, e.g. 100 to 200
- **Calculates** the *sum* of all the prime numbers identified within this range
- **Prints** the *prime numbers*, their *count*, and their *sum* to the terminal

Requirements:

- Call the script *getprimes.sh*.
- Your script will prompt the user for the *lower* and *upper* bounds of the range
- User inputs must be **fully** validated to ensure they meet the following range rules:
    1. Accepts **only** numbers, e.g. 100, and nothing else, i.e., rejects strings, nulls, etc
    2. Does **not** accept the numbers *0* and *1* as valid range values
    3. Ensures the *lower* range bound provided is less than the *upper* range bound provided
    4. Ensures that there is at least one number between the *lower* and *upper* bound values provided, e.g. a minimum viable range would be 100 to 102, but **not** 100 to 101
    5. Whenever the user's input does not pass validation, the user is to be advised of the applicable error and prompted to try again, i.e. the user is to be looped back to prompt; the program is not to terminate
- If no prime numbers exist within the stipulated range, then advise the user of this via the terminal, e.g. *no prime number(s) exist within the range x and y*
- In the event one or more prime numbers are found within the stipulated range, these are to be printed to the terminal along with their *count* and *sum* in a neat, easily readable format
- To construct your script, use any combination of commands, utilities, control structures etc covered through *Modules 1-4* inclusive.

Watch the **Get Prime Count and Sum** demonstration video that accompanies this assignment brief on Canvas to see this script in action. This will give you a good idea of what your own script needs to do.

Marking Key

Marks will be awarded in 1.0 increments as follows:
[*Not addressed or does not work at all* - 0.0] [*Partially works* - 1.0] [*Works exactly as stipulated* - 2.0]

ITEM#	ITEM	VALUE
1	User prompted for lower and upper bounds of the range	2
2	User input logic only accepts integer numbers, and not strings, nulls or floats	2
3	User input logic rejects numbers 0 and 1 as valid range values	2
4	User input logic ensures lower range bound less than the upper range bound	2
5	User input logic ensures at least one number between lower and upper bound values provided	2
6	When user input fails validation, user advised of applicable error and prompted to try again (script does not terminate)	2
7	Where no prime numbers exist within the stipulated range, user is advised of this via the terminal	2
8	Where prime numbers do exist within the stipulated range, *all* are included in the *count* displayed to terminal	2
9	Where prime numbers do exist within the stipulated range, *all* are *displayed* to terminal	2
10	Where prime numbers do exist within the stipulated range, *all* are factored into the correct *sum* which is displayed to terminal	2
	TOTAL	**/20**

How to submit your portfolio to Canvas

Submit the shell script (bash) file you have created in a **zipped folder** with the following naming format:

`[surname]_[student-ID]_CSP2101_PF1.zip`

END OF ASSIGNMENT BRIEF