BSc (Hons) Computing

Module Code: QH0305   Module Title: Problem Solving

Assessment Sheet 5

Instructions:

This is one of the eight assessment tasks, which will contribute to the overall marks. You will need to complete the tasks as outlined below and then document them in a Word document. As a minimum, you should provide screenshots of the following:

- Your code
- The output that your code generates

In instances where your code could generate different outputs (depending on what values it is given), you should provide multiple screenshots of the console screen, showing the different outputs to demonstrate that the code is working correctly.

This assessment sheet will focus on all the concepts you have learnt up until now, especially arrays.

You must attempt all tasks on this sheet to achieve a higher grade. For example, if you want to gain Grade A, you must complete all other grades first and add them to your portfolio with screenshots.

A zip folder with all Grade codes must be attached inside of the portfolio (MS Word file).

Task 5: Climate data analyser

In this task, you will create a program that helps analyse daily temperature readings for a given period. The program will utilise arrays and various operations to calculate statistics and provide insights into the temperature data.

- Prompt the user to enter the number of temperature readings (up to 30).
- Create two parallel arrays: one for storing the temperature readings and another for storing the corresponding temperature IDs (integers). Ensure that the arrays have enough space to accommodate the specified number of readings.
- Allow the user to input the temperature readings and the IDs one by one, storing them in the arrays.

The program will offer a menu with the following options:

- Display all temperature readings. List all entered readings in a user-friendly format.

- Calculate and display the average temperature.
- Calculate and display the highest temperature.
- Calculate and display the lowest temperature.
- Count the number of days above a certain temperature (e.g., 15°C).
- Count the number of days below a certain temperature (e.g., 5°C).
- Exit

Here is an example of how the program should behave:

```
"Please enter your choice from the following menu:

A - Enter temperature readings

B- Display all temperature readings

C - Calculate and display statistics

D - Display analysis results

X - Exit
```

Please note that the program is designed to eventually incorporate all of the following options. **However, for a D grade, it is sufficient to complete option A and option B only.**

Input temperature readings:

- User selects option A.
- User enters the number of temperature readings they want to input: 5.
- User enters the exam scores one by one: 34, 12, -3, 4, 17.
- Program acknowledges the readings and stores them.
- The program generates a unique ID for each reading and stores both the ID and the reading in parallel arrays.

Display all temperatures:

- User selects option B.
- Program displays all entered temperature readings are displayed with their corresponding IDs.

Calculate and display statistics:

- User selects option C.
- Program calculates and displays statistics: the average, highest, and lowest temperatures from the entered readings.

Display analysis results:

- User selects option D.

- Program displays additional analysis results, such as count days based on temperature thresholds: The program counts and displays the number of days with temperatures above and below specified thresholds.

Complete the above task and then document it in your portfolio.

To achieve a D grade

1. You need to provide screenshots of your code and the different outcomes it can give.

2. Your Word document should have appropriate headings to ensure that this task can easily be identified alongside the rest of your work.

3. The program should work as follows:
   - Get the user to input their menu choice via the keyboard.
   - If the user selects 'A', prompt the user to enter the number of temperature readings they want to input and then ask them to enter each temperature reading, storing them in an array. The program generates a unique ID for each reading and stores both the ID and the reading in parallel arrays.
   - If the user selects 'B', display all entered temperature readings with their corresponding IDs.

To achieve a C grade

4. Complete all previous steps, then modify the above program so it performs as follows:
   - Add an option 'C' to the menu to calculate and display statistics based on the entered temperature readings, including the average temperature, highest temperature, and lowest temperature.
   - This could involve loops.
   - Modify the program to handle multiple locations. The program should continue to prompt the user to enter temperature readings for different locations until they choose to exit.

   - Add "X - Exit" to the menu:

   - The whole program should be put in a loop so that once the user has performed an action, the program will display the menu again and allow the user to select another option. The program should only end if the user enters X as their menu choice.

   - If the user enters an invalid menu choice, they should be informed of their mistake.

- Both uppercase and lowercase letters should be accepted for menu choices.
- As always, with chars, any erratic behaviour should be fixed. Handle invalid menu choices and inform the user of their mistake.

To achieve a B grade

5. Complete all previous steps, then modify the above program so it now works as follows:

- Add an option 'D' to the menu to display additional analysis results, including the following:
    - Number of readings above a certain threshold (e.g., above 15°C).
    - Number of readings below a certain threshold (e.g., below 5°C).
- Modify the program so that before the menu is displayed, the user is forced to enter temperature readings.
- Ask the user how many temperature readings they want to enter (up to 30).
- Then, ask the user to enter each value and add them to the array of temperatures.
- Use a specific terminator (e.g., entering '999') to signify the end of the input. Ensure that the program includes a **FOR** loop to process the inputs and stops taking inputs when the terminator value is entered.
- The program should then run as before but ensure that all menu actions work with the entered scores. The menu is displayed, the user enters what they want to do, the chosen action is performed and then the system loops back around to display the menu again. Be careful about what you place inside the main loop. The system should not force the user to enter values each time the program loops.

6. All tasks must be accompanied by written descriptions or annotations. These must show satisfactory understanding of how the code works.

To achieve an A grade

Complete all previous steps, then:

7. In most assessment sheets, to attain a higher grade, tasks will require some independent research. For instance:

- Demonstrate any alternative approaches you may have found and explain them.
- For the menu choice, you should receive the users input using a function called **getchar()** rather than **scanf()**. Investigate **getchar()**, demonstrate it and explain it.
- Please give this section an appropriate heading to ensure it is easy to find.

Assignment Preparation Guidelines

- All components of the assignment report must be Word-processed **(handwritten text or hand drawn diagrams are not acceptable)**, font size must be within the range of 11 point to 14 point including the headings, body text and any texts within diagrams.
- Standard and commonly used fonts such as Times New Roman, Arial or Calibri should be used.
- All figures, graphs and tables must be numbered and labelled with short explanations.
- Material from external sources must be properly acknowledged and cited within the text using the Harvard referencing system.
- All components of the assignment (text, diagrams. code etc.) must be submitted in one Word file.
- The report should be logically structured, the core of the report may start by defining the problem / requirements, followed by the proposed solution including a detailed discussion, analysis and evaluation, leading to the implementation and testing stage, finally a conclusion and a personal reflection on learning.
- Screenshots without description / discussion are not suitable as they do not express your understanding or support your work adequately.

Submission instructions

- This is a portfolio assignment with eight tasks in total. Each task will be completed and saved in the portfolio. Once the portfolio is completed, it should be submitted on Turnitin. The submission link to Turnitin can be found under the “Assessment Tab” in your module section in the SOL VLE.

- Please note file size limitation might apply. Your report must be under 250MB.

- The source code for each task should be **zipped** and **attached** to your Word document report submission in the appendix.

- The Assignment Brief can be found under “Assessment Tab” in your module section in the SOL VLE.

- **Refer to the Assignment Brief** to find the links to Late Submissions, Extenuating Circumstances, Academic Misconduct, Ethics Policy, Grade marking and Guidance for online submission through Solent Online Learning (SOL).