Lab08

1. Lab Topics

This lab covers different topics, but more emphasis is placed on function documentation, random number generation, pass by reference.

2. Documentation

REMEMBER LAB07. This lab builds on that in which majority parts of the code remain the same but now you are adding more functions to the student portal **and properly documenting your functions.**

Step 1:

Write a proper documentation for **ALL** functions that you have written for lab07. Make sure to write a brief description, the Precondition, and the Post-condition.

3. Student Portal – Smash the Bug

You are to write a short game for the student portal. You are tasked with creating a simple game students can play whenever they would like to distract themselves from the stress of school. The game is called Bug smasher.

The idea of the game is as follows; on a straight line, there exist a bug laughing at you. You are to smash the bug by guessing which position the bug is. If the number you guessed matched the exact position of the bug, you smashed it. Else, your program will give you an idea location of where the bug is by telling you to move either left or right. This will repeat till you are able to finally smash the bug and then print out the number of attempts you made.

Step 2:

1. Implement a function **positionOfBug()** that generates a random number between 1-100. Let it return an integer which correspond to the position of the bug.
2. Implement a function **getUserGuess()** that returns a user's input.
3. Implement a function **catchTheBug()** that takes in the position of the bug and the user's guess. Check these two numbers and return to the user if the user has successfully smashed the bug, should make a guess to the right or to the left.
4. Remember to ask if the user wants to play the game again.

Example output 1: Bug Smasher

Welcome to Bug Smasher!

```
Enter your guess (1-100): 45
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 90
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 34
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 95
```

```
The bug is to the left of your guess. Try a lower number.
Enter your guess (1-100): 92
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 93
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 94
Congratulations! You smashed the bug in 7 attempts.
Do you want to play again? (y/n): Y
Welcome to Bug Smasher!
Enter your guess (1-100):
```

4. Integration of the Bug smasher with the Teacher/Student portal

Integrate the Smash the bug game with the portal you have in lab07.

Step 3:

Edit your code in lab07 to integrate it with what you've done in **03.**

Step 4:

Test your program by entering sample data. You can use the example output as guidelines.

General Hints:

1. Remember to initialize your variables appropriately and at the right location.
2. Implement input validation to ensure user input is within valid ranges.
3. Keep the code organized and use clear variable names.
4. Make sure the program gracefully exits when the user chooses to do so.
5. Feel free to copy the codes from your previous lab.
6. We are assuming the user will only enter the first name as input into the **getUserName** function such that we can use *cin* to collect user's input

Example output 2: Student Menu

```
Enter your name: Jo
Enter your 5-character ID: ,./rt
Welcome student Jo
You logged in at Sun Oct 15 22:43:49 2023

Student Menu:
1. Under Construction
2. Smash the bug
3. Exit
Enter your choice: 2
Bug Smasher
Welcome to Bug Smasher!
Enter your guess (1-100): 23
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 50
```

```
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 70
The bug is to the left of your guess. Try a lower number.
Enter your guess (1-100): 60
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 66
The bug is to the right of your guess. Try a higher number.
Enter your guess (1-100): 68
The bug is to the left of your guess. Try a lower number.
Enter your guess (1-100): 67
Congratulations! You smashed the bug in 7 attempts.
Do you want to play again? (y/n): n
```

Example output 3: Teacher Menu
Sample output in teacher's menu remains the same.

BONUS (10 points):
Implement a 2D matrix for the smash the bug program such that the user have to enter 2 numbers to represent the X and Y coordinate and then instead of your program directing the user based on left and right, it directs the user based on left, right, up down.
Note: Little/No help will be provided for this bonus. Students are expected to research about it and learn how it is being used.

5. How to get full marks

To get a 100% on this lab your code should:

1. **Proper Documentation of ALL functions**
2. Use good variable names such that one can easily understand a variable's purpose just by looking at the name.
3. The program needs to be intuitive (e.g., display proper messages while you are taking user input or printing the result)
4. Follow all good coding conventions such as proper indentation.
5. Adhere to all coding standards outlined in lab2.
6. Follow the instructions of cloning, making dir, and submitting your code to git as previously discussed in lab01 and lab02
7. Comment your code properly (do not write comments for things that are obvious)
8. Push your most recent code in git and submit through canvas as well. The canvas submission should include following files:
    a. **The cpp file downloaded from git.**
    b. **At least 1 Image files**
        i. **Screenshot of the right result when a student logs in.**