Lab06

1. Lab Topics

This lab covers different topics, but more emphasis is placed on functions and modularization, switch case statement, ternary operators.

2. Grading System Part 2

REMEMBER LAB04. This lab builds on that in which major parts of the code remains the same but now you are to modularize it.

You are tasked with creating a simple C++ program that helps a teacher calculate and display statistics for a class of students. The program will allow the teacher to enter student data and then provide statistics like the average score, highest score, lowest score, and the number of students who passed or failed. Your task is to create this program using functions for better organization and readability. Follow the steps below:

Step 1:

Write a C++ program that calculates and displays the statistics for a class of students. Use the following guidelines:

Functions:

1. Create a function called **getNumberOfStudents**() that asks the user to input the number of students in the class. Ensure that the input is a valid number between 1 and 100. Return the number of students as an integer.
2. Create a function called **getStudentData**() that takes the student number (e.g., 1st, 2nd, 3rd) as an argument and allows the user to input the student's name and score. Ensure that the score is a valid number between 0 and 100. Update the relevant statistics (total score, highest score, lowest score, etc.) within this function. You may want to make use of pass by reference for this function as given below.

```
void getStudentData(double &totalScore, double
    &highestScore, double &lowestScore, string
    &studentWithHighestScore, string
    &studentWithLowestScore, int &passCount, int
    &failCount)
```

3. Create a function called **displayStatistics**() that calculates and displays the following statistics:
   - Average score for the class.
   - Highest score and the student's name who achieved it.
   - Lowest score and the student's name who scored it.
   - Number of students who passed (scored 60 or higher).
   - Number of students who failed (scored below 60).

Main Program Flow:

1. In the main function, implement an infinite loop that keeps the program running until the user decides to exit.
2. Call the function **getNumberOfStudents()** to get the number of students, initialize the variables totalScore, highestScore, lowestScore, studentWithHighestScore, studentWithLowestScore, passCount, failCount.
3. Create a for loop that goes through the number of students. Inside this for loop, call the **getStudentData()** to obtain the various values.
4. After getting the necessary data, call the **displayStatistics()** functions to show the results from the **getStudentData()** function.
5. After each menu option, ask the user if they want to continue running the program. If the user's input is not one of the affirmative options (e.g., "Yes," "YES," "y," "Y"), exit the program gracefully.

Step 2:

Test your program by entering sample data for a class of students and verify that it correctly calculates and displays the statistics just as in Lab04.

General Hints:

1. Use functions to break down your previous lab program into smaller, manageable parts.
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. Essentially, you should have the same output from your previous lab.

Example output still remains the same as in lab04

3. Simple Calculator

Write a C++ program that simulates a simple calculator using functions, switch case statements, and ternary operators. Your program should perform basic arithmetic operations (+, -, *, /) on two numbers entered by the user. Create functions for each operation and display the result. Additionally, create a function to handle invalid input.

Requirements:

1. Create four functions, one for each arithmetic operation (+, -, *, /).
2. Use a switch case statement to determine which operation to perform based on user input.
3. Use a ternary operator to handle division by zero and display an error message in such cases.

Example Output:

```
Enter an operation (+, -, *, /): +
Enter two numbers: 45
23
Result: 68
```

BONUS (10 points):

1. Create at least 2 other functions from the ones prescribed in question 1 that could further modularize the whole program.
2. Use Arrays to handle the Score and names of the students.

Note: No help will be provided for this bonus. Students are expected to research about it and learn how it is being used.

4. How to get full marks

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

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