

**Student Lab Activity**

1. Lab # CIS CIS170C-A3
2. Lab 3 of 7: Looping

1. Lab Overview – Scenario/Summary

You will code, build, and execute a program that will use looping to determine the score for a diver based on individual judge’s scores.

Learning outcomes:

1. Become familiar with the different types of looping structures.
2. Be able to debug a program of syntax and logic errors.
3. Be able to use the debug step-into feature to step through the logic of the program and to see how the variables change values.
4. Deliverables

|  |  |  |
| --- | --- | --- |
| **Section** | **Deliverable** | **Points** |
| **Lab 3** | Step 6: Program Listing and Output | **45** |

1. Lab Steps

**Preparation:**

If you are using the Citrix remote lab, follow the login instructions located on the iLab tab in Course Home.

Locate the Visual Studio 2010 icon and launch the application.

**Lab:**

|  |
| --- |
| **Step 1:** Requirements - DIVER Scoring Program |
| Your State Dive Association presently scores its diving competitions with pencil and paper. They would like for you to design and develop a Dive Program in C++.  **The paper forms that they presently use have the following:**   Diver’s Name, City JudgeScore1 - The scores entered are from 0 to 10. JudgeScore2 JudgeScore3  JudgeScore4 JudgeScore5  DegreeOfDifficulty - This is assigned once for each diver.  OverAllScore - The overall score is the individual diver’s scores totaled and then divided by the degree of difficulty. The highest and lowest scores are removed as they are often skewed entries. Total the three scores left, divide them by 3, and then multiply that by the DegreeOfDifficulty. The degree of difficulty ranges from 1.00 to 1.67.    Display the diver's information and overall score.  When the competition is complete, there is a *summary report*created that lists the total number of divers and the average of the overall scores.  **Lab hints:** When writing this lab, use nested loops. A nested loop is when one loop is completely contained in another loop. In an inner loop, you will read the five scores one at a time. Every time you read the score (in the loop), you will compare the score to the highest so far and also to the lowest so far so you can determine the highest and lowest scores, in addition to adding the scores up one at a time.  **You also need to have your program process multiple divers.** Put this in an outer loop. After you process the information for one diver, prompt the user if she/he wants to process another diver. Allow the user to type either a “Y” or “y” to enter another diver's information; otherwise, exit the loop. Write an event summary by calculating and displaying the average score for all divers and the total number of divers participating.  **Garbage in Garbage Out (GIGO):** The data being entered by the user needs to be validated. Scores by judges may range between 0 and 10. If the user enters an invalid score, display an error message, and prompt for the score again. Keep doing this until the user enters the score correctly. The degree of difficulty may range from 1.00 to 1.67.  **Sample output from program**  **Report to the media**  **Event: Diving competition**  **Enter the diver's name: Sue Jones**  **Enter the diver's city: Dallas**  **Enter the score given by judge #1: 45**  **Invalid score - Please reenter (Valid Range: 0 - 10)**  **Enter the score given by judge #1: 3**  **Enter the score given by judge #2: 4.5**  **Enter the score given by judge #3: 6.7**  **Enter the score given by judge #4: 89**  **Invalid score - Please reenter (Valid Range: 0 - 10)**  **Enter the score given by judge #4: 8**  **Enter the score given by judge #5: 9.2**  **What was the degree of difficulty? 1.9**  **Invalid degree of difficulty - Please reenter (Valid Range: 1 - 1.67)**  **What was the degree of difficulty? 2**  **Invalid degree of difficulty - Please reenter (Valid Range: 1 - 1.67)**  **What was the degree of difficulty? 1.2**  **Diver: Sue Jones, City: Dallas**  **Overall score was 7.68**  **Do you want to process another diver (Y/N)? y**  **Enter the diver's name: Dave Smith**  **Enter the diver's city: Houston**  **Enter the score given by judge #1: 5.7**  **Enter the score given by judge #2: 6.8**  **Enter the score given by judge #3: 7.6**  **Enter the score given by judge #4: 8.7**  **Enter the score given by judge #5: 6.7**  **What was the degree of difficulty? 1.1**  **Diver: Dave Smith, City: Houston**  **Overall score was 7.74**  **Do you want to process another diver (Y/N)? n**  **EVENT SUMMARY**  **Number of divers participating: 2**  **Average score of all divers: 7.71**  **Press any key to continue . . .** |
| **Step 2:** Processing Logic |
| Using the pseudocode below, write the code that will meet the requirements.  Write report heading  Loop as long as there are divers to process  Input diver's name and city  Initialize highest score, lowest score and total score  Using a do-while loop input the 5 judge's scores  Validate the score to ensure it is between 0 and 10  Add score to total  Determine highest and lowest scores  Input and validate the degree of difficulty  Calculate the overall diver's score  Display the diver's information and overall score  Add diver's overall score to the final score  Add 1 to the number of divers    Prompt the user if she wants to process another diver  End-Loop  Calculate the average score for all divers  Display the number of divers and the average score for all divers |
| **Step 3:** Create a New Project |
| Create a new project and name it LAB3.  Write your code using the Processing Logic in Step 2. Make sure to save your program. |
| **Step 4:** Build Solution |
| To compile the program, click **Debug** then **Build solution (F7).** You should receive no error messages. If you see some error messages, check the code above to make sure you didn’t key in something wrong. Once you make your corrections to the code, go ahead and click Build >> Build Solution again. |
| **Step 5:** Execute the Program |
| Once you have no syntax errors, to execute or run your program, click **Debug** on the menu bar and then click **Start Debugging.** |
| **Step 6:** Capture the Output |
| 1. Capture a screen print of your output. (Do a PRINT SCREEN and paste into an MS Word document.) 2. Copy your code and paste it into the same MS Word document that contains the screen print of your output. 3. Save the Word document as Lab03\_LastName\_FirstInitial. |
| **END OF LAB** |