|  |
| --- |
| **CS 1400 Lab #16**  **Dice Game Implementation**  **Version 1.1**  **Objectives:**  The objective of this lab is to help you understand how to write code that uses methods, validates input, uses conditionals, and loops together with random numbers. You designed a solution to this problem, as Pseudo-Code, in the previous lab.  **The problem**  For this assignment, you will be writing a program that simulates rolling a pair of dice. Your program should be written as a set of methods. DO NOT use only class level const and static Randowm randNumb = new Random( ) in your program. The program will perform the following steps:   * Create a Random Number generator object. Make sure this object is a class level variable. * Ask the user if they want to play the dice game. * Verify that the menu selection is valid and notify the user that the input is invalid, play the game or quit the program. * If the user reponds with ‘***Q***’ quit the application. * If the user responds with a '***Y***' then * Play the dice game by   + Generate two random numbers in the range 1 - 6   + If the two numbers are 6 and 6, display the message " ***BOX-CARS***" and the values of the two die.   + If the two numbers are 1 and 1 display the message "SNAKE-EYES" and the values of the two die.   + In all other cases display the message "{die\_1} – {die\_2}" values of the two random numbers.   + When the user presses the “Enter Key” roll the dice again; however, if the user enters the ‘Q’ key and presses the “Enter Key” return to step 2 and ask the user again if they want to play the game again.   **Deriving the Solution**  You should have developed the design in an ***Activity Design Worksheet***, ***Activity Diagram*** and ***Pseudo-Code*** for this program from lab #15.  **Writing the Code**  Start Visual Studio and start a new Project Lab\_16 and copy the ***Pseudo-Code*** you created for ***Lab\_15*** and paste it in this editor page. Now, fill in the C# code that will implement the ***Pseudo-Code*** that you wrote in ***Lab\_15***. Notice how your ***Pseudo-Code*** now serves as documentation for your code. It makes it easier for others to understand how your program works.  Pay special attention to the example executable in the ***Lab\_15*** folder on Canvas.  **File(s) to Submit:**  Place your complete project folder in a zip file and name the zip file Lab\_16\_your-initials\_V1.0.zip. For example, I would name my file Lab\_16\_DAF\_V1.0.zip. Submit this assignment as Lab #16 on Canvas.  **Grading Checklist** |

|  |  |  |  |
| --- | --- | --- | --- |
| # | Program | 1st Submission | 2nd Submission |
| 1 | Meets & works to specifications | Correct | Xnot | 6 points |
| 2 | Error Free, elegant & efficient | C | X | 4 points |
| 3 | Pseudo-Code | C | X | -3 points |
| 4 | Style Guidelines | C | X | -2 points |
| 6 | Source Files(s) & Formatting | C | X | -2 points |
| 7 | Project Prolog | C | X | -1 points |
| 8 | Function Prologs | C | X | -1 points |
| 9 | Zip Filename | C | X | -1 points |
| 10 | Lab & Project Names | C | X | -1 points |
| 11 | Zip File is invalid or will not unzip | Lab = 0 pts | Lab = 0 pts |
|  | Total Points | 10 | 0 | 10 | 0-9 |