**Diploma in Software and Design**

**Assignment Cover Sheet**

|  |  |
| --- | --- |
| **Course name:**  Diploma in Software and Design | **Student’s name:**  Harmanpreet Singh |
| **Module Name /or number: Object Oriented Programming & Data (25 credits)** | |
| **Assignment title and/or number**:  C#.Net Assessment | |
| **Assessment weighting** | *Need to pass the assessment to complete the course* |
| **Passing Criteria:** | Need to score 50 or more marks to pass the assessment.  **Total Marks : 100** |
| **Due date**: | **Date submitted**: June 5, 2020  (late submissions incur 10% penalty, after 7 days late, the assessment will not be marked) |
| **Assessment conditions:** | This is a resource-based assessment. This means that you may have access to any relevant resources to assist you. This could include, for example, your learning materials, information on the Internet, and so on. However, all work must be your own with no assistance from any other person. |
| **Submission requirements:** | You’re required to upload the following on Cloud Campus:   * This document, completed where appropriate * Visual Studio project files * Upload your project on Github and paste the link below   GitHub link below: <https://github.com/chahalharman3426/Harman-assignment> |
| **Learning Outcomes:** | User experience (Ux) design including user interface (UI), HCI principles, and universal accessibility;  Coding – object oriented, procedural;  Source and version control;  Application of the core software development concepts and practice, underpinned in the fourth outcome of the New Zealand Certificate in Information Technology (Level 5) [Ref: 2595]. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Assignment Checklist:** | |  |  | | --- | --- | | **Requirement** | **Completed** | | Database | [Symbol][Symbol] | | User interface | [Symbol][Symbol] | | Functionality | [Symbol][Symbol] | | Coding | [Symbol][Symbol] | | Testing | [Symbol][Symbol] | |

**Disclaimer of Plagiarism and Collusion**

I declare that, to the best of my knowledge, this assessment is my own work, and has not been copied from any other student's work or from any other source.

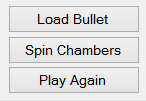
Enter your name here to indicate you agree to the above statement.

Your Name Harmanpreet Singh

## DSD -01 UI Based Simple Game

*Due Date: September 4, 2019*

This project involves playing Russian roulette with a gun, or any other method that meets the criteria specified below.

The program runs like this. (with a button for each step)

1 You **load** the bullet into the chamber of a revolver (not an automatic)

2 You **spin** the chamber to create a random place where the bullet is with a 1 in 6 chance.

3 You pull the **trigger** repeatedly until the gun fires through to the number where the bullet is stored.

4 The player has **2 chances** to shoot away during the game, this means that if the bullet is fired during that time they survive. If they shoot away twice and still the bullet has not been fired then they better make their will because the next shot they die.

## What your project needs.

* At least one class
* At least two Unit Tests of major parts
* At least one sound
* At least one image
* Sound and image should be loaded from the resource folder.
* All significant code to be commented
* Project to be hosted on **Github** (include your url)
* **No variables in the code – all in the classes**
* **Include WIN / LOSE and Total scores**

You can and should use any examples we have made to date for help.

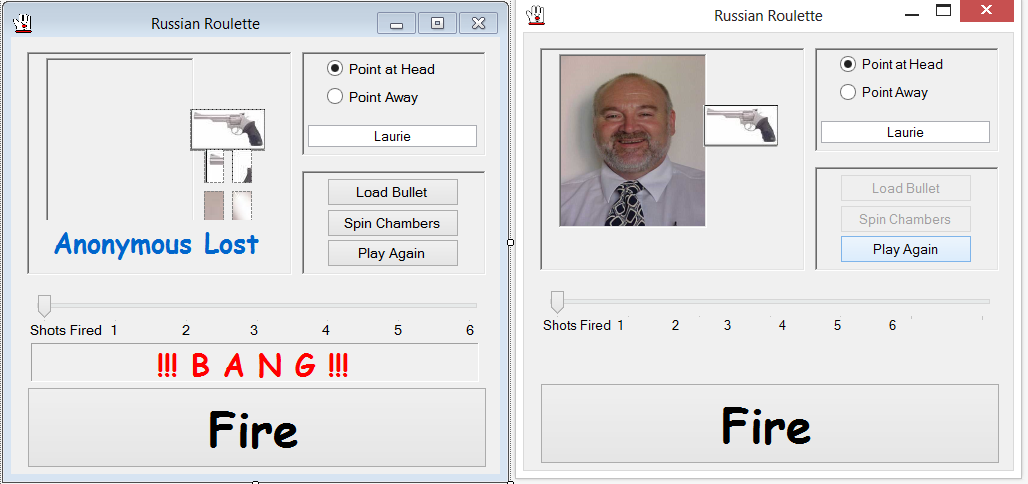
You can use the net, (although that won’t be much help).

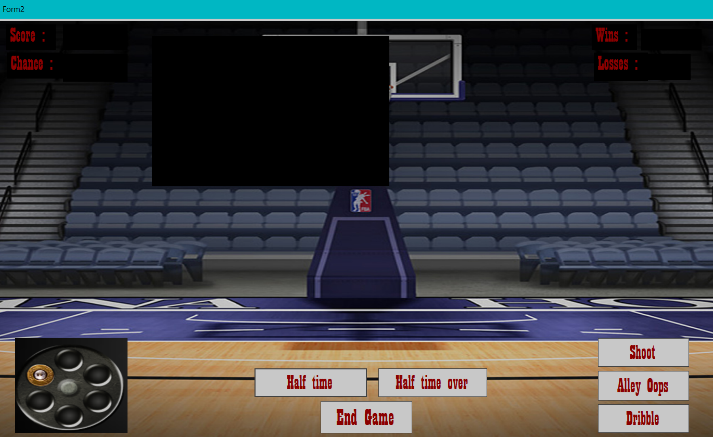
You can ask for help.

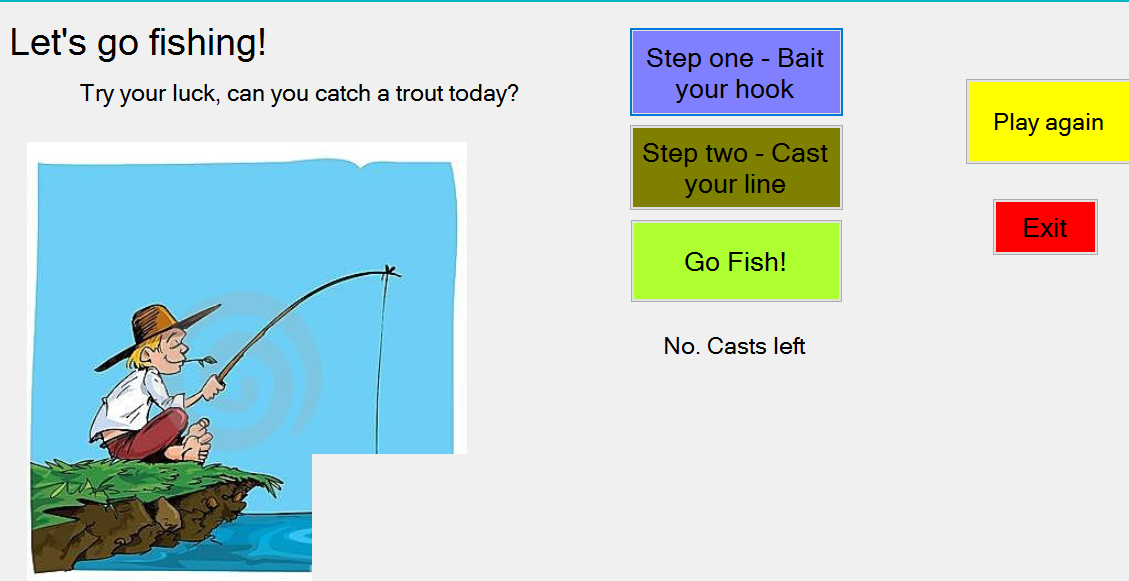
But the SIGNIFIGENT MAJORITY of your program has to be your own work

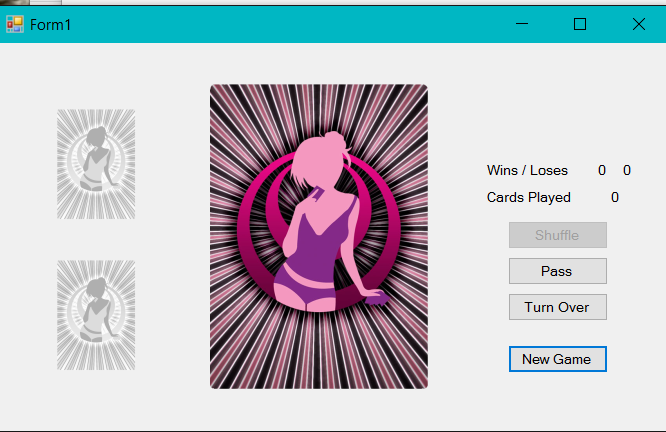
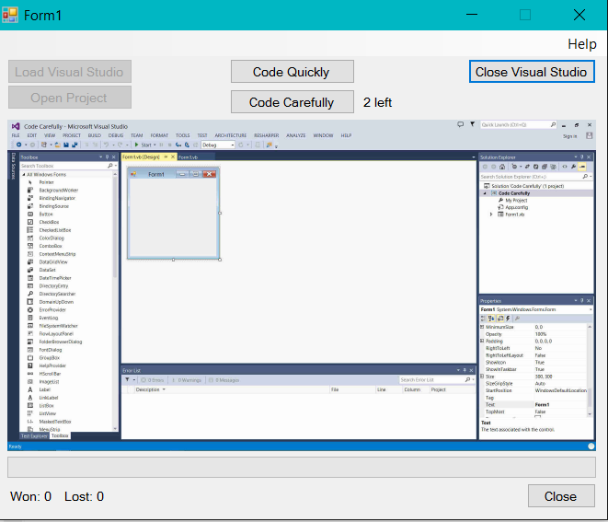
Look at the examples of past students work for ideas.

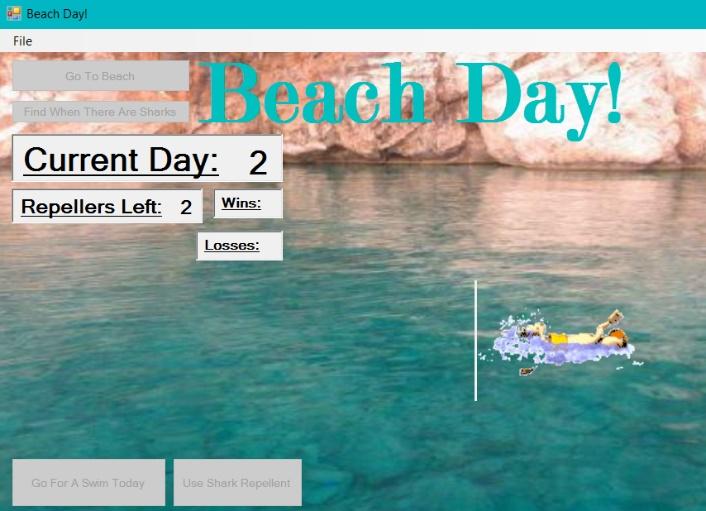
Be creative. **DO NOT make your program look like these images show.** Use your imagination and Coding Ninja skills to change it into something far more entertaining and surprise us.



Examples of Students work



.  



***You should pay attention to ensure that your code is well structured; appropriately commented and meaningful variable names have been used.***

### Marking Schedule

|  |  |  |
| --- | --- | --- |
|  | Form features | |
|  | 1.1 | At least one Image |
|  | 1.2 | At least one sound |
|  | 1.3 | Include WIN / LOSE and Total scores |
|  | Class Operations | |
|  | 2.1 | At least one class |
|  | Unit test | |
|  | 3.1 | At least two Unit Tests of major parts |
|  | Code Requirements | |
|  | 4.1 | At least one sound |
|  | 4.2 | Sound and image should be loaded from the resource folder. |
|  | 4.3 | All significant code to be commented |
|  | 4.4 | No variables in the code – all in the classes |
|  | 4.5 | Project to be hosted on Github (include your url) |

## Marking matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **% of Grade** | **Excellent 100%** | **Adequate 80%** | **Poor 60%** | **Not Met 0%** |
|
| **Program Specifications / Correctness** | | |  |  |
| 50% | No errors, program always works correctly and meets the specification. | Minor details of the program specification are violated, program functions incorrectly for some inputs. | Significant details of the specification are violated, program often exhibits incorrect behaviour. | Program only functions correctly in very limited cases or not at all. |
| **Mark** | 50 | 40 | 30 | 0 |
| **Readability** | | |  |  |
| 20% | No errors, code is clean, understandable, and well-organized. | Minor issues with layout, variable naming, or general organization. | At least one major issue with layout, variable names, or organization. | Major problems with at three or four of the readability subcategories. |
| **Mark** | 20 | 16 | 12 | 0 |
| **Documentation** | | |  |  |
| 20% | No errors, code is well-commented. | One or two places that could benefit from comments are missing them **or** the code is *overly* commented. | Complicated lines or sections of code uncommented or lacking meaningful comments. | No comments present. |
| **Mark** | 20 | 16 | 12 | 0 |
| **Code Efficiency** | | |  |  |
| 5% | No errors, code uses the best approach in every case. | *N/A* | Code uses poorly-chosen approaches in at least one place. | Many things in the code could have been accomplished in an easier, faster, or otherwise better fashion. |
| **Mark** | 5 | 4 | 3 | 0 |
| **Assignment Specifications** | | |  |  |
| 5% | No errors | *N/A* | Minor details of the assignment specification are violated, such as files named incorrectly or extra instructions slightly misunderstood. | Significant details of the specification are violated, such as extra instructions ignored or entirely misunderstood. |
| **Mark** | 5 | 4 | 3 | 0 |