# Project Milestone #3

**Important Information:**
This is an individual assignment for you to complete on your own. If you need help, take advantage of the Piazza discussion board, online help hours or contact your TA directly. You may not work with or assist another student or receive help from anyone other than the CS 177 teaching staff.

**Due Date**

Check BrightSpace for the **Milestone 3** due date. Keep in mind that late submissions will not be accepted for credit.

*Read the entire Milestone 3 document before starting this assignment*

**Project Description: The *Black-n-Goldtzee* Game**

*Black-n-Goldtzee* is an animated video game where players roll six dice to earn points. Each project milestone is designed to build another portion of the overall *Black-n-Goldtzee* game which will be complete after the final milestone. In **Milestone 1**, you wrote a program that generates a series of random Integers and displays them in a formatted report. For **Milestone 2** you created the *Black-n-Goldtzee* game window using the Python Graphics Library and gave the user the ability to "roll" six dice.

**The *Black-n-Goldtzee* Scoring Window**

Your goal for **Milestone 3** is to enable full game play by modifying the *Dice* window from Milestone 2, creating the *Scorecard* window and enabling three program modes. These will allow the user to decide if they want to play a game or exit (`Startup` mode), to play and score the game by rolling the dice 5 times (`GamePlay` mode) and to confirm their final game score (`GameOver` mode).


*Example of **Black-n-Goldtzee** scorecard window*

The *Black-n-Goldtzee Scorecard* window has a black background and several gold *Lines* and *Rectangles* designed to separate and organize the title, labels and scoring sections. As the player rolls the dice, the score for the current roll is displayed as a gold *Text* object as shown. Scoring is determined using the *product* of the total value of the dice and the maximum number of duplicate dice in the roll. More information about scoring and examples can be found in *Appendix A* of this assignment.

For your **Milestone 3** submission, your Python program will be required to display the fully functional *Black-n-Goldtzee Dice* and *Scorecard* windows and allow the user to play the game using the three game modes described above.

You may write your program and write whatever functions you need to meet the requirements, however the functionality of the program must very precisely match the specifications given in this document.

A demonstration video for the fully functional *Black-n-Goldtzee* **Milestone 3** program can be found here:
https://youtu.be/T1eVHXG9v-o

## About Milestone 3:

In this project milestone, you will complete several parts:

1. Setup your `milestone3.py` file
2. Create the *Scorecard* window
3. Modify the *Dice* window for `Startup` mode
4. Enable `GamePlay` and `GameOver` program modes
5. Submit your completed program

## Part 1: Setup your `milestone3.py` file

The Python program file should be setup with the following guidelines:

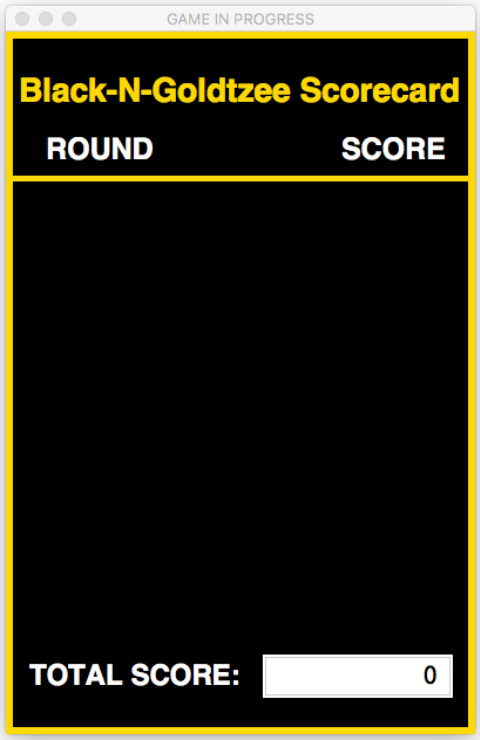
- Start with a fully-functional, complete `milestone2.py` program with all outstanding issues resolved
- File name is `milestone3.py`
- All library import statements should be at the top of the file
- File includes a header describing program, its purpose and function
- Header includes your name, the program name (`milestone3.py`) and a description of its function
- All Python code should be contained within function definitions
- Plan and fully document your program changes using descriptive pseudocode comments

## Part 2: Create the *Black-n-Goldtzee Scorecard* window

Define a new, separate function that creates the *Black-n-Goldtzee Scorecard* window. This should closely match the example shown here and this function should accomplish the following tasks:

- Create a `400x600` *Graphics* window with a black background titled `"GAME IN PROGRESS"`
- Using *Text* objects, create the title and column labels using Helvetica font using the appropriate location, size and color to match the example as closely as possible.
- Create an *Entry* box, width `10` and anchored at `300,550` using `24pt`, white `Courier` font. *Strings* displayed in this *Entry* box should be right-aligned and initially be `"0"` as shown.
- Create five gold *Lines* (width `5`) to create the outside and horizontal border as seen in the *Scorecard* window example.
- **Return** the *Graphics* window and the *Entry* box objects.

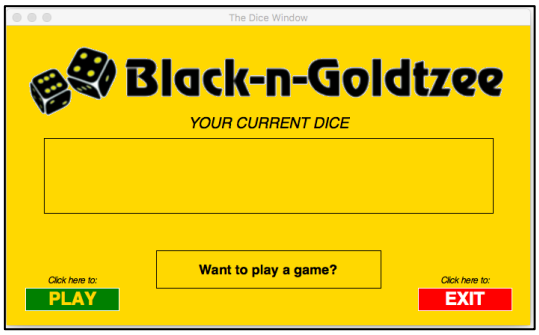
*When specific `x,y` coordinates or font sizes are not provided for a specific graphics object, use your best judgement to match the examples shown.*




*The scorecard window*

## Part 3: Modify the *Dice* window for `Startup` mode

Modify the `diceWindow()` function so that window initially represents the first program mode, `Startup`. The *String* displayed in the *Text* object centered in the `ROLL` control *Rectangle* will be changed depending on the program mode, so it will also need to be **returned** by `diceWindow()`.




*The Dice window in* `Startup` *mode*

1. Change the fill color of the `ROLL` control *Rectangle* to green and the *Text* object to display the *String* `"PLAY"`
2. Change the *Text* in centered within the `300x50` *Rectangle* to display `"Want to play a game?"`
3. Modify the `diceWindow()` function's **return** statement to also include the `"PLAY"` *Text* object.

   *NOTE: This will also requires a change to the* `diceWindow()` *function call in* `main()`

## Part 4: Enable the `GamePlay` and `GameOver` program modes

When the user clicks `PLAY` to start a game, the program should operate in `GamePlay` mode which performs the following tasks:


1. Change the fill color of the `ROLL` *Rectangle* to black and the *Text* object to display the *String* `"ROLL"`
2. Call the function that creates the *Scorecard* window and assign the **returned** objects to variables
3. Use a loop to allow the user to `ROLL` five times. After each roll:
   a. Generate and display the six random dice in the *Dice* window (following the **Milestone 2** specifications)
   b. Calculate a score for the roll (see *Appendix A for scoring details and examples*).
   c. Update the *Text* in the `300x50` *Dice* window *Rectangle* to display the results of the roll
   d. Display the results of each roll in the *Scorecard* window
   e. Update the `TOTAL SCORE` *Entry* box with the sum of the roll scores

After five (`5`) rolls, the program should operate in `GameOver` mode which performs the following tasks:

- Displays a gold-filled Rectangle above the `TOTAL SCORE` and *Text* objects containing the *Strings* `"GAME OVER"` and `"Click HERE to continue"` displayed as shown
- Wait for a click within this *Rectangle* before closing the *Scorecard* window

  NOTE: The player must click within the `GAME OVER` *Rectangle* to close the *Scorecard* window
- After closing the *Scorecard* graphics window, the *Black-n-Goldtzee* game should go back to `Startup` mode (*see Part 3*).

**Part 5: Submit to BrightSpace**

- Upload your completed file (`milestone3.py`) to the Milestone 3 assignment on BrightSpace.
- Your assignment must be submitted by the due date to receive credit.

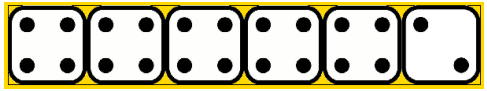
| **Grading Rubric** | **Points** |
|---|---|
| **Part 1: Setup Your** `milestone3.py` **File** | |
| All **Milestone 2** requirements and specifications are complete, functional and included | 10 |
| File name is `milestone3.py` | 5 |
| All library import statements are at the top of the file | 5 |
| Program header included with student name, program name and description | 5 |
| All Python code is contained within functions | 5 |
| Program is fully documented with descriptive pseudocode comments | 15 |
| **Part 2: Create the *Black-n-Goldtzee Scorecard* window** | |
| New, separate function is defined to create the *Scorecard* window | 5 |
| *Scorecard* window is 400 x 600 with a *black* background | 5 |
| *Text* objects created to display the title and Helvetica column labels as specified | 10 |
| *Entry* box object width `10`, anchored at `300,550` as specified | 5 |
| Entry box **always** displays *String* `"0"` in right-aligned, 24pt `Courier` font face | 5 |
| Five gold *Lines* (width `5`) are drawn to create the outside and horizontal borders | 5 |
| Function **returns** the *Graphics* window and *Entry* box objects | 5 |
| **Part 3: Modify the Dice window for `Startup` mode** | |
| Fill color of `ROLL` control *Rectangle* is green in `Startup` mode | 5 |
| *Text* object displays `"PLAY"` in `Startup` mode | 5 |
| `diceWindow()` function's **return** statement is modified to include `"PLAY"` *Text* object | 5 |
| **Part 4: Enable the `GamePlay` and `GameOver` program modes** | |
| `ROLL` *Rectangle* fill color is black and *Text* object displays `"ROLL"` during `GamePlay` | 5 |
| Function to create the *Scorecard* window is called at the start of `GamePlay` mode | 5 |
| Loop is defined to allow the user to roll the dice five (`5`) times | 5 |
| *Roll Score* is correctly calculated for each roll using sum of dice * max dups multiplier | 10 |
| *Text* object in `300x50` *Dice* window *Rectangle* is updated to show score as specified | 10 |
| *Text* objects created in the *Scorecard* window shows *Roll #*, *Result Name* and *Roll Score* | 15 |
| `TOTAL SCORE` value in the *Entry* box is updated correctly during `GamePlay` | 10 |
| Program switches to `GameOver` mode after `5` rolls, displays gold *Rectangle* and *Text* objects | 10 |
| User must click `GAME OVER` *Rectangle* to close the *Scorecard* window and continue | 5 |
| After closing the *Scorecard* window, the program returns to `Startup` mode | 5 |
| Program ends **only** when the EXIT control Rectangle is clicked in Startup or GamePlay modes | 10 |
| **Style / Format:** Python code is formatted well and easy to understand | 10 |
| **Total Points** | **200** |

## Appendix A: Scoring the Dice in a Roll

To determine the score for an individual roll, the first step is to determine the maximum number of duplicate values. This will determine the "*Result Name*" and dice total multiplier in one of the following categories:

| Result Name | Description | Multiplier/ Max Dups |
|---|---|---|
| `No Duplicates` | 6 unique dice with no duplicate values | 1 |
| `Pair` | Any combination of one, two or three pairs of matching dice | 2 |
| `Three of a kind` | Any combination of three matching dice | 3 |
| `Four of a kind` | Four matching dice | 4 |
| `Five of a kind!!` | Five matching dice | 5 |
| `BOILER UP!` | All six dice with the same value | 6 |

The *Result Name*, the sum of the dice values and the *Roll Score* will be displayed in the *Dice* window after each roll. The roll number (`1-5`), *Result Name* and *Roll Score* are displayed in the *Scorecard* window along with the sum total of all the *Roll Scores*. Review the examples provided in this assignment and the demonstration video to visualize how this all comes together.

| Example Roll: | Result and Analysis | Roll Score |
|---|---|---|
|  | `No Duplicates`<br>*All unique values*<br>**Max duplicates = 1** | *Dice total*: `6+4+3+2+5+1 = 21`<br>*Total * Max Dups:* `21 * 1 =` **21** |
|  | `Pair`<br>*There are two `4's`*<br>**Max duplicates = 2** | *Dice total*: `5+3+4+2+1+4 = 19`<br>*Total * Max Dups:* `19 * 2 =` **38** |
|  | `Three of a kind`<br>*There are three `3's` & two `1's`*<br>**Max duplicates = 3** | *Dice total*: `1+3+3+3+6+1 = 17`<br>*Total * Max Dups:* `17 * 3 =` **51** |
|  | `Pair`<br>*There are two `2's` & two `5's`*<br>**Max duplicates = 2** | *Dice total*: `2+6+4+2+5+5 = 24`<br>*Total * Max Dups:* `24 * 2 =` **48** |
|  | `Four of a kind`<br>*There are four `4's` & two `5's`*<br>**Max duplicates = 4** | *Dice total*: `4+5+5+4+4+4 = 26`<br>*Total * Max Dups:* `26 * 4 =` **104** |
|  | `BOILER UP!`<br>*There are six `6's`*<br>**Max duplicates = 6** | *Dice total*: `6+6+6+6+6 = 36`<br>*Total * Max Dups:* `36 * 6 =` **216** |
|  | `Three of a kind`<br>*There are two `5's` & three `2's`*<br>**Max duplicates = 3** | *Dice total*: `5+2+3+5+2+2 = 19`<br>*Total * Max Dups:* `19 * 3 =` **57** |
|  | `Three of a kind`<br>*There are three `4's` & three `5's`*<br>**Max duplicates = 3** | *Dice total*: `5+4+5+5+4+4 = 27`<br>*Total * Max Dups:* `27 * 3 =` **81** |
|  | `Five of a kind!!`<br>*There are five `4's`*<br>**Max duplicates = 5** | *Dice total*: `4+4+4+4+4+2 = 22`<br>*Total * Max Dups:* `22 * 5 =` **110** |