|  |  |
| --- | --- |
| **CIS 2531 - Final Project** | **Comprehensive Project** |
| **Chapters 2 - 13** | **200 Points Available** |
| **Assignment Purpose:**  The purpose of this lab assignment is to demonstrate knowledge python programming and object-oriented design | |

You are going to create a game or application that utilizes many of the features of the course. You are to build a single game or application that is built using an object-oriented paradigm and utilizes many of the features from the course. Most of these features you will come across naturally while building your game or application so picking the right project is the most important part early on. Not only will you code the project, but you will also develop a short video of you using the game or application and fill out a series of short written responses relating to your program and program code.

**You are to use our graphics library for this project. Using tKinter or other libraries will need written consent from your instructor and pyGame is off limits.**

**Part A: Project Code Requirements:**

1. Object Oriented: There is at least one class and one instance of that class created to simplify and encapsulate the actions of the program.
2. Graphical: Interaction with the user must be graphical and not done through the command line. That can be through our graphics library we have used throughout the course or with an application using a GUI and the tKinter Library.
3. Input from the User: Program must include some form of input from the user. That could be information entered in a text field, mouse clicks, and key button pushes, input from another device or input from a file.
4. A Data Structure: At least one data structure such as a list, tuple, or dictionary to represent a collection of data that is stored to manage program complexity
5. Functions: A minimum of 3 functions in which one has at least one or more parameters and a return to help manage program complexity.
6. An algorithm that includes sequencing, selection, and iteration
7. Object instantiations and function calls.

**Part B: Video of Demonstration:**

Submit a video file that demonstrates your program running. The video must include:

1. You execute the code file(s) to start the program
2. input to your program
3. output produced by your program.
4. Voice narration explaining what you are doing(**input**) and what the **output** is and why it is
5. Video must be no more than 2 minutes in length.

**Part C: Written Responses**

In the responses below you will be explaining your code and project. You may be asked to copy and paste or take a screenshot of code segments and then explain them. Your responses should be short and to the point.

1. Describe the overall purpose of your program. Describe the game or app that was built, how we interact with it and what we expect it to do. Who is the app or game intended for? Are there any special requirements for it to run?

The purpose of my program is to be a challenging game, with some of the difficulty coming from the poor controls and handling. The game is inspired by a game me and my friends used to play during school, and that is who its intended for. The only requirements to run it is to have all the files installed in the same place. You run it by starting the program, then you control the small ball trying to survive the shapes moving around you. If you hit anywhere but the top of a shape, you will die, otherwise you bounce up with a higher peak, hurting or killing that shape in the process. One flaw I couldn’t resolve was rolling while jumping, you can interrupt your roll with a jump, but you wont keep your inertia. you can roll in the air after you jump however.

2. Capture and paste two program code segments that contain a data structure(list, tuple, or dictionary) being used to manage complexity of your program

a. First the code segment that creates and shows the data being stored in the structure.

b. Second the code segment that shows the data structure being used.

c. Explain what the data type of the data being stored is and how it manages complexity of the program. Explain why your code could not have been written without it or how it would have been very difficult to write the program without it.

A.

Text

Description automatically generated

Stores the initial 3 enemies, one of each type. More elements are added as the list iterates through

B.

Text

Description automatically generated

The list updates each enemy position and goes through some necessary checks.

C. The data type stored here is of various classes, although they each share the same parent class of ‘Enemy’. There are some functions they share the same name for, but differ in exact execution, so they couldn’t simply be different instances of the same class. Without the list being able to support near same sub classes, I would’ve had to make 3 lists, and manage the enemy count with a bit more code than a simple if < check. If lists didn’t exist altogether I would’ve had to do a lot more work going through each enemy and updating it. Id also need to work around the appearance of new enemies.

3.Capture and paste a programming segment of a function that utilizes a parameter and a return.

Explain the value of the parameter and return and how it manages program complexity. Also state how the utilization of global variables for this operation would not be preferred.

A screenshot of a computer

Description automatically generated with medium confidence

While this function doesn’t explicitly return something, I think it is interesting because it takes a list as its parameter, and returns/appends to it an object it just created, or more specifically the location of that new object. type could be stored as a global variable, although that wouldn’t be preferred because it would then be free to change in other places, and it is named something which very well could mean different things in different contexts. although the function randomizes it each time its run, that could throw off other functions using it.

4. Capture and paste a programming segment of the above procedure being called. Give examples of two different arguments this function can be called with or is called with and explain the output of the function given the different arguments.

Text

Description automatically generated

The argument this function takes here is the enemy list, made to hold all the enemies in action. The output of it is a new enemy to the list, which is given the current window.

Though this function is pretty much only called once, If I wanted to it could alternatively be called to add a different class of enemies to a list. Such as something like a ghost list, where it adds a new moving ghost enemy for each enemy killed to obscure your vision.

5. Capture and paste a programming segment that utilizes both selection and iteration. Generally explain how to contribute to the overall functionality of the program. Explain how this function works in enough detail that your teacher could recreate it.



…..



The iteration in this program is very broad. It iterates essentially every function until the player collides with any enemies killbox. This is checked for in the enemy list explained earlier, for each enemy. Since I explained running through that list earlier, Ill explain the other half of that outcome which uses selection.

Text

Description automatically generated

for as long as the player is alive, the program will iterate through this segment which moves the player up and down. The bigger iterative loop alternates between moving the player, and enemies. And here it moves the player by first checking if he should bounce (in the cases of initially hitting up, or colliding with an enemy from the top), then it changes a few Boolean variables. after that it checks if its bouncing or falling, and then either moves up or down, all while checking if it should change those variables again.

**Rubric**

|  |  |  |
| --- | --- | --- |
| **Code** | **Description** | **Points** |
| Object-Oriented | Project contains a class that includes a constructor with multiple attributes, and multiple behaviors | 10 |
| GUI | Project contains a graphical interface using tKinter or our graphics library. User interacts with the interface and not with the console | 10 |
| User input | User input is obtained and processed by the program. Input can be text, button presses, mouse controls/click | 10 |
| Data Structure | Program contains a data structure to manage program complexity | 10 |
| Functions | A minimum of 3 functions/methods exist. One must include a parameter and a return | 10 |
| Algorithm | An algorithm that includes sequencing, selection and iteration | 10 |
| Instantiation and calls | The creation of at least one instance of the object and proper function/method calls | 10 |
| **Video** | **Description** | **Points** |
| Start of Program | Video demonstrates the execution of the program code | 5 |
| Input | Video demonstrates input being collected into the program. If it can’t be seen then it is voiced what input is being pushed to provide input (such as mouse movement, or button clicks) | 10 |
| Output | Video demonstrates the output of the program. Voice narration is also describing what we see for **the** output and why we see it | 10 |
| Length | Video is under 3 minutes in length | 5 |
| **Written Responses** | **Description** | **Points** |
| Response 1 | Describes the program | 20 |
| Response 2 | Captures of program code and explanation | 20 |
| Response 3 | Capture of program code and explanation | 20 |
| Response 4 | Capture of program code and explanation | 20 |
| Response 5 | Capture of program code and explanation | 20 |