1. Introduce the project problem you chose, including a few background details (ex: a sentence or two regarding the history of the game would be appropriate for Project Option 3).

Our group chose to make a Magic 8 Ball in Python. The original, physical Magic 8 Ball is a toy that you can ask questions to and “get” an answer from after you shake it. Our program also lets you ask questions (without shaking your computer), as well as see statistics about how many of each answer you’ve received.

2. Provide basic documentation as to how your code is to be used by somebody unfamiliar with your implementation (ex: is it a user prompt, a class with methods that a user must invoke, etc.)

The Magic 8 Ball file is a .ipynb file, so it should be opened in Jupyter Notebook. There will be one cell containing the code: this code includes the Magic 8 Ball class, as well as an instantiation of it at the bottom. Running this cell will prompt a user input loop: you type your question into the console and receive an answer. Alternatively, you can input a pound sign/hashtag , # , to see which and how many answers you’ve gotten, or input an empty string (just hit enter) to terminate the loop.

**In summary**, the Magic 8 Ball is a class with a specific method that runs the ball’s loop, but has been written in a way that lets the user immediately start the loop without knowing how to instantiate an object from a class.

3. Provide technical details, including the following:  
 i. What functions/methods you created to solve the problem, if any.

The 8 Ball has a few custom methods to meet the design requirements, as well as the use\_Magic\_8Ball() loop that lets users run the 8 Ball.

First, we have the built-in \_\_init\_\_(self) method for our class. We only include self as an argument as no other arguments are needed to run the 8 Ball, and add a few variables tied to self. See the section about data storage for more detail.

Second, there is the ask\_question(self) method, which assigns the user input of the question they wish to ask to a ‘self.question’ variable.

Third, there is the answer\_question(self) method. This question takes a random integer between 0 and 9, assigns that integer to a variable ‘rand\_index’, and indexes the list of responses with it to generate a random response. It then increments by 1 the ‘self.times\_used’ and ‘self.responses’ variables that track answer counts.

Fourth, we have the response\_details(self) method. This iterates over the list of responses and their frequency using a for loop. This loop prints out “You received this answer this many times: “, followed by each string and its corresponding number, for each sub list in ‘self.responses’.

Finally, use\_Magic\_8Ball(self) runs a loop using a flag. This loop, in order, runs the ask\_question method, and checks the following ‘self.question’ variable against 3 if statements that correspond to the three ways the loop could continue. Any string besides an empty or a # will generate an answer, a # will run the response\_details method, and an empty string input will print a short message and then set the loop flag to false.

ii. What data types and collections you used for variables/storage

User questions: strings

Response detail variables: ints

Possible responses: this variable was kept as a list of 10 sub lists: each sub list had a response at index 0 and a counter at index 1. The ‘self.rand\_index’ variable would index into this list in the answer\_question method.

Finally, the use\_Magic\_8Ball() method had a Boolean variable set up as its condition to loop.

iii. What standard and/or third-party libraries, classes, and functions you have employed.

The only pre-made library we used was the third party random module, from which we imported randint to generate an index and output a random response.

4. Explain any aspects to your chosen problem that you initially set out to tackle that you were unable to do so, or anything that isn't quite working correctly, and what you believe the issue to be.  (If nothing applies here, you should state as much.)

N/A; As far as we know, all relevant issues are debugged and everything should run as normal

5. Explain in a paragraph or so how your code could be built upon to solve a larger scale problem or include additional features that were not present.  *Note: I will be extremely flexible about your choice here, but you must put something.*

Most of what we wanted designed was accomplished, but actually using the program prints a lot of text to the console; being able to limit this or delete older messages from the console would improve the user experience, but that feature doesn’t exist right now.