


(lab) Magic Eight Ball

[Start Assignment](#)

- Due Friday by 11:59pm
- Points 100
- Submitting a file upload

Overview

This week we're going to create a wonderful old toy called the 'Magic Eight Ball.' Here's an online version you can look at if you're unfamiliar with the concept: <https://magic-8ball.com/>  (<https://magic-8ball.com/>) Essentially, it works like this:

1. You ask a yes or no question
2. You shake the eight ball
3. One of a limited sets of responses pops up in the window
4. You make all your important life decisions based on the whims of a random toy

Our version works online, and it has three different capabilities. One allows you to see all of the options. Another lets you see any of the choices by picking a number. The third part allows you to ask a question and it randomly chooses one of the responses. It looks way more magical than it really is. Here it is in action:

```
What will you do?
1: print all the fortunes
2: print a specific fortune
3: get a random fortune

Please choose 1, 2, or 3: 1
YOUR FORTUNES
0) yes
1) no
2) it's likely
3) it's doubtful
4) absolutely positive
5) not going to happen
6) for sure
7) never never never

What will you do?
1: print all the fortunes
2: print a specific fortune
3: get a random fortune

Please choose 1, 2, or 3: 2
what number do you want? (0-7): 1
no

What will you do?
1: print all the fortunes
2: print a specific fortune
```

```
3: get a random fortune
```

```
Please choose 1, 2, or 3: 3
```

```
Your question: Will I pass this course?  
absolutely positive
```

Note that I ran the program three times. We still don't have repeating behavior, but I promise we'll get to that soon.

Process

The eight ball is fun to build and fun to use. We will learn a number of very important programming ideas along the way. It might surprise you that the main point of this project is not really the code (which is relatively simple) but the way we arrange the data. There are many ways to write this program, but we will begin by building a data structure that holds all the possible results. This has a number of advantages, as you'll see.

New ideas for the week

We'll be using everything we've used before, and adding a few new twists this week. Here's the new ideas you'll need:

- investigating the random module
- using random features to get a random number
- building lists
- pulling an element out of a list
- Using a for loop to step through a list
- converting from a string to a numeric data type
- building a user menu
- selecting behavior from that menu
- handling errant input

Planning the Menu

This game has a couple of new components. The most obvious new idea is the menu. Numeric menus are a very common tool in the command-line style programs we are beginning with. This type of menu has some advantages:

- You allow the user to choose from a number of pre-arranged options
- Typing numbers eliminates spelling and other user input errors
- It's possible to set up submenus to manage a lot of options
- The command is accepted as a string, so no input will break the menu
- Asking for numeric values rather than words or characters simplifies things as there's no

uppercase '3.'

You'll be writing a very simple menu for this program, but you'll see the concept used a lot. For now, your menu has just two choices. You'll need to print out the menu, get the user's choice in a variable, and decide what to do with the user's input. Note that it's possible (likely) that the user will type in something you don't want, so you'll need to account for that. Eventually we will have the menu repeat until the user chooses an 'exit' option, but that can wait for another week. As usual, you'll need your documentation file to explain what each step of the menu does. You don't need this in code, just clear enough English that you'll be able to translate to code when you need to.

Planning the eight ball

The thing that's going to act as an eight ball in our program is a new type of data called a 'list.' Be sure you've reviewed the materials about how to use lists, because that's going to be the key to this program (and most games, honestly.) Here's some things to think about our list:

- It needs to have at least eight fortunes, but it can have more
- Some should be negative, some positive, and some neutral
- You can use my examples if you have no imagination, but I'm sure you can come up with something better
- As usual, we want to be classroom-appropriate. Nothing that could make a classmate uncomfortable.
- All these fortunes will be strings that will be stored in a list variable
- Probably the list should be created early in the program as you'll be using it quite a bit.

The first option will let you see all of the possible fortunes.

- It will loop through the entire list and print each option number and value.
- You might need to review how for loops work with lists
- there are a few ways to do this in Python.
- You might also look at ways to format the text so it looks good

The next option is meant for debugging. If you give the computer a value between 0 and 7 it will return the corresponding element in the list.

- This will be useful to make sure that the list is working correctly.
- Note that when you ask a user for a number, it will come in as a string
- You'll need to convert it to a numeric value to use it for lookup
- Use the lookup operation to retrieve and print the requested fortune

The final option is more like a traditional eight ball:

- Ask the user to input a question

- In the spirit of highly accurate fortune telling, the question doesn't matter at all. We don't even care what it is.
- Pick a random number in the correct range. You may need to look up a function in the random library to do this
- Choose the fortune corresponding to that number
- Print out the fortune

As always, plan out the code as pseudocode first. Make sure you know what you want to do. Once you have that down (and can explain it all to your partner) you can begin to implement it in code. Since there are four distinct problems to solve, try to solve each problem independently before moving on.

Testing

For such a seemingly simple program, a lot can go wrong here. Part of the art form is to figure out what's happening when your program does not work correctly. Here's a few things to watch out for:

- There are a few ways to use the for loop in Python.
- Think about which is the best for this situation (you need the index and the value)
- The random function you'll need is in a library called 'random'
- Make sure you look at that library and know which function does what you want
- Maybe you should carefully decide what you do want.
- There's a number of ways to do this, but I'm looking for a numeric value to use as a list index.
- (while there is a way to randomly choose an element from an array, many languages do not have this, so I want you to start with a technique that will work in all languages)
- Be sure you've imported the library correctly and used the name of the function correctly (there are a number of ways to do this.)
- If you try to look up the element of an array with a string value, Python will complain. Your indices need to be ints, and you might need to do a conversion to get there.
- If you have a list with 7 elements and you ask for element 14, you will get a well-deserved disapproval from Python. Indices need to be integers (at least for now.)
- Indentation, spelling, and consistency still matter. Be sure you're careful. Sloppy typing has killed many otherwise fine programs.

Turning it in

As usual, you will be submitting a documentation file and at least one Python program. Your program should be called `<username>_eightball.py`. Documentation should be `<username>_eightball_docs.txt`, and the blackbelt project (if you do one) should be `<username>_eightball_blackbelt.py`

Have a good time!

General Assignment rubric			
Criteria	Ratings		Pts
<p>Algorithm and pseudocode</p> <p>Describe clearly the steps needed to solve the problem. Best solution is in a separate file, with each concept broken into small enough steps they can be implemented in a single line of code. Later in the semester, functions and classes should have their own algorithms. Can be in a text or readme file. For some projects, a diagram may also be useful or necessary.</p>	50 pts Full Marks	0 pts No Marks	50 pts
<p>Follows the guidelines</p> <p>Every assignment has specific guidelines listed in canvas. Full points will be awarded for following these guidelines. You will earn fewer points if the guidelines are not completely met.</p>	20 pts Full Marks	0 pts No Marks	20 pts
<p>Code runs as expected</p> <p>The best solution has all code working without modification. This involves eliminating syntax and logic errors, as well as being thoughtful on file naming conventions, documentation, and including all required dependencies.</p>	20 pts Full Marks	0 pts No Marks	20 pts
<p>Personal Style</p> <p>A great program shows creativity and flair. Did you make some efforts on personalizing the project to make it uniquely your own?</p>	5 pts Full Marks	0 pts No Marks	5 pts
<p>Pushing the envelope</p> <p>Once you've got everything working, what did you do to push beyond the basics? This can be a blackbelt extension, or it can be some kind of extra effort implementing a new feature. Make sure the effort you are making here is clearly marked in your documentation so we can see it.</p>	5 pts Full Marks	0 pts No Marks	5 pts
Total Points: 100			