WHO WANTS TO BE A MILLIONAIRE PORTFOLIO PROJECT REPORT

1)This is a Who Wants to be a Millionaire Game with two life lines.

Phone a friend and 50/50. It also has the ability to walk away.

The User starts at a $100 question and works his/her way up to 1 million.

The User can use 2 life-lines only once each. If the user gets 1 wrong answer the game ends and just like the real game she gets the base amount $0,$1000 or $32000.However,if the user chooses to walk away, then he walks away with exactly what the user earned.

2) For the questions, I created a question class which contains arguments for the main question, option a, option b, option c, option d, the answer, and the phone a friend suggestion

```
class Question:
    def __init__(self, mainQuest, optiona, optionb,optionc,optiond,right,phone):
        self.mainQuest = mainQuest
        self.option1 = optiona
        self.option2 = optionb
        self.option3 = optionc
        self.option4 = optiond
        self.right = right
        self.phone = phone
```

-Then I instantiated 15 question objects, passing in different questions from a dictionary, then manually passing in all of the corresponding options for the question and the phone a friend suggestions.

```
Question_1 = Question(question_list['question1'],'Pacific Ocean  ','Atlantic Ocean ','Indian Ocean   ','Arctic Ocean   ','a','a')
Question_2 = Question(question_list['question2'],'Barack Obama   ','George Bush    ','Donald Trump   ','Bill Clinton   ','c','c')
Question_3 = Question(question_list['question3'],'Las Vegas      ','Los Angelos    ','Califonia      ','Gotham City    ','d','d')
Question_4 = Question(question_list['question4'],'Hops           ','Yeast          ','Malt           ','Vinegar        ','d','d')
Question_5 = Question(question_list['question5'],'Liquids        ','Gases          ','Solids         ','None           ','c','c')
Question_6 = Question(question_list['question6'],'Connecticut    ','New Jersey     ','Rhode Island   ','Maryland       ','b','b')
Question_7 = Question(question_list['question7'],'Germany        ','United States  ','France         ','Canada         ','a','a')
Question_8 = Question(question_list['question8'],'25 km/hr       ','50 km/hr       ','65 km/hr       ','165 km/hr      ','c','c')
Question_9 = Question(question_list['question9'],'Stationary     ','In motion      ','Rising         ','falling        ','c','d')
Question_10= Question(question_list['question10'],'Jamaica       ','Cuba           ','Puerto Rico    ','Hispanola      ','b','b')
Question_11= Question(question_list['question11'],'Colombia      ','Brazil         ','Venezuela      ','Ecuador        ','c','a')
Question_12= Question(question_list['question12'],'Illinois      ','New Hampshire  ','Montana        ','Alaska         ','b','b')
Question_13= Question(question_list['question13'],'10 percent    ','20 percent     ','35 percent     ','50 percent     ','b','a')
Question_14= Question(question_list['question14'],'Morocco       ','Greece         ','Israel         ','Hungary        ','d','d')
Question_15= Question(question_list['question15'],'Cytoplasm     ','Nucleus        ','Chloroplast    ','None           ','a','d')
#player2 = Question('Aaron,1200',1300)
```

-Then I created a for loop that iterates from 1-17, this is to iterate through all of the questions.

In the first iteration of the loop or when x is equal to one and the user's earnings is 0

open the questions module and load the question 1 function.

-in the question_1 function pass in an options list, the question object that contains our unique question, the question_index so we know which question that we're on and worth to signify how much each question is worth but not how much the user has earned.

```
for x in range(1, 17):
    #area_func = globals()[question_index + str(x)]
    #area_func = globals()[question_index + str(x)]
    #area_func()
    if x == 1 and options[0] == 0:
        options = questions.question_1(options,Question_1,question_index,worth)
        if options[3] == True:
            print('Your have earned: $' + str(float(question_index[worth])))
            worth +=1
```

-all questions are in 1 function instead of having a function for every question

-the user is now shown choices and if she chooses the same string as in the right answer argument for that object, the options list at index 0 is incremented by 1 and returned.

```
def question_1(options,Question_,question_index,worth):
    print("  === Who Wants to be a Millionaire ===   ")
    print('$' + str(float(question_index[worth])) + ' Question')
    print(Question_.mainQuest)
    print("-----------------------------------------")
    print("| A-"+ Question_.option1 +"   | B-"+Question_.option2+"|")
    print("-----------------------------------------")
    print("-----------------------------------------")
    print("| C-"+ Question_.option3 +"   | D-"+Question_.option4+"|")
    print("-----------------------------------------")
    print("|           E-Phone A Friend            |")
    print("-----------------------------------------")
    print("-----------------------------------------")
    print("|           F-    50/50                 |")
    print("-----------------------------------------")
    print("|           G-  Walk Away               |")
    print("-----------------------------------------")
```

-in the next iteration the program checks if the question_index list is pointing to the next value which is 100 in the List which signifies that the user has earned $100 and is eligible for a $200 question.

This process repeats as long as the User keeps answering the right questions, otherwise the user

Doesn't earn enough to qualify into the next question and the game ends.

```
elif x == 2 and question_index[options[0]] == 100:
    options = questions.question_1(options,Question_2,question_index,worth)
```

- Details of how you converted from design to the actual realization of your project, in terms of implementing the code.

-I didn't do a design I just had a thought about what the final product should look like then I implemented procedural design until I learned about OOP, the I upgraded the game to include

More OOP features

- Any choices that you made, and any modifications that you made to the design, in response to difficulties that you might have encountered while implementing the project.

-50/50 I need to make more random. Based on the answer to the particular question, the user loses

Certain options always, which works but just not optimal yet.

3)

- Discuss what you personally learned from your project.

-I learned that I must not get too frustrated when I encounter a problem but spend a lot of time

On it because there is an answer

- Discuss the best features and the shortcomings of the project.

The best feature in my opinion is the questions class and the question function because it saved

Me from reprinting each question over and over.

- Discuss any choices that you might have made differently, in hindsight after completing the project.

I wouldn't include as much if and elif's but its hard to get around that.

- Describe any additional features you may want to add in the future.

I would like to add a lot more questions that pertain to the dollar amount. For example five $100 questions five $200 questions etc. then randomize them all.