Quiz 2 Question Bank

Computational Thinking Fall 2019

1. Create a python class called retirement\_fund that initializes

2 variables – a float self.balance variable that starts at 0 and an empty list self.deposits in the def \_\_init\_\_(self).

Then, make 2 additional functions within the class:

def deposit(self,amount):

def get\_deposits(self):

deposit function will

1. add to self.balance variable the given amount
2. append the deposit amount to self.deposits list
3. return an f-string “Your current fund balance is ..”

get\_deposits function will return the list of all deposits made into your retirement fund.

Sample code in main program:

my\_fund = retirement\_fund()

print(my\_fund.deposit(1000))

print(my\_fund.deposit(4000))

print(my\_fund.get\_deposits())

Sample output in main program:

Your current fund balance is $1000.

Your current fund balance is $5000.

[1000, 4000]

1. Develop a python **function** called grade\_dict that takes as an input parameter a list of letter grades (e.g., ‘A’, ‘B’, ‘C’, ‘D’, ‘F’) called grade\_list and outputs the total number of students for each letter grade.

In your function:

1. Make a blank dictionary
2. For each key in your dictionary (‘A’, ‘B’, ‘C’, ‘D’, ‘F’) set the initial value to 0.
3. Loop through the grade\_list and update the counts for each grade in grade\_dict as needed (‘A’ >= 90, ‘B’ >=80, ‘C’ >= 70, ‘D’ >= 60, F < 60 )
4. return the grade\_dict

Sample code from main program:

grades = [100,90,83,59,47,72]

grade\_dictionary = grade\_dict(grades)

print(grade\_dictionary)

Sample output:

{'A': 2, 'B': 1, 'C': 1, 'D': 0, 'F': 2}

1. Create a python class frequent\_flyer that keeps track of the flights and miles for a frequent flyer of an airline. Your class should have three functions:
2. def \_\_init\_\_ (self, name)

Set the self.name variable to the input name entered.

Set the self.flights variable to empty list.

Set the self.miles variable to 0.

1. A flight function that has self, origin, destination, miles as input parameters.

Append an (origin, destination) tuple to your self.flights list. self.flights is a list of tuples.

Add your miles from the flight to your self.miles variable.

1. A get\_miles\_and\_flights function that has self as an input parameter and returns an f-string that gives the total miles and past flights of the passenger (See sample output below).

Sample code from main program:

capt\_sully = frequent\_flyer('Sully')

capt\_sully.flight("CID","JFK",1000)

capt\_sully.flight("JFK","ORD",800)

print(capt\_sully.get\_miles\_and\_flights())

Sample output:

Sully has 1800 miles on flights: [('CID', 'JFK'), ('JFK', 'ORD')]

D) Create a Python class called Point class. Your class should have three functions/methods:

1) def \_\_init\_\_ (self, x, y) that initializes self.x and self.y to the input parameters x and y that are initial coordinates for the point.

2) A method show that returns the coordinates of the point as a string

3) A method move that changes the coordinates. For example,

move(10, -10) will add 10 to self.x subtract 10 from self.x.

Sample input and output:

p1 = Point(2, 3)

p2 = Point(3, 3)

print(p1.show())

(2, 3)

print(p2.show())

(3, 3)

p1.move(10, -10)

print(p1.show())

(12, -7)

print(p2.show())

(3, 3)

E) Write a Python function named median\_odd( example\_list ) that takes an unsorted example\_list as an input parameter and returns the median.

* Set a variable list\_len = length of the example\_list
* To calculate median:
  + Sort the list
  + Hint: Use %2 and == to test if list\_len is even or odd
  + If the length of the list is odd, the median can be found by getting the index of the middle number of the sorted list.

If example\_list = [1,2,3], you can see that

**median** = example\_list[int( 3 / 2 – 0.5)] = example\_list[1] = **2**

* + If the length of the list is even, set median to an error message string similar to the output below.

Sample input in main program:

median1 = median\_odd( [0,-5,13,17,-22] )

print(median1)

median2 = median\_odd( [1,2,3,4] )

print(median2)

Sample output:

0

The function only works for odd-length lists.

F) Write a Python class named Rectangle constructed by a length and a width and two methods which will compute and return the area and the perimeter of the rectangle. Your class should have 3 methods/functions:

1) def \_\_init\_\_ (self, x, y) that initializes self.length and self.width to the input parameters x and y respectively.

2) A method get\_area that returns the area of the rectangle object.

3) A method get\_perimeter that returns the perimeter of the rectangle object.

Hint – area of a rectangle is its length multiplied by its width

- perimeter of a rectangle is two times its length plus two times its width

Sample input in main program:

newRectangle = Rectangle(12, 10)

print(newRectangle.get\_area())

print(newRectangle.get\_perimeter())

Sample output:

120

44

G) Write a Python class named Circle constructed by a radius and two methods that will return the area and circumference of the circle. Your class should have three methods:

1) def \_\_init\_\_ (self, radius) that initializes self.radius with the input parameter radius.

2) A method get\_area that returns the area of the rectangle object.

3) A method get\_circumference that returns the circumference of the circle object.

You may use a variable pi = 3.14 as a variable substitute for pi

Hint: The area of a circle is pi multiplied by the square of the radius

The circumference of a circle is 2 multiplied by pi multiplied by radius

Finally, using an f-string, print out a sentence including the area and circumference as shown below.

Sample input in Main Program:

new\_circle = Circle(10)

area = new\_circle.get\_area( )

circumference = new\_circle.get\_circumference( )

print( f-string using area and circumference)

Sample output:

This circle has an area of 314 and a circumference of 62.