Practice 10 [Class – Objects – Functions – Operators]

Note: This is not a routine practice; this is also your assignment 1. The viva of these tasks will be taken in next lab.

1. Consider Point class, discuss and shared in class room and add following functionality:

a. reverse the point by changing signs of data members. For example, (2, 5) results in (-2, -5), (3, -4) results in (-3, 4)
b. check and return quadrant, where point lies. In first quadrant both x & y are positive, in second quadrant x is negative, in third quadrant both are negative and in fourth quadrant y is negative
c. rotate function to rotate point at given angle (angle is the parameter of the rotate function). A small program is given below, how sin & cos functions work. The rotation formula is:
   - x' = x * cos theta + y sin theta
   - y' = - x * sin theta + y cos theta

   The function is to create a new point with x & y coordinator (0, 0). On write side of the equation write data members of new point; whereas, on the left side use data member of current object. Lastly, note that your answer should be in integers

```
import math

def main():
    angle = int(input("Enter Angle:"))
    print(math.sin(angle*math.pi/180))
    print(math.cos(angle*math.pi/180))
```

d. `__str__` function to represent object in string like (3, 5)
e. function compare_rotation: The difference between float values after rotation and integer values after rotation. Suppose x_int, y_int are integer values after rotation and x_float, y_float is float values after rotation, the difference is:

$$d = \sqrt{(x\_float - x\_int)^2 + (y\_float - y\_int)^2}$$

Write main function and test your class functions. Especially, find if a point (5,0) is rotated at 30 degrees 12 times, what is the total difference that will occur, if we will keep values in float instead of int.

2. Consider Set class discussed in class and shared in google class room and add following functionality:

a. is_subset, check whether element all the elements of current object exist in second set. If yes, return true, otherwise return false
b. is_superset, check whether all the elements of second set exist in current set. If yes, return true, otherwise return false
c. compare two sets and return true, if both elements have same elements, where order can be different. Here, you need to check turn by turn (not together) elements of first set in second set, then element of second set in first set. If any element of set not found in other set, return false.
d. count_difference, find number of elements of current object not exist in second set

Write main function and test your class functions.

3. Write a class Element. Element has five private data members [value, hour, minute, second & count]. In **initializer** function pass only one parameter the "value". The initializer function will call function "now" (a small code is given to use "datetime") and set the value of hour, minute, & second and initialize count by 0. The idea is whenever, value is accessed (for any purpose) count is added by 1 and time is checked and updated by current time. Add following functions:

a. `__str__` function to return string having value followed by hyphen followed by hour, minute & seconds separated by colon and finally count in parenthesis, like **234 – 10:45:12 (3).** Here, 234 is value, which is last accessed at 10:45:12. This value is accessed 3 times. Note, this function will not change the time and count
b. set function to set the new value (update count and time)
c. get_value to get the value (update count and time)
d. get_count to get the count (without changing time and count)

e. get_hour to get the hour (without changing time and count)
f. print time, print hour:minute:seconds (without changing time and count)
g. overload "==" operator to return true if values are equal in both operands, ignore other data members (update time and count of both objects)
h. overload ">" operator to return true if value of current object is greater than value of second object, ignore other data members (update time and count of both objects)
i. overload "<" operator to return true if value of current object is smaller than value of second object, ignore other data members (update time and count of both objects)
j. increment_by, pass n and add n to value. (update count and time)
k. is_older, pass another object and compare time, if current object has time in past as compare to second object, return true, otherwise return false
l. is_equal_time, return true, if both objects has same hour, minute & seconds, otherwise return false
m. is_more_frequent, return true, if current object has count greater than the second object's count

Finally, write main function and test these functions, especially check time & count after each function.

```
from datetime import datetime


def main():
    time = datetime.now()
    print(f'Current time is:{time.hour}:{time.minute}:{time.second}')
```