SI@UCF Introduction to Programming in Python Test #1 Solutions
6/14/2022

1) (6 pts) Write a **single statement** in python the produces the following output:

```
Backslash:\\
Newline:\n
"He said, 'yes!', after waking up."
```

```
print("Backslash:\\\\\nNewline:\\n\n\"He said, 'yes!',after waking up.\"")
```

Grading: 1 pt print, 1 pt regular chars, 2 pts escape \\, 1 pt escape \n, 1 pt escape either " or '

2) (10 pts) What are the values of the following expressions in Python?

a) `3 + 4*8` **35**

b) `515%300` **215**

c) `(16 - 2*4)//3` **2**

d) `3 + 38/10 - (72%15)%7` **1.8**

e) `3*(18 - 16//(2 + 2**2))` **48**

Grading: 2 pts per answer, all or nothing.

3) (12 pts) Air Hockey costs 50 cents to play. But, if you beat Kyle, he'll pay your cost so it's free to you. Complete the program below so that it reads in how many times you played Kyle and how many times you won, and prints out how much you spent playing. (For example, if you played 10 times and won 3 times, you would pay $3.50 for the 7 times you lost.)

```
GAMECOST = 0.50
numPlayed = int(input("How many times did you play Kyle?\n"))
numWon = float(input("How many times did you win?\n"))

cost = GAMECOST*(numPlayed-numWon)

print("You paid $",cost, " for playing foosball.", sep="")
```

Grading: 4 pts correct subtraction, 2 pts mult, 2 pts mult by .5
4 pts printing correct item.

4) (12 pts) Ice cream at Toppers has a base price of $4.00. Each topping added costs an additional 50 cents. But, if you get 7 or more toppings, the cost is just $3.00 for all of the toppings you get. Complete the program below so that it asks the user how many toppings they are getting and prints out how much their ice cream will cost.

```
BASE_COST = 4
TOP_COST = 0.50
toppings = int(input("How many toppings do you want?\n"))


MAX_TOP_COST = 3.00
cost = 0

if toppings < 7:
    cost = BASE_COST + toppings*TOP_COST
else:
    cost = BASE_COST + MAX_TOP_COST

print("Your order will cost $"+str(cost))
```

```
Grading: 2 pts any if,
         2 pts separate by correct Boolean exp
         3 pts for <7 formula,
         3 pts for >= 7 formula
         2 pts print
```

5) (15 pts) Complete the program below so that it prints out all of the integers in between start and end, **inclusive**, that are divisible by 2, 3 and 5.

```
start = int(input("What is the starting number?\n"))
end = int(input("What is the ending number?\n"))

for i in range(start, end+1):
    if i%2 == 0 and i%3 == 0 and i%5 == 0:
        print(i, end=" ")
print()
```

```
Grading: 5 pts loop (give 4 pts if off by 1)
         6 pts if check (give partial as needed)
         4 pts print – no need to advance to next line.
```

6) (10 pts) What is the output of the following Python program?

```
a = 2
b = 5
for i in range(5):
    c = b + a*2
    print(a,end=" ")
    a = b
    b = c
```

2 5 9 19 37

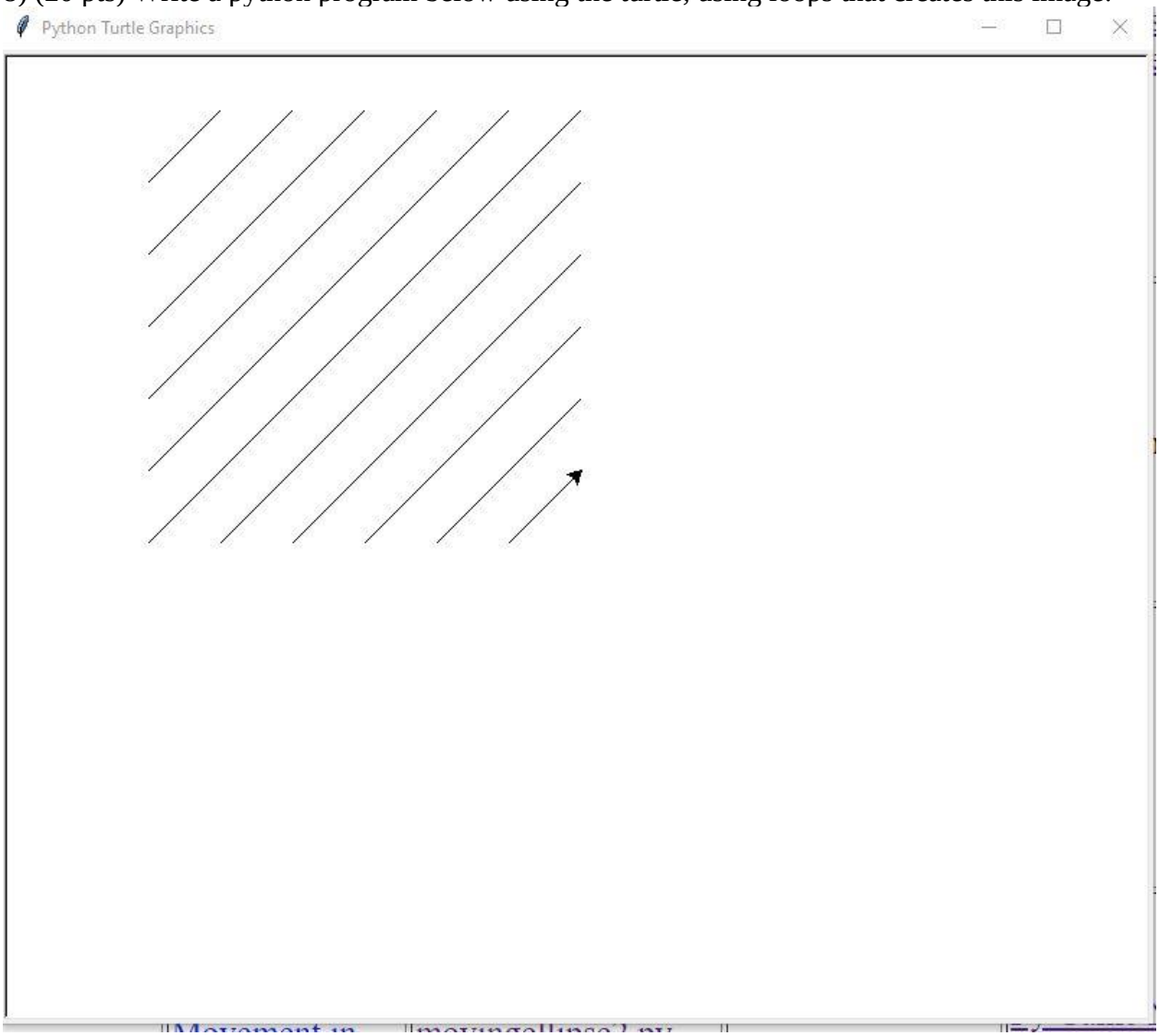
Grading: 2 pts per each one, all or nothing.

7) (12 pts) Recall that the function random.randint(a,b) returns a random integer in between a and b, inclusive. Write a program that reads in from the user the number of sides on the first die, the number of sides on the second die, and the number of times both dice will be rolled. (Assume the labels on the sides of the die are the first n positive integers, where the die has n sides.) Simulate rolling the dice the appropriate number of times and count how many of those times result in the second die showing a number strictly greater than the first die. Then, print both this value and the **percentage** of the trials where the second die showed a number greater than the first die did.

```
import random

n1 = int(input("How many sides on the first die?\n"))
n2 = int(input("How many sides on the second die?\n"))
trials = int(input("How times do you want to roll the dice?\n"))

success = 0

for i in range(trials):
    die1 = random.randint(1, n1)
    die2 = random.randint(1, n2)

    if die2 > die1:
        success += 1

print("Number of successes =", success)
print("Percentage of success =", success/trials*100)
```

Grading: 1 pt accumulator, 1 pts loop, 2 pts die1, 2 pts die2,
2 pts if, 1 pt add accumulator, 1 print success,
2 pts print percentage

8) (20 pts) Write a python program below using the turtle, using loops that creates this image:



Specifically, each line segment has slope 1. The top left segment connects the coordinates (-300, 250) with (-250, 300). The one directly below it connects (-300, 200) to (-200, 300), and so forth. The longest segment connects (-300, 0) to (0, 300). Thus, the picture is entirely within the second quadrant. The first line "below" the longest segment connects (-250, 0) to (0, 250), and so forth, with the bottom most segment connecting (-50, 0) to (0, 50).

Please write your solution on the following page. **In order to receive significant credit, you must use loops (either one or two) to solve the problem. A solution which separately calls forward 11 times to draw the 11 separate lines will receive 4 points out of 20, at most.**

Note: This problem is challenging, so it's okay if you don't get it!

```
import turtle

for y in range(250, -50, -50):
    turtle.penup()
    turtle.setpos(-300, y)
    turtle.pendown()

    turtle.setheading(45)
    turtle.forward( (300-y)*(2**.5) )

for x in range(-250, 0, 50):

    turtle.penup()
    turtle.setpos(x, 0)
    turtle.pendown()

    turtle.setheading(45)
    turtle.forward(abs(x)*(2**.5))
```

Grading: Many ways to do this. 10 pts for each "half" of the design. Points out of 10 as follows:

3 pts – loop that runs 5 or 6 times
3 pts – calculations for either x or y that fit the correct pattern.
2 pts – moving to the right starting spot without drawing
2 pts – drawing the right distance (1 pt if close)

9) (3 pts) What type of food can you find at the UCF restaurant, Burger U?

Burgers – Give to All