CS5001: Lab 6. Due on Friday, Oct-20-2023.

Name(s): Xujia Qin

Email(s): qin.xuj@northeastern.edu

You can work on this lab either individually or in small group of two or three students. If working in a group, include names of all the students in the submission PDF.

Getting credit for this lab. This lab handout has several empty boxes that prompt you to answer a question. As part of the lab, you are to write the answers to these questions inside the boxes/blanks. When you are finished, you should create a PDF and upload it on Canvas. If you don't finish, you have until 11:59 PM on Friday, Oct-20-2023 to submit.

What computer to use? If your primary computer is a laptop, bring it to the lab to work on, as lab is an excellent opportunity to get started with Python on your machine. You should follow the instructions on the course website. Ask a TA for help if you have problems with your installation. If you prefer, you could also use one of the machines in the lab room to work on this lab assignment.

Lab Materials. Lab materials can always be found on Canvas under the appropriate lab posting.

For today's lab, you need this handout (which is also online), and the file ShowRand.py (on Canvas).

1 Random simulation basics

```
# ShowRand.py
# CS5001
# Feb, 2023

from random import randint as randi
from random import uniform as randu

N = 100000
count=0
for k in range(N):
    x = randu(0,100)
    if 10<=x<=30 or 40<=x<=70:
        count +=1
print (float(count)/float(N))

count=0
for k in range(N):
    x = randu(0,100)
    if 10<=x<=50 and  20<=x<=60:
        count +=1
print (float(count)/float(N))

count=0
for k in range(N):
    x = randi(1,20)
    if x%2==1:
        count +=1
print (float(count)/float(N))

count=0
for k in range(N):
    x = randi(1,20)
    if 13<=x<15:
        count +=1
print (float(count)/float(N))
```

The above script produces 4 lines of output. Fill in the following table: (**40 points, 5 points each blank,** no credit given for empty boxes)

		I Think it's This	Python Says	Notes
1	First Output Line	0.5	0.5012	because the random module is utilized, we can not derive an accurate possibility for a limited 100000-time trials.
2	Second Output Line	0.3	0.29848	
3	Third Output Line	0.5	0.49989	
4	Fourth Output Line	0.1	0.09932	

2 Random Walk

Here is code that simulates a random walk inside a square with corners (L,L), (L, −L), (−L,L), and (−L, −L):

```
x = 0
y = 0
T = ''
while abs(x) < L and abs(y) < L:
    # The robot has not reached the edge
    r = randi(1,4)
    if r==1:
      # Hop North
      y = y + 1; T = T + 'N'
    elif r==2:
      # Hop East
      x = x + 1; T = T + 'E'
    elif r==3:
      # Hop South
      y = y-1; T = T + 'S'
    else:
      # Hop West
      x = x-1; T = T + 'W'
# Code Here
```

A. The robot starts at (0,0) and hops its way to the edge of the square. A hop is either one unit North, one unit East, one unit South, or one unit West. The string T "records" the itinerary. An example T would be 'NENSWSWE'. Assume that L is a positive integer and that T has been produced by the while loop. Add code after the "Code Here" comment so that the number of hops that the robot needed to reach the edge is assigned to a variable nSteps. **(20 points)**

```
from random import randint as randi
x = 0
y = 0
T = ""
L = 5 # test example
nSteps = 0 #initialize the counter variable
while abs(x) < L and abs(y) < L:
    r = randi(1,4)
    if(r == 1):
        y = y+1; T = T + "N"
    elif(r == 2):
        x = x+1; T = T + "E"
    elif(r == 3):
        y = y -1; T = T + "S"
    else:
        x = -1;T = T + "W"
    nSteps = nSteps + 1 #increment nSteps by 1 after moving one step
```

B. Add code after the Code Here comment so that the x-y coordinates of the robot's final position are printed out. (Hint. Think about T.count('N'), T.count('E'), T.count('S'), and T.count('W'). **(20 points)**

```
from random import randint as randi
x = 0
y = 0
T = ""
L = 5 # test example
nSteps = 0 #initialize the counter variable
while abs(x) < L and abs(y) < L:
    r = randi(1,4)
    if(r == 1):
        y = y+1; T = T + "N"
    elif(r == 2):
        x = x+1; T = T + "E"
    elif(r == 3):
        y = y -1; T = T + "S"
    else:
        x = -1; T = T + "W"
    nSteps = nSteps + 1 #increment nSteps by 1 after moving one step

x_coordinate = T.count('E') - T.count('W') # Calculate the final position based on the counts of each direction
y_coordinate = T.count('N') - T.count('S')
print("x-y coordinates of the robot's final position: x =", x_coordinate, ", y =", y_coordinate)
```

C. As it stands the simulation has the robot hopping until it reaches an edge of the square. When the simulation ends, what can you say about the values of abs(x), abs(y) if we change the while-loop condition to abs(x) < L or abs(y) < L? **(20 points)**

Changing the loop condition to abs(x) < L or abs(y) < L will make the robot once it reaches any edge of the square.
In detail, when either x or y, not both, exceeds or equal to the L value, the loop will stop.
The final values of abs(x) and abs(y) will indicate how far the robot is from the nearest edge in vertical or horizontal direction.