Assignment 2

Introduction to Software Systems, Spring 2024

Due: 24^{rd} Feb, 2024

Task 1

a) Pseudo Random Number Generator

How can deterministic computers produce random numbers? And does it even need to? Turns out Classical Computers unlike Quantum Computer don't have access to 'Truly random numbers'. Computers need to access random numbers in order to encrypt information, perform computer simulation, do ray tracing in Computer Games, make Machine Learning happen and so much more.

What we mean by random numbers here is a sequence of independent random numbers with a specified distribution. In this subtask, you will be implementing the famous 'Middle square algorithm' by John von Neumann to generate pseudo-random numbers. Take a 4 digit number called the **seed**, say 9876, now square it and extract the middle 4 digits, thus $9{,}75{,}35{,}376 -> 5353$. And we repeat this process i.e. now take this as the four-digit number, square it and extract the middle 4 digits and so on. We highly recommended you to watch this video to get an intuition about what pseudo-random numbers are.

Write a Python function `pseudo_rand_num_gen(seed, k)` that generates k pseudo random numbers using the above process and uses your current system time as the seed to this function. The function returns a list containing all the k random numbers generated.

Update: The middle-square algorithm needs to be implemented for a general seed with even number of digits. (Hint: A faster way to extract the middle digits is by converting the number to a string first.) In case the seed after squaring has odd no. of digits pad it with a zero in the front and then extract the middle digits.

Also plot a histogram of the generated random values (after dividing by 10^n where n is the no. of digits in the seed) using matplotlib.

b) Estimating π

Imagine a Cartesian coordinate plane and a square of side length 2 centered at the origin. The square will have vertices at (-1, -1), (-1, 1), (1, 1), and (1, -1). Inside the square, a circle of radius 1 unit centered at origin as shown below.

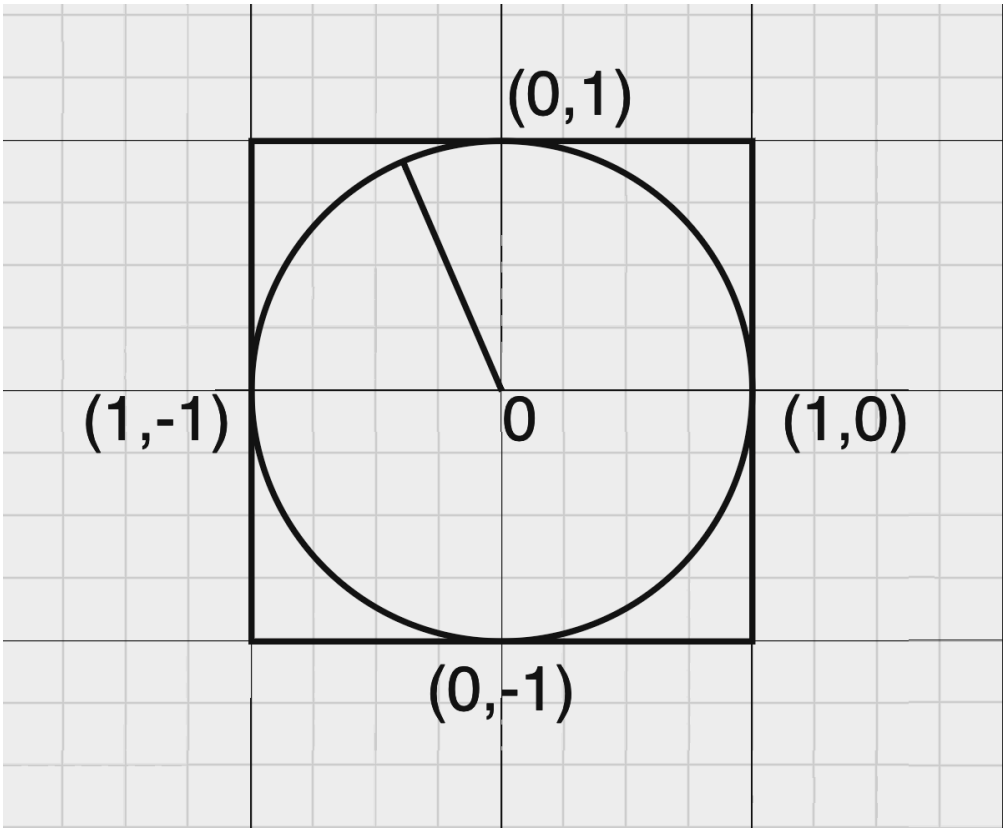

Figure 1: Monte-Carlo Simulation

Now, we randomly generate a large number of points (x, y) within the square. The x and y coordinates of these points should be between -1 and 1. For each generated point, we check if the point falls within the circle ($x^2 + y^2 \leq 1$) and keep a count of no. of points falling inside the circle and total no. of random points generated.

Now the P(random point lie in the circle) $= \frac{\text{Are of the circle}}{\text{Area of the square}} = \frac{\pi}{4}$. We empirically estimate this probability by random sampling done above. Therefore, $\pi = 4 \times \frac{\text{Are of the circle}}{\text{Area of the square}}$.

Update: Use Python to simulate the above process and compare the π value obtained using the pseudo-random number generator from the previous subtask and `random.random()` function from `random` module in python.

Note: You must import the `pseudo_rand_num_gen(seed, k)` function from part (a) .py file, instead of copy-pasting the function code.

Bonus: Calculate and plot the estimated error value in calculation of π for each iteration. You are allowed to use matplotlib library for plotting purposes.

Task 2

Check out the Jupyter notebook uploaded for this Task. Write your code in that Jupyter Notebook and submit that notebook for this task.

Task 3

In this task, you will be learning how to scrape website (extracting data from websites and cleaning the scrapped Data) and data visualization using matplotlib.

a) Write a Python script to scrape the Top 250 TV-shows of all time from the IMDB website. After scraping the data, save it to a MySQL database named 'top-250-shows' for further analysis. You must also use the data from the obtained database to plot the following graphs:

i. A bar graph representing Genre (on x-axis) to no. of TV-shows belonging to that genre (on y-axis). (Note: A TV Show might have multiple genre)

ii. A line graph representing the frequency count of TV-shows having n episodes, n varies from 1 to maximum no. of episodes present. Represent no. of episodes (on x-axis) and frequency count (on y-axis).

b) Write a Python Program that allows user to filter the TV Shows based on:-

1. Genre
2. IMDB rating
3. No. of episode

For each filter take user-input to choose the criteria. The user must be prompted a range (inclusive of both the limits) for IMDB rating and No. of episode and Genre must be a string input consisting of genres separated by spaces. Print the TV-show in the descending order based on the user-filtering.

```
>>./q3_b.py
Comdey Thriller Drama Documentary
8.5 9.5
10 20
```

Reference: Helpful reference for learning about web-scrapping using Python.

Bonus: Create a word cloud of the above filtered TV Show list. The relative size of the word in the word cloud will depend on the above priority order i.e. TV Show name on top of the list should be printed with larger font-size.

”

Figure 2: Word Cloud

Submission Format

```
rollnumber.zip
├── q1
│   ├── q1_a.py
│   ├── q1_b.py
│   └── q1_bonus.py
├── q2
│   ├── q2.ipynb
│   └── any images
└── q3
    ├── q3_a.py
    ├── q3_b.py
    └── q3_bonus.py
```

Please submit a zip as per the above submission format on moodle.