Random Walk

Read the instructions carefully. Not following the instructions will result in you not getting the credit you want for the assignment.

Learning Outcomes

1. Use random choices to simulate random behavior in an experiment
2. Group data in an appropriate collection data structure such as a dictionary or list
3. Use Python matplotlib to put visual information on a graph and save it to a file
4. Separate generating numerical results from visualizing results
5. Use loops to avoid repeated code in your solution when working with dictionaries and lists

Background

In 1827, the Scottish botanist Robert Brown observed that pollen particles suspended in water seemed to float around at random. He had no plausible explanation for what came to be known as Brownian motion, and made no attempt to model it mathematically. Louis Bachelier presented a clear mathematical model in his doctoral thesis, The Theory of Speculation in 1900. His thesis was largely ignored by respectable academics because it dealt with the then disreputable field of understanding financial markets. In 1905, Albert Einstein used similar stochastic thinking in physics to describe how it could be used to confirm the existence of atoms. People seemed to think that understanding physics was more important than making money, and the world started paying attention.

Brownian motion is an example of a random walk. Today, random walks are widely used to model physical processes like diffusion, biological processes like the kinetics of displacement of RNA from heteroduplexes by DNA, and social processes like movements of the stock market.

We are interested in random walks because of their wide applications to many problems, and for learning more about how to structure simulations nicely in Python.

Problem to Solve

Farmer John has three interesting pets: Chuck the chicken, Daisy the dog, and Chester the cat. Each of these animals has a different style of wandering around:

1. **Chuck the chicken** loves to wander randomly. Each second, he takes a step in a random direction: North, South, East, or West, with equal probabilities. Chuck's marker in a graph is a blue circle.

2. **Daisy the dog** loves to explore but she has a slight preference for the North. Each second, she takes a step with the following probabilities: North (50%), South (16.67%), East (16.67%), and West (16.67%). Daisy's marker in a graph is a red square.

3. **Chester the cat** is a peculiar feline who only moves East or West, never North or South. Chester's marker is a green triangle.

Your task is to simulate the movement of each animal for \(1000\) steps and plot their steps. Assume they always start at the barn door, which we will label \((0,0)\). In the file `random_walk.py` we give you code for a function `plot_walk(animal)` that you call to visualize an animal's path using `matplotlib`. Do not modify this function, just call it.

Add your implementation to the file `random_walk.py` in a directory called `walk`. Your program should produce 3 output files when run:

- Chuck_rw.png
- Daisy_rw.png
- Chester_rw.png

Below is shown example plots for each animal. Your plots should look similar to these, but are not expected to exactly match these ones. The location of \((0,0)\) is different in each plot and that's OK.

How to Test

- Run your program and verify that Chuck, Daisy, and Chester move differently, each as described.
- Each time you run the simulation, it produces randomly different output graphs/images for each animal.

Correctness

In your terminal, execute `ruff` to check your work's correctness.

```
ruff check random_walk.py
```

Style

Execute the below to evaluate the style of your code using `ruff`.

```
ruff format random_walk.py
```

How to Submit

From your Codespaces workspace in your assignment repository, when ready, submit your code by clicking Commit button and writing a meaningful commit message.

Frequently Asked Questions

Q: Why are we using matplotlib for this project?

A: Three main reasons: 1) it's easy to save images produced as files, 2) it produces publication quality graphs/charts and 3) you've already learned how to use it.

Q: What data type is the `animal` parameter in `plot_walk`?

A: It represents a single animal.

Q: Do I have to plot walks of any lengths or only \(1000\)?

A: Only \(1000\).

Q: Is there user input or command line input for this program?

A: No there is not. In fact, it would be incorrect and inconvenient to have that. `check50` will fail your code if you have user input to the program.

Q: There's more than one way to do this problem. Do I have to use your `plot_walk` function?

A: Yes, for two main reasons: 1) generating the data is the main point of this program, and 2) `matplotlib` is a well-known, time-honored module

Q: Should we plot lots of walks instead of just one walk?

A: That would be a natural next step, but we have to start somewhere.

Q: Isn't this assignment a piece of cake in the ChatGPT era?

A: If you REALLY want to be good, work for it.

Grading

Criterion	100 (Mastery)	85 (Proficient)	70 (Basic)	50 (Needs Improvement)
Chuck, Daisy, and Chester move differently, each as described	Each animal moves differently and accurately as described	Most animals move differently and accurately as described	Some animals move differently and accurately as described	Animals do not move differently or accurately as described

Criterion	100 (Mastery)	85 (Proficient)	70 (Basic)	50 (Needs Improvement)
Randomly different output graphs/images for each animal	Simulation produces randomly different output graphs/images for each animal every time	Simulation produces randomly different output graphs/images for most runs	Simulation produces randomly different output graphs/images for some runs	Simulation does not produce randomly different output graphs/images
Use of dictionaries or lists to group information	Dictionaries or lists are used effectively to group information about each animal	Dictionaries or lists are used but not always effectively	Dictionaries or lists are used inconsistently	Dictionaries or lists are not used
Only global code is information about each animal	The only global code is the information about each animal	Minimal global code; mostly information about each animal	Some global code; some information about each animal	Significant global code; not limited to information about each animal
Loops to avoid repeated code	Loops are used effectively to avoid repeated code	Loops are used but not always effectively	Loops are used inconsistently	Loops are not used
Main function with conditional execution of main()	Program has a main function with clear conditional execution of main()	Program has a main function with some conditional execution of main()	Program has a main function but lacks clear conditional execution of main()	Program does not have a main function with conditional execution of main()
One function one job rule	Each function performs a single, clear task	Most functions perform a single, clear task	Some functions perform a single, clear task	Functions perform multiple tasks
Using doctest snippets where it makes sense	Doctest snippets are used effectively where appropriate	Doctest snippets are used but not always effectively	Doctest snippets are used but inconsistently	Doctest snippets are not used
Passes `ruff check` for syntax errors, violations, and potential bugs	Code passes all `ruff check` for syntax errors, violations, and potential bugs	Code passes most `ruff check` for syntax errors, violations, and potential bugs	Code passes some `ruff check` for syntax errors, violations, and potential bugs	Code does not pass `ruff check` for syntax errors, violations, and potential bugs

Students should strive for mastery level. Lower than that indicates areas where more practice is needed or more learning is needed. Code score is the average of the individual feature scores.

Total score is 1/4 * style score + 3/4 * code_score.