CAREERFOUNDRY

Python for Web Developers Learning Journal

Objective

We find that the students who do particularly well in our courses are those who practice metacognition. Metacognition is the art of thinking about thinking; developing a deeper understanding of your own thought processes. With the help of this Learning Journal, you'll broaden your metacognitive knowledge and skills by reflecting on what you learn in this course.

Thanks to this Learning Journal, when you finish the course you'll have a complete and detailed record of your learning journey and progress over time. We really recommend that you take the time to complete this Journal; students do better in CF courses and in the working world as a result!

Directions

First complete the pre-work section before you start your course. Then, once you've begun learning, take time after each Exercise to return to this Journal and respond to the prompts.

There will be 3 to 5 prompts per Exercise, and we recommend spending about 10 to 15 minutes in total answering them. Don't overthink it—just write whatever comes to mind!

Also make sure that, once you've started filling this document in, you upload it as a deliverable on the platform. This is so that your mentor can also see your Journal and how you're progressing over time. Don't worry though—what you write here won't affect how you're graded for the Exercise tasks. The learning journal is mostly for you and your self-evaluation!

Pre-Work: Before You Start the Course

Reflection questions (to complete before your first mentor call)

1. What experiences have you had with coding and/or programming so far? What other experiences (programming-related or not) have you had that may help you as you progress through this course?

I completed the Career Foundry Intro and Immersion course in Full Stack Development. During the course, I gained experience with HTML, CSS, JavaScript, React, Angular, NoSQL (specifically mongoose), and SQL. Before that, I studied business administration, which included an introductory module in business informatics where I learned basic Python. Additionally, I attended a short university course focused on PHP and SQL.

2. What do you know about Python already? What do you want to know?

I'm familiar with basic Python concepts like if-then statements and loops. I want to learn how to apply Python in real-world projects and how to integrate it with databases.

3. What challenges do you think may come up while you take this course? What will help you face them? Think of specific spaces, people, and times of day of week that might be favorable to your facing challenges and growing. Plan for how to solve challenges that arise.

I anticipate challenges with exercises that involve more math. To tackle them, I plan to work on them during late morning hours (9-12), when I feel most energized and focused.

Remember, you can always refer to Exercise 1.4 of the Orientation course if you're not sure whom to reach out to for help and support.

Exercise 1.1: Getting Started with Python

Learning Goals

- Summarize the uses and benefits of Python for web development
- Prepare your developer environment for programming with Python

Reflection Questions

1. In your own words, what is the difference between frontend and backend web development? If you were hired to work on backend programming for a web application, what kinds of operations would you be working on?

Frontend development focuses on creating the user interface that users interact with directly, while backend development handles the hidden operations that support the frontend, like interacting with servers and databases, processing data, and managing requests. If hired for backend programming, I'd focus on tasks such as database management, server-side logic, API integration, and ensuring smooth data processing and communication.

2. Imagine you're working as a full-stack developer in the near future. Your team is asking for your advice on whether to use JavaScript or Python for a project, and you think Python would be the better choice. How would you explain the similarities and differences between the two languages to your team? Drawing from what you learned in this Exercise, what reasons would you give to convince your team that Python is the better option?

   (Hint: refer to the Exercise section "The Benefits of Developing with Python")

Python and JavaScript both emphasize readability with clear syntax and keywords. Both have various frameworks with ready-to-use features for web development, like setting up web pages and databases. Python, however, offers an extensive library of built-in packages, simplifying complex tasks like math operations and data manipulation. Plus, Python is highly advanced in areas such as Machine Learning, Robotics, and AI.

3. Now that you've had an introduction to Python, write down 3 goals you have for yourself and your learning during this Achievement. You can reflect on the following questions if it helps you. What do you want to learn about Python? What do you want to get out of this Achievement? Where or what do you see yourself working on after you complete this Achievement?

- *I plan to apply Python to real-world projects, improving my web development abilities*
- *I'm interested in learning frameworks like Django to build dynamic web applications.*
- *Master Python essential concepts like variables, data types, loops, and functions to build a strong foundation.*

Exercise 1.2: Data Types in Python

Learning Goals

- Explain variables and data types in Python
- Summarize the use of objects in Python
- Create a data structure for your Recipe app

Reflection Questions

1. Imagine you're having a conversation with a future colleague about whether to use the iPython Shell instead of Python's default shell. What reasons would you give to explain the benefits of using the iPython Shell over the default one?

   Using the iPython Shell offers clearer code readability with syntax highlighting, making it easier to distinguish keywords and lines of code. Unlike Python's default shell, iPython automatically handles indentation for nested statements, enhancing code organization. Overall, iPython provides a more user-friendly environment.

2. Python has a host of different data types that allow you to store and organize information. List 4 examples of data types that Python recognizes, briefly define them, and indicate whether they are scalar or non-scalar.

Data type	**Definition**	**Scalar or Non-Scalar?**
bool	Bools representing True or False values.	Scalar
dictionary	A dictionary in Python is a collection of key-value pairs, where each key maps to a corresponding value.	Non-Scalar
list	A list in Python is a mutable and stores multiple values	Non-Scalar
tuple	A tuple is an immutable and stores multiple values	Non-Scalar

3. A frequent question at job interviews for Python developers is: what is the difference between lists and tuples in Python? Write down how you would respond.

Lists and tuples both used to store multiple values, but there is a key difference. Lists are mutable, meaning you can add, remove, or modify elements after creation, while tuples are immutable and cannot be changed once created.

4. In the task for this Exercise, you decided what you thought was the most suitable data structure for storing all the information for a recipe. Now, imagine you're creating a language-learning app that helps users memorize vocabulary through flashcards. Users can input vocabulary words, definitions, and their category (noun, verb, etc.) into the flashcards. They can then quiz themselves by flipping through the flashcards. Think about the necessary data types and what would be the most suitable data structure for this language-learning app. Between tuples, lists, and dictionaries, which would you choose? Think about their respective advantages and

limitations, and where flexibility might be useful if you were to continue developing the language-learning app beyond vocabulary memorization.

I would choose the dictionary data type for the flashcards because it allows me to store the information in key-value pairs. This means I can store the word, definition, and category as keys, making it easy to organize and retrieve data. So, the advantage compared to lists or tuples would be the efficient and fast access to the information needed. Additionally, dictionaries are mutable, so I can modify the flashcards as needed, adding or updating data. This could also be useful, because I might want to add more information to the flashcards later on. However, a limitation is that to retrieve the desired information, I need to know the specific key associated with the information first.

Exercise 1.3: Functions and Other Operations in Python

Learning Goals

- Implement conditional statements in Python to determine program flow
- Use loops to reduce time and effort in Python programming
- Write functions to organize Python code

Reflection Questions

1. In this Exercise, you learned how to use **if-elif-else** statements to run different tasks based on conditions that you define. Now practice that skill by writing a script for a simple travel app using an **if-elif-else** statement for the following situation:

    - The script should ask the user where they want to travel.
    - The user's input should be checked for 3 different travel destinations that you define.
    - If the user's input is one of those 3 destinations, the following statement should be printed: "Enjoy your stay in ______!"
    - If the user's input is something other than the defined destinations, the following statement should be printed: "Oops, that destination is not currently available."

Write your script here. *(Hint: remember what you learned about indents!)*

```
travel_destinations = ["japan", "taiwan", "vietnam"]
country = str(input("Enter where you want to travel: "))
if country.lower() in travel_destinations:
    print("Enjoy your stay in", country + "!")
else
    print("Oops, destination is not currently available.")
```

2. Imagine you're at a job interview for a Python developer role. The interviewer says "Explain logical operators in Python". Draft how you would respond.

Logical operators are commonly used in conditional statements to make decisions based on if the condition is True or False. The ***and*** *operator returns True only if both conditions are True. The* ***or*** *operator returns True if at least one of the conditions is True. The* ***not*** *operator returns the opposite boolean value of its condition boolean value.*

3. What are functions in Python? When and why are they useful?

Functions in Python are reusable blocks of code created using the def keyword. Functions are used when some operation in the code keeps repeating. In that case you can encapsulate the specific task in a function and then just use the function when you need to perform that task. Functions help organize code, avoid repetition, and improve readability.

4. In the section for Exercise 1 in this Learning Journal, you were asked in question 3 to set some goals for yourself while you complete this course. In preparation for your next mentor call, make some notes on how you've progressed towards your goals so far.

- *I plan to apply Python to real-world projects, improving my web development abilities. → not yet*
- *I'm interested in learning frameworks like Django to build dynamic web applications. → not yet, will happen in achievement 2*
- *Master Python essential concepts like variables, data types, loops, and functions to build a strong foundation. → I have practiced all of these in the last three exercises and start to feel more confident working with them.*