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?

Other than my experience with CareerFoundry courses, I've taken web development courses at a community college a few years ago and learned the html, css, and javascript basics.

2. What do you know about Python already? What do you want to know?
   I have not yet learned about Python. I've only researched and read that for developers with coding knowledge it is a beginner friendly programming language to learn. I would like to know how to create applications with Python and become familiar with the language/its syntax so I can use those skills for future jobs and for my own personal projects.

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.
   Personally, time management may be an issue as I try to balance this course with work and my personal life. However, I will try my best to get tasks done on time and schedule out my weekly study times. From past CF achievements I know I may encounter technical errors that may hold me back and take a while to solve. I'll try my best to research solutions and reach out to the cf community for help, including my mentor.

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?
   When developing code for the front end of an application you're developing the interface that users will see on their devices and the elements that they interact with (buttons, navigation bar, etc.). When developing code for the backend you're usually dealing with code for servers, storing user data on the client side and the server-side, APIs, managing databases, creating models, etc.

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 is one of the most widely used and recommended programming languages. Both Python and Javascript are dynamically typed, however, one of Python's core advantages is its easy readability and clean syntax. That helps to keep Python code a lot more error free due to easier documentation and debugging. Python also has many packages at hand to suit whatever project may come about through its built-in package management with pip. It also has one of the largest communities backing and supporting it.

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 want to become familiar with Python to be able to broaden my skill set as a web developer and be able to use those skills for future projects. I also want to be able to use my knowledge of Python to become a better asset to companies that I may work for as learning Python opens up many different opportunities for me. I've learned that Python is very helpful to automate a lot of things for developers and make things move faster. I would love to know how to incorporate automated scripts with Python for current and future personal projects of mine. There are also several Python hackathons/events in my area that are held. Once I have more knowledge of the language I will feel more comfortable attending one of these events and networking at them.

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?
   When using Pythons shell things may get a bit confusing due to the colors/ui of the python shell being quite bland. The input and output within the shell look the same and can be easily mistaken for one another. It's just not user friendly. In contrast, the iPython shell contains highlighting for input and

outputs as well as for different parts of your code. For example when I type strings surrounded by " the text color is orange, without the " the text color is grey. The regular python does not have features like that.

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?
Tuples	Linear arrays that can store multiple values of any type. `odd_numbers = (1, 3, 5, 7, 9)`	Non-scalar
Lists	An ordered sequence, similar to tuple, but they are mutable. Meaning any of the internal elements of a list can be modified or deleted and you can even rearrange or insert new elements in lists. `animals = ['ferret', 'cat', 'bat', 'dog', 'iguana']`	Non-scalar
Strings	Immutable array of characters denoted as **str** and surrounded by single or double quotes `s_1 = "This is a fruit."`	Non-scalar
Dictionaries	Stores values and objects within itself indexed by identifiers/keys. An unordered set of items, each of the items being a key-value pair. `scores = { 'math': 7, 'economics': 8, 'geography': 10 }`	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.
The difference between a list and a tuple is that a list is mutable, meaning you can modify, delete, or even rearrange the elements of a list. You cannot do that in a tuple, making it immutable.

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 use dictionaries. With dictionaries we can continue to add to and modify new word inputs. I would possibly use lists as the outer structure for the 'Words' so that there can be an infinite amount of words added that can be modified or removed. Then each word would have the dictionary structure with a name, definition, and type as the keys.

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!)*

```
destination = input(“Where would you like to travel to? ”)
if destination == “New York”
   print(“Enjoy your stay in New York!”)
elif destination == “Italy”
   print(“Enjoy your stay in Italy!”)

elif destination == “Australia”
   print(“Enjoy your stay in Australia!”)

else:
   print(“Sorry, that is an invalid destination”)
```

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.
   There are three boolean logic operators in Python which are “and”, “or” and “not”. Boolean meaning that they return a result of either true or false. These operators are often used to check multiple conditions at the same time. You can use the and operator to check if two conditions are true, the result will only return true if BOTH conditions are met. Otherwise it will be false. The or operator is used to check if either one or both conditions are true, as long as one condition is true the result will be true. And the not operator is used to reverse the result of a logical expression, which in result flips the answer of a true or false statement.

3. What are functions in Python? When and why are they useful?
   Functions are sets of instructions that you can use to process your code and achieve certain things. They save a lot of time when you have repetitive code or an operation in your code that you need to keep repeating every so often. Instead of retyping all those lines of code every time, you can just call the name of the function instead. Functions also help to keep code well organized, clean, and concise.

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’ve progressed on all of the goals I mentioned. I am having fun learning Python and getting comfortable with the basics as it has a very user friendly syntax. I’m still at the beginning of learning it but I already feel like I can use this knowledge to become a better asset for future companies that I will work for. I’m also starting to see how you can use Python to create automated scripts and becoming more familiar with the process, especially after this task, which is another goal I listed.

Exercise 1.4: File Handling in Python

Learning Goals

- Use files to store and retrieve data in Python

Reflection Questions

1. Why is file storage important when you're using Python? What would happen if you didn't store local files?
   File storage in python is important so your data is not lost after running your script. If you don't store local files, then after running your script and inputting data into it, that data you just spent time inputting will be lost once the script is done running. Such as when you use variables in your script to assign and keep track of values, that data no longer exists when the script stops running.

2. In this Exercise you learned about the pickling process with the **pickle.dump()** method. What are pickles? In which situations would you choose to use pickles and why?
   Pickles are able to convert complex data into a packaged stream of bytes, that stream of bytes being a pickle. Those bytes are then written into a binary file which can only be read by a machine, not humans. Pickles are best used for complex data structures to retain their structure, such as when you're using dictionaries. Simpler data structures can be stored within txt files.
3. In Python, what function do you use to find out which directory you're currently in? What if you wanted to change your current working directory?
   To find out what directory you're currently in you can execute the following lines:
   Import os
   print(os.cwd())
   and to change directories you can use:
   os.chdir(<path to folder>)

4. Imagine you're working on a Python script and are worried there may be an error in a block of code. How would you approach the situation to prevent the entire script from terminating due to an error?
   To prevent the entire script from terminating due to an error you can use the try-except block. You first try a block of code where you'd expect an error to occur. If no errors are found, the rest of your code is executed as normal. If there's an error, an except block appears to notify the user of the error and guide them to correct it.

5. You're now more than halfway through Achievement 1! Take a moment to reflect on your learning in the course so far. How is it going? What's something you're proud of so far? Is there something you're struggling with? What do you need more practice with? Feel free to use these notes to guide your next mentor call.

So far things are going pretty good with this course, Pythons syntax is very simply written and easy to learn from. It's just a matter of me memorizing the syntax and the meaning of specific vocabulary. I think I need more practice with for loops in general to get used to writing and understanding them. I also hope to find the time soon to follow along with the youtube video my mentor recommended to code a beginner python game!

Exercise 1.5: Object-Oriented Programming in Python

Learning Goals

- Apply object-oriented programming concepts to your Recipe app

Reflection Questions

1. In your own words, what is object-oriented programming? What are the benefits of OOP?
   OOP, or Object Oriented Programming, creates objects from classes. Using OOP and its techniques has many benefits such as easily being able to reuse code through inheritance and flexibility with method names thanks to polymorphism.

2. What are objects and classes in Python? Come up with a real-world example to illustrate how objects and classes work.
   In Python, classes are like the parent. Objects are like individual children. For this recipe app, the 'Recipe' is the class while its objects aka its different parameters are the name, cooking time, ingredients, and difficulty.

3. In your own words, write brief explanations of the following OOP concepts; 100 to 200 words per method is fine.

Method	Description
Inheritance	**Inheritance comes in handy when you have methods/properties of one class that could be useful to another, with Python you can inherit methods from one class to another. The class being inherited from is the parent class or base class. The class which is copying all the parents attributes is called the subclass or inherited class. One thing to remember is that inheritance only flows one way, from the parent down. This helps to keep code from being lengthy and repetitive.**
Polymorphism	**This helps with flexibility when it comes to the names of methods and data attributes. You can use the same name across different classes and data types, and the operation that is performed depends on where in the code you are defining it.**

Operator Overloading	**This is used to apply operators on custom classes since they won't automatically work unless its on a built-in Python class. For the process you define a function with a name that Python already uses such as add and surround it with double underscores on each side and then parenthesis: __add__()**

Exercise 1.6: Connecting to Databases in Python

Learning Goals

- Create a MySQL database for your Recipe app

Reflection Questions

1. What are databases and what are the advantages of using them?

2. List 3 data types that can be used in MySQL and describe them briefly:

Data type	Definition

3. In what situations would SQLite be a better choice than MySQL?

4. Think back to what you learned in the Immersion course. What do you think about the differences between JavaScript and Python as programming languages?

5. Now that you're nearly at the end of Achievement 1, consider what you know about Python so far. What would you say are the limitations of Python as a programming language?

Exercise 1.7: Finalizing Your Python Program

Learning Goals

- Interact with a database using an object-relational mapper
- Build your final command-line Recipe application

Reflection Questions

1. What is an Object Relational Mapper and what are the advantages of using one?

2. By this point, you've finished creating your Recipe app. How did it go? What's something in the app that you did well with? If you were to start over, what's something about your app that you would change or improve?

3. Imagine you're at a job interview. You're asked what experience you have creating an app using Python. Taking your work for this Achievement as an example, draft how you would respond to this question.

4. You've finished Achievement 1! Before moving on to Achievement 2, take a moment to reflect on your learning in the course so far:
    a. What went well during this Achievement?
    b. What's something you're proud of?
    c. What was the most challenging aspect of this Achievement?
    d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Python skills?
    e. What's something you want to keep in mind to help you do your best in Achievement 2?

Well done—you've now completed the Learning Journal for Achievement 1. As you'll have seen, a little metacognition can go a long way!

Pre-Work: Before You Start Achievement 2

In the final part of the learning journal for Achievement 1, you were asked if there's anything—on reflection—that you'd keep in mind and do similarly or differently during Achievement 2. Think about these questions again:

- Was your study routine effective during Achievement 1? If not, what will you do differently during Achievement 2?

- Reflect on your learning and project work for Achievement 1. What were you most proud of? How will you repeat or build on this in Achievement 2?
- What difficulties did you encounter in the last Achievement? How did you deal with them? How could this experience prepare you for difficulties in Achievement 2?

Note down your answers and discuss them with your mentor in a call if you like.

Remember that 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 2.1: Getting Started with Django

Learning Goals

- Explain MVT architecture and compare it with MVC
- Summarize Django's benefits and drawbacks
- Install and get started with Django

Reflection Questions

1. Suppose you're a web developer in a company and need to decide if you'll use vanilla (plain) Python for a project, or a framework like Django instead. What are the advantages and drawbacks of each?

2. In your own words, what is the most significant advantage of Model View Template (MVT) architecture over Model View Controller (MVC) architecture?

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

Exercise 2.2: Django Project Set Up

Learning Goals

- Describe the basic structure of a Django project
- Summarize the difference between projects and apps
- Create a Django project and run it locally
- Create a superuser for a Django web application

Reflection Questions

1. Suppose you're in an interview. The interviewer gives you their company's website as an example, asking you to convert the website and its different parts into Django terms. How would you proceed? For this question, you can think about your dream company and look at their website for reference.
   (*Hint: In the Exercise, you saw the example of the CareerFoundry website in the Project and Apps section.*)

2. In your own words, describe the steps you would take to deploy a basic Django application locally on your system.

3. Do some research about the Django admin site and write down how you'd use it during your web application development.

Exercise 2.3: Django Models

Learning Goals

- Discuss Django models, the "M" part of Django's MVT architecture
- Create apps and models representing different parts of your web application
- Write and run automated tests

Reflection Questions

1. Do some research on Django models. In your own words, write down how Django models work and what their benefits are.

2. In your own words, explain why it is crucial to write test cases from the beginning of a project. You can take an example project to explain your answer.

Exercise 2.4: Django Views and Templates

Learning Goals

- Summarize the process of creating views, templates, and URLs
- Explain how the “V” and “T” parts of MVT architecture work
- Create a frontend page for your web application

Reflection Questions

1. Do some research on Django views. In your own words, use an example to explain how Django views work.

2. Imagine you’re working on a Django web development project, and you anticipate that you’ll have to reuse lots of code in various parts of the project. In this scenario, will you use Django function-based views or class-based views, and why?

3. Read Django’s documentation on the Django template language and make some notes on its basics.

Exercise 2.5: Django MVT Revisited

Learning Goals

- Add images to the model and display them on the frontend of your application
- Create complex views with access to the model
- Display records with views and templates

Reflection Questions

1. In your own words, explain Django static files and how Django handles them.

2. Look up the following two Django packages on Django’s official documentation and/or other trusted sources. Write a brief description of each.

Package	Description
ListView	
DetailView	

3. You're now more than halfway through Achievement 2! Take a moment to reflect on your learning in the course so far. How is it going? What's something you're proud of so far? Is there something you're struggling with? What do you need more practice with? You can use these notes to guide your next mentor call.

Exercise 2.6: User Authentication in Django

Learning Goals

- Create authentication for your web application
- Use GET and POST methods
- Password protect your web application's views

Reflection Questions

1. In your own words, write down the importance of incorporating authentication into an application. You can take an example application to explain your answer.

2. In your own words, explain the steps you should take to create a login for your Django web application.

3. Look up the following three Django functions on Django's official documentation and/or other trusted sources and write a brief description of each.

Function	Description
authenticate()	
redirect()	

include()	

Exercise 2.7: Data Analysis and Visualization in Django

Learning Goals

- Work on elements of two-way communication like creating forms and buttons
- Implement search and visualization (reports/charts) features
- Use QuerySet API, DataFrames (with pandas), and plotting libraries (with matplotlib)

Reflection Questions

1. Consider your favorite website/application (you can also take CareerFoundry). Think about the various data that your favorite website/application collects. Write down how analyzing the collected data could help the website/application.

2. Read the Django official documentation on QuerySet API. Note down the different ways in which you can evaluate a QuerySet.

3. In the Exercise, you converted your QuerySet to DataFrame. Now do some research on the advantages and disadvantages of QuerySet and DataFrame, and explain the ways in which DataFrame is better for data processing.

Exercise 2.8: Deploying a Django Project

Learning Goals

- Enhance user experience and look and feel of your web application using CSS and JS
- Deploy your Django web application on a web server
- Curate project deliverables for your portfolio

Reflection Questions

1. Explain how you can use CSS and JavaScript in your Django web application.

2. In your own words, explain the steps you'd need to take to deploy your Django web application.

3. (Optional) Connect with a few Django web developers through LinkedIn or any other network. Ask them for their tips on creating a portfolio to showcase Python programming and Django skills. Think about which tips could help you improve your portfolio.

4. You've now finished Achievement 2 and, with it, the whole course! Take a moment to reflect on your learning:
    a. What went well during this Achievement?
    b. What's something you're proud of?
    c. What was the most challenging aspect of this Achievement?
    d. Did this Achievement meet your expectations? Did it give you the confidence to start working with your new Django skills?

Well done—you've now completed the Learning Journal for the whole course.