Python Learning Journal

Exercise 1.1: Getting Started with Python

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 web development is what the user sees and interacts with. Backend is where operations that occur for handling requests, interacting with databases and files the server. If I was hired as a backend programmer, I would anticipate working on CRUD operations, working with databases, security concerns, and working with servers.

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?**

    JavaScript and Python are both scripting languages that use dynamic typing. JavaScript is not optimized for backend development, however Python's straightforward syntax and high readability and range of libraries and frameworks may make more sense in this situation.

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 learn basic Python syntax.
    - In this achievement I want to be able to setup an environment to get started using Python.
    - After this achievement I want to continue to build my knowledge and skillset to become marketable as a backend developer.

Exercise 1.2: Data Types in Python

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?**

iPython is more user-friendly. It utilizes different colors making reading the code easier. The lines are numbered. There is syntax highlighting and indenting is done automatically.

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?
String str	immutable, can be composed of alphanumeric characters and symbols contained in single or double quotes	Non-Scalar
Numeric int	integer (whole number) negative or positive (to infinity)	Scalar
Boolean	True or False statement	Scalar
Sequence list	mutable, ordered sequence of data using square brackets and commas	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.**

While both are a sequence of data, a tuple is immutable while a list is mutable. Data in a tuple is write-protected. A list can be more easily manipulated. In the code, a tuple is written using parenthesis and commas while a list utilizes brackets and commas.

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.**

For each input they would be stored as strings. Having words, definitions, and category may get confusing if they were just stored as a list. If you used a dictionary you can use key-value pairs to distinguish between categories. Also with a dictionary you can always extract the keys or values and place those into a list if needed in the future.

Exercise 1.3: Functions and Other Operations in Python

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:**

```
destination = input("Where would you like to travel to? ")
if destination == 'Hawaii':
   print("Enjoy your stay in Hawaii!")
elif destination == 'Alaska':
   print("Enjoy your stay in Alaska!")
elif destination == 'New York':
   print("Enjoy your stay in New York!")
else:
   print("Oops, that 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 operations are used on conditional statements and will return True or False. The and operator will return true if both statements are true and false if either statement is false. The or operator will return true is either statement is true and false only if both statements are false. The not operator will return the opposite of the statement (if statement is true it will return false but a false statement will return true).

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

   Functions are blocks of code run only when they are called. They are useful for manipulating data or code. They cut down on repetitive code and makes it easier to read. They are many builtin functions in Python or you can create your own to carry out a specified task.

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 am learning the language by completing exercises and looking up Python documentation. My mentor has given me resources to explore online as well. I am getting comfortable with using the Python Shell and executing a script in the terminal. After meeting with my mentor for the first time, I am gaining industry insight on what I need to become marketable as a backend developer. This will entail learning another coding language to expand my knowledge base.

Exercise 1.4: File Handling in Python

1. **Why is file storage important when you're using Python? What would happen if you didn't store local files?**

   When you use variables to assign and keep track of values that data doesn't exist when the script stops running. All the data is lost and cannot be used later. Creating local files allows you to store data permanently.

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 convert data into a byte stream. This binary file can store complex information. Pickles need to be used for anything that is not basic data, which can be stored as a simple text file.

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?**

   The os module using the os.getcwd() command to find out what file directory you're currently in. The os.chdir('<path to desired folder>') command is used to navigate to another directory.

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?**

   By using a try-except block. The try block is where you place the code that the error may occur. If there is no error than the rest of the code is executed as normal. If there is an error then the except block handles the error. After which, the code continues to run until the end.

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.**

   I am really enjoying this achievement. I had to take a week off and that is not ideal in learning any new skill but fortunately I was able to jump back in. I find the hardest part for me is starting writing the code for a task. Once I figure out where to start I can logically think my way through it. I hope to get over this hurdle of "I don't know where to start" by just consistently doing the assignments and becoming more familiar with Python.

Exercise 1.5: Object-Oriented Programming in Python

1. **In your own words, what is object-oriented programming? What are the benefits of OOP?**

   OOP is when you bundle data and methods into individual objects. The benefit of OOP is that it keeps code simple and non-repetitive. You can pass information to the "template" and it will give you the functionality of that template for use with your particular object.

2. **What are objects and classes in Python? Come up with a real-world example to illustrate how objects and classes work.**

   Everything is an object in Python. Each object has a corresponding class. A class is a template that provides an internal structure. One class can have many objects. An example is a class, shoe. You can then create an individual shoe object, like a sneaker. You could describe color, function, and other attributes for that individual shoe. Another shoe object could be a dress shoe with its individual characteristics that are setup in the shoe class.

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

Method	Description
Inheritance	This is when you pass the properties and methods of a class to another class for use. This is done when declaring your class you pass the inherited class into the parentheses at the beginning. Then it is available to the subclass you just created for use without having to redefine those attributes.
Polymorphism	This is where a data attribute or method has the same name across different classes or data types but performs different operations. This can be built in or custom.
Operator Overloading	This is when you use operators on a custom class. The method for the operator use has to be a function defined in the class. An example is __add__() would need to be a defined function in your class and then the + operator can be used to call this function.