**Reflection on Fundamentals of Python**

1. **In the past week(s) you have learned (i.e., researched, read, applied) new aspects of your  
   chosen programming language**

* **What did I learn about python**

Over the past few weeks, I have gained valuable insights into Python while working on my first project. Some key aspects I have learned include:

1. Object-Oriented Programming (OOP):

I explored how to structure code using classes and objects in python, which helped

in organizing data efficiently and improving code reusability.

1. Data Structures:

I gained experience in using built-in data structures to store and manage data effectively, for example list data structure can store different data types and I can used built-in methods to easily access and manipulate records.

1. Exception Handling:

I learned how to handle unexpected errors in Python to ensure smooth program execution. I found Python's error handling simpler and more intuitive compared to other OOP languages.

1. Commenting and Documentation:

I learned two types of commenting in Python (single-line ‘#’ and multi-line “””). Writing clear and concise comments helped make the code more understandable for others and useful for future reference.

* **What was interesting or fun about python learning**

I found Python’s simplicity and readability to be the most enjoyable aspects of learning. Unlike other programming languages I have experience with, such as C and Java, Python’s straightforward syntax made writing code faster and easier to understand.

1. **With regard to learning resources, that you used to learn (books, videos, web resources, other programmers etc.)**

To learn and apply Python concepts, I used various learning resources such as online documentation, YouTube tutorials, chatbots and forums.

**Resources That Worked Best for Me:**

* **Python Documentation and W3Schools:** These were the most helpful because they provided clear explanations and step-by-step examples, making it easy to understand and apply new concepts.
* **Chatbots and Forums (ChatGPT, Stack Overflow):** They were useful for quick troubleshooting and clarifying specific doubts during coding.

**Resources That Were Less Successful:**

* **Some YouTube Videos:** Although useful, some were too lengthy and covered broad topics that were not always relevant to my specific project needs.
* **Long Articles and Blogs:** They contained too much information at once, making it challenging to quickly find the specific details I needed.

**Time-Consuming vs. Quick Resources:**

* **Most Time-Consuming:** Python Documentation and YouTube tutorials with detailed examples took the most time but were highly informative.
* **Least Time-Consuming:** W3Schools for Python provided easy-to-understand, example-based learning, and ChatGPT was useful for quick explanations and clarifications, helping me compare my previous knowledge of C and Java with Python.

**Effective Resources Based on My Experience:**

* Python Documentation for in-depth learning.
* W3Schools for easy-to-follow tutorials.
* ChatGPT and Stack Overflow for quick problem-solving.

**List of Resources I used:**

Python documentation- <https://docs.python.org/3/library/unittest.html>

W3schools- <https://www.w3schools.com/python/python_intro.asp>

YouTube tutorials- <https://www.youtube.com/watch?v=kqtD5dpn9C8> <https://www.youtube.com/watch?v=DmHSwTiD5Tk>

Blogs and forums- <https://www.datacamp.com/blog/how-to-learn-python-expert-guide>

<https://realpython.com/learning-paths/> ; <https://discuss.python.org/>

1. **In Research Assignment 1, I have created a Work Breakdown Structure and Gantt chart**

* **WBS accuracy**

I used my professor's WBS structure example as a guide and worked on to fit my project needs. It provided a clear outline and helped me stay on track. However, I realized that some tasks needed more detail, especially for testing and debugging. In the future, I will focus on breaking down tasks further to better manage my time and workload.

* **Lessons learned on WBS for future work**

In future projects, I will ensure that I include smaller, more detailed tasks and allocate time for potential challenges such as debugging and documentation.

* **Time estimation in Gantt chart**

I found that I underestimated the time required for testing and refining my code. Some tasks, such as learning file handling and exception management, took longer than expected.

* **Improvement on time estimation**

To improve, I will analyze my past projects and plan buffer time for potential roadblocks. I will also rely on my experience to make more realistic estimates in future assignments.