# Term Portfolio Project

**Taha Fareed**

**V00990356**

**Seng265: Software Development Methods**

## I. Introduction

Software development methods are extremely important in today's day and age with endless exponential growth in technology and information. Software developers are important because they

create software that helps humans deal with the limitless information on the internet and new technologies that are being invented every day. The purpose of this portfolio

is to document my journey of learning Python and C; specifically, by learning these languages for software development methods. This portfolio consists of two parts and

a approximately 16 pages with weekly contributions. The first part has 5-6 pages. Part one of the portfolio consists of my weekly contributions that document my learning

journey and experiences of first starting Python and C in software development methods. Part one will answer the following questions and allow me to further explain my

understanding and journey exploring these topics.

**The topics that are discussed are:**

- The Core Functionality of Jupyter Notebook
- How To Write Mathematical Formulas Using Latex markdown in Jupyter Notebook
- The Core Functionalities Git, Github, and Gitlab
- The Most Important Programming Languages

- The Python Concept of Comprehension and Slicing
- Advantages of Classes for Packaging and Encapsulation and the Significance of Object-Oriented Design
- The Concepts and Applications of Objects, Class Variables, and Methods
- Understanding Deep Copy and Shallow Copy in Python: The Purpose of copy() and deepcopy() Methods in the Python Copy Module
- The Notion of and Motivation for Typing and Typing Hints in Python
- How Challenging Assignment Three was for Me

To conclude, it is extremely important to consistently reflect on your academic journey, career, and life decisions. Due to the importance of documenting one's academic journey, this

portfolio project will show you my reflection on the new ideas that were taught to me throughout this course and how I have gone about learning this knowledge. I hope this

portfolio serves as a reflection of my progress, and I would like to thank you for taking the time to review it.

# II. The Core Functionality of Jupyter Notebook

Jupyter Notebook is a versatile web application whose main purpose is to create and share documents that contain live code,

math equations, visualizations of any sort, and text such as this document (Software Developers,Quality Assurance Analysts, and

Testers: Occupational Outlook Handbook:: U.S. Bureau of Labor Statistics, 2023). Jupyter Notebook has four main features.

**The overview of these four features are:**

1. Notebook Document
    - The Jupyter notebook app is used to create documents that can contain code, visuals, text, and equations; making them "human-readable" versions of whatever software is using these notebooks. These documents are also executable files, meaning that they can be executed into workable programs or software. According to the Jupyter/IPython Notebook Quick Start Guide (n.d.), the notebooks created using this app have become increasingly popular among researchers and data scientists due to their versatility and ease of use.

2. Jupyter Notebook App

    - The Jupyter Notebook App allows the editing and running of notebook documents using a web browser. The app can be launched and run files on a local computer with or without internet access (What Is the Jupyter Notebook? — Jupyter/IPython Notebook Quick Start Guide 0.1 Documentation, n.d.).

3. Kernel

    - A notebook Kernel is a "computational engine" that executes the code contained in a Notebook document. The python Kernel, referenced in this guide, executes python code.

4. Notebook Dashboard

    - The dashboard is shown first when you launch the Jupyter Notebook App. The Dashboard's main focus is to create or open documents and manage the currently running kernels (What Is the Jupyter Notebook? — Jupyter/IPython Notebook Quick Start Guide 0.1 Documentation, n.d.)

I have used Jupyter Notebook once before for my introductory computer science class (CSC-110). At the time, I did not

understand just how popular and helpful Jupyter Notebook is. Learning how to use Jupyter Notebook for this very portfolio

project has made me realize how many people use Jupyter Notebook and after researching Jupyter Notebook, I am going to use this

application for my future python projects.

# III. Writing Mathematical Formulas Using Latex Markdown in Jupyter Notebook

Markdown and Latex are two seperate tools, however, they are used together often. Markdown is a conversion tool for web writers; Markdown allows a user to write using a plain text format

and then convert it to a structurally valid XHTML or HTML file (Mirzaee, n.d.). Latex is a typesetting system that was created to put mathematical and scientific

documentation on an HTML or XHTML file. Markdown is different from Latex as Latex is a typesetting toolthat can be used inside of a Markdown file to increase readability.

**How to use Latex to write mathematical formulas:**

$$e^{i\pi} + 1 = 0$$

Another example of using Latex to form mathematical equations would be to show that it can also form complicated written equations like Taylor's Theorem:

$$f(x) = \sum_{k=0}^{\infty} f^{(k)}(a)\frac{(x-a)^k}{k!}$$

Before taking SENG265, I had not used a Markdown file. However, with the use of the Markdown files used for the assignments in the course, I have realized that Markdown files help hold

easy-to-read information for users. I have also utilized Latex in the past for a first-year discrete mathematics course (Math-122). Although I was introduced to Latex in

that course, I rarely experimented with Latex, despite it being a very helpful and important tool; mainly because I have not yet had a situation to where the application of

this knowledge was useful. Although, I have limited experience with both Markdown and Latex; I envision myself applying knowledge I have gained in this lass on future projects.

## III. The Core Functionalities of Git, Github, and Gitlab

Git is a DevOps tool that was originally developed in 2005 for source code management. Github and Gitlab both use Git, however, they include different features and capabilities. Github,

originally, was a cloud-based Git platform that would allow developers to host and monitor changes to their code. However, presently Github and Gitlab have evolved to become a

development platform which allows developers to edit, monitor, test, and deploy their code. Github, unlike GitLab, allows developers to integrate whatever apps and integrations

the developer chooses with the use of the GitHub marketplace. Gitlab on the other hand, much like GitHub, is a cloud-based DevOps platform with the same ideas as such as

allowing developers to monitor, edit, test, and deploy their code. Developers find Gitlab attractive due to its range of features such as cloud-hosted Git repositories,

continuous integration for automated testing, robust security measures, tools for deploying applications, and various other DevOps functionalities. Github and Gitlab are

extremely similar with the features they provide, the main difference is up to the developer and how they choose to access the features provided to them. Either a much more

structural version like GitLab or a much less structured and more free version like GitHub. Overall, both these softwares are similar with the features they offer but differ in

the way they maintain and publish code. Github's open marketplace allows for more flexibility in integrating third-party apps and services, and Gitlab offers a more structured

approach that could potentially offer greater control and security over the software development process. Over my short programming career, I never used GitLab and

always used GitHub. Therefore, I tried to stray away from GitHub because as a new programmer, I did not understand how to use GitHub effectively due to the lack of structure

it provided. I realize now, after researching this topic, that Github's lack of structure is actually what makes it so popular and I will try to learn how to use the

GitHub marketplace to my advantage for my future projects.

# IV. The Most Popular Programming Languages

According to the software company, TIOBE, Python, C, and Java are the three most popular programming languages. Python is an object-oriented high-level programming language with built-in

data structures, dynamic typing, and dymanic building. These features make Python one of the main languages for rapid application development. Python is popular because

of its easy-to-read syntax and it does not need a compiler; making the edit-test-debug cycle extremely fast (What Is Python? Executive Summary, n.d.). The C programming

language is a general purpose programming language that is flexible to use; C's main purpose is used for developing system applications. C has many application uses,

such as UNIX, Google, and MySQL. Java is a multi-platform, object-oriented, and network centric language designed for having lesser implementation dependencies. Java's ecosystem

is thouroughly built with a warehouse full of libraries that allow programmers to save time by not building redundant functions. Despite these three languages being the

most popular, they have their differences. Java is a platform-dependent and is a compiled programming language, Python is interpreted and platform-independent, and C is a

compiled programming language that can create vastly complex systems due to its bare-boned structure. I have had a lot of experience with both Java and Python, with Java being my

most preferred. I struggle with grasping the concepts of C due to it having a perceived stripped understanding. After taking the first half of SENG-265, I have realized the

importance of C in software development, it has encouraged to pursue a further understand of the language in hopes of one day furthering my progress in computer science.

# V. The Python concept of comprehension and slicing

Python is the most popular language in the world and there are many different concepts within Python that are important to understand. There are four different comprehensions in python:

list comprehensions, dictionary comprehensions, set comprehensions, and generator comprehensions (GeeksforGeeks, 2018). List comprehension is a cultivating way to create new

lists by shortening the syntax of creating new lists. List comprehensions may contain singular or mulitple if statements within the program, this in turn leads to nested list comprehensions. For

example, if you have a list of ten numbers in ascending order `[1,2,3,4,5,6,7,8,9,10]` the list comprehensions create a new list with even numbers. After the program runs, the result list will appear as:

**The Program:**

```
li = [1,2,3,4,5,6,7,8,9,10]

resultLi = []

for i in li:

    if (i%2==0):

        resultLi.append(i)
```

**Output:**

```
[2,4,6,8,10]
```

Dictionary comprehension uses the idea of list comprehensions, dictionary comprehensions differ by using a key/value pair. A dictionary would be in the form of `myDict = {1: "soccer", 2: "basketball", 3:`

`"baseball", 4: "volleyball"}` . If you want to print the word "basketball" from the dicitonary, you would have to use the print statement as it is shown: `print(myDict[2])` . In Code,

this would look like this:

**The Program:**

```
myDict = {1: "soccer", 2: "basketball",  3: "baseball", 4: "volleyball"}

print(myDict[2])
```

**Output:**

```
basketball
```

Set comprehension is virtually similiar to a list comprehension, excluding the fact that it uses curly brackets "{ }" and not square brackets "[ ]"; instead of creating a new list it

creates a new set. Set comprehension and list comprehension both have their different uses such as set comprehension not allowing duplicate data types. Lets

say you have a list `li = [1,1,2,2,3,3,4,4]` , you can use list comprehension to get rid of the duplicates and create only `{1,2,3,4}` . Here is an

example of a working program that demonstrates sets:

**The Program:**

```
li = [1,1,2,2,3,3,4,4]

resultSet = set()

for i in li:

    resultSet.add(i)

print(resultSet)
```

**Output**:

```
{1,2,3,4}
```

Generator comprehension is almost identical to list comprehension, however, generator comprehension uses ciruclar brackets "( )" while list comprehension uses square brackets "[ ]"

. Generator comprehensions are primarily memory efficient as it does not allocate memory for the whole list but instead one by one for each element. Here's an example of creating a generator:

**The Program:**

```
li = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

oddGen = (x for x in li if x % 2 != 0)

print(oddGen)
```

If I were to run this code, since the generator does not allocate memory for the whole list and just adds memory as it goes, it will not print a list of only odd numbers but instead

print out a string representation of the generator object which includes the type if object and the memory address. If I wanted to print the elements in oddGen, I would

have to iterate over the generator comprehension like:

```
for i in oddGen:
        print(i)
```

and it will print each element in the oddGen generator as such:

**The Output:**

```
1
3
5
7
9
```

Slicing in Python is "a flexible tool to build new lists out of an existing list" (Boiko & Boiko, 2021). Python supports list slicing for any sort of abstract data type (lists, stacks,

queues, etc...) and is very popular in the pandas and NumPy library (Boiko & Boiko, 2021). The syntax for list slicing is: `list[start,stop,step]` where "start" is the

beginning of the slice, "stop" is the end of the slice, and "step" is the every "nth" element of the list. For example, when you have the method "sliceList" where you have

the list "li" be `[1,"hello",2.5,3,"seng265",5,1.2,"python"]` and I want to turn it into `[1, 2.5,'seng265',1.2]` I would insert `li[0:7:2]` .

The code would look something like this:

**The Program**

```
def sliceList():
```

```
li = [1,"hello",2.5,3,"seng265",5,1.2,"python"]
li = li[0:7:2]
print(li)
```

**Output:**

```
[1, 2.5,'seng265',1.2]
```

Overall, Python comprehension is an extremely important aspect to know if you are using the Python programming language. As a student, I have not yet learned about sets and generators in Python

prior to this term portfolio project. This lack of knowledge encouraged me to research these types of comprehensions and try to apply them to my own projects/assignments

# VI. Advantages of Classes for Packaging, Encapsulation, and the Significance of Object-Oriented Design

Object-Oriented Programming (OOP) is an extremely popular computer programming model that orgranizes programs and softwares to focus on the data and logic instead of the function and logic. OOP

is a useful program model that popular languages like Java, Python, and C++ which all focus on an object-oriented design. OOP simplifies the programming process for complex programs that are

large, complex, and routinely updated (Gillis & Lewis, 2021). Classes are a blueprint of sorts for objects that perform certain functions for certain problems. Encapsulation is

one of the fundamentals of OOP and refers to the methods that bundle and operate on a specific piece of data in a class. The main purpose of encapsulation is used to hide

values of a data object inside of a class from unauthorized users. By implementing encapsulation, it becomes difficult for unauthorized parties to gain direct access to this

data and creates a layer of security (Braunschweig, 2018).Three advantages for encapsuation are:

**Better Control Over Class Data:** Helps the authors of the program achieve better control over class data by hiding the internals of the class making it harder for unauthorized resources to view the data

**Improved Code Maintainabiliity:** Reduces the complexity of a program by isolating the implementation details of a class making it easier to modify or update the code without affecting other parts of the program

**Increased Code Reusability:** Encapsulation of classes makes it simpler to reuse them in different parts of the program reducing the amount of code that needs to be written.

OOP is extremely helpful and important in today's world where the majority of the popular programming languages have similiar or direct OOP programs integrated within the language. According to Half (2023), there are four

main advantages to OOP:

1. **Reusability:**
   OOP makes code much more reusable compared to programs without OOP, making it much more time-efficient

2. **Maintainability:**
   OOP has a modular structure which makes it extremely easy to update and maintain code, allowing developers to modify or update code without impacting the rest of the program

3. **Flexibility:**
   OOP's ability to add or remove components as needed without causing major aids with the creation of highly adaptable software.

4. **Security:**
   The encapsulation feature, as discussed earlier, creates a layer of security to a program that uses OOP to secure sensitive data within classes.

After taking Seng 265, I learned that the readability and accessibliity portions of a program are extremely important. By utilizing OOP principles, developers can create code that is not only

more readable and accessible but also easier to maintain and update in the future which is why

# VII. The Concepts and Applications of Objects, Class Variables, and Methods

Objects and class variables are all valuable pieces incorportated within Object-Oriented Programming (OOP). Objects are instances of a class created with specifically designed data with their own set of

characteristics and behaviours (Doherty, 2015). Class variables are variables that are shared between all objects within a specific class. Changing the value of a class

variable in one object will change the value of that class variable for every single object within the class. Methods are functions within a class that can be called inside

of an object to perform some sort of action on a variable (Methods (OOP) | Brilliant Math & Science Wiki, n.d.). OOP is extremely powerful and helpful in the world of

programming. According to Cheguri (2022), a few popular applications of OOP are: client-server

systems, hyptertext, hyptermedia, and application design. Within my

programming experience so far, I have not really used OOP as I never saw the need for it in my classes. I was already familiar with these concepts, however, while researching them for

this project I reviewed original content and recieved a deeper understanding and appreciation for OOP

## VIII. Understanding Deep Copy and Shallow Copy in Python: The Purpose of copy() and deepcopy() Methods in the Python Copy Module

One of the reasons Python is so popular is because of the vast amount of libraries and modules that can be imported into Python. One of these modules is the copy module, assignment

statements in Python do not actually copy the object but instead creates bindings between a target and an object. Mutable collections occassionaly use copy so a developer

can change one copy without changing the other (Copy — Shallow and Deep Copy Operations, n.d.). The copy module creates two different types of copies, shallow or deep.

There are two different types of functions in the copy module, .copy(x) which returns a shallow copy of x and .deepcopy(x [,memo] ) which returns a deep copy of x (Copy —

Shallow and Deep Copy Operations, n.d.). There is a difference between shallow and deep copy, however, the difference is only important when talking about compound objects

(objects that contain other objects).

**Shallow Copy:** A copy of an object whose properties share the same memory references as the original "x" being copied

**Deep Copy:** A copy of an object whose properties do not share the same memory references as the original "x" being copied

Here is an example showcasing the difference between deep and shallow copy:

```
import copy
    list = [1,2,[3,4]]

    shallowCopy = copy.copy(list)

    deepCopy = copy.deepcopy(list)

    list = [2][0] = 5

    print("Original list: ", list)
```

```
    print("Shallow copy list: ", shallowCopy)

    print("Deep copy list: ", deepCopy)
```

Output:

```
 Original list:  [1, 2, [5, 4]]

Shallow copy list:  [1, 2, [5, 4]]

Deep copy list:  [1, 2, [3, 4]]
```

As you can see with the shallow copy and original list, the value three got updated to five but in the deep copy the value did not get updated. This is because the elements in shallow copy uses

the same memory address as the original list, but deep copy creates new memory addresess for each element which is why the deep copy list did not get updated.

I have never had any experience with the copy module in Python, and I can not imagine many scenarios where it is useful. However, if I ever find a problem that would need to use the copy

module, I will be able to use it efficiently after researching the topic.

# IX. The Notion of and Motivation for Typing and Typing Hints in Python

Type hints are a very popular feature in python that allows developrs to annotate the types of function parameters and return values which makes the code much more readable. Type hints are

extremely useful for developers in all sorts of major projects with either multiple teams or mulitple individuals working on it together. Type hints are extremely beneficial

to a program. Typhints can document your code, makes the code easier to read and debug, and certain IDE's can use type hints to also help debug and analyize your code.

**Example of a program that returns the number of even numbers in a list without type hints:**

```
 def howManyEven(list):


    count = 0


    for i in list:
```

```
        if(i%2==0):

            count+=1

    return count
```

**The same example but with type hints:**

```
def howManyEven(list: List[int]) -> int:

    count = 0

    for i in list:

        if(i%2==0):

            count+=1

    return count
```

As you can see, the type hints make the program an effortless read as you can see exactly what the variable "list" is and what type of value the program returns.

I never used type hints before Seng 265, but a big part of the Seng 265 assignments is the readbility of the code. I lost a lot of marks in the first assignment due to my lack of type hints

and how my code appeared overly complex. Over the semester I have worked towards using more type hints and making my code readable which is reflected in my grade where

with the first assignment I got 65% and on the second assignment I got 85%.

# X. How Challenging Assignment Three Was For Me

The whole point of Assignment three was to get flight information in a YAML file and output the specific N airlines, countries with the least appearances, and destination airports.

The three questions we were supposed to answer were:

**q1:** What are the top N airlines that offer the greatest number of routes with destination the country as Canada?

**q2:** What are the top N countries with the least appearances as destination country on the routes data?

**q3:** What are the top N destination airports?

I found this assignment to be the easiest out of all the assignments I have done so far because of the help I got throughout my labs. The biggest struggle I had was actually extracting

the information from the YAML file. The hint that helped me find the information was that there were always 13 elements in each route in the YAML file. Another issue

that was prominent were the segmentation faults I would get because of my issue with pointers. I was not very confident with using pointers before the assignment, however,

after spending 10-13 hours on this assignment I feel much more confident with my C programming skills and understand pointers to a level where I feel like I would not need to

Google the errors I recieved with pointers.

# XI. References

Aziz, K. A. (2021, December 14). Learn How to Write Markdown & LaTeX in The Jupyter Notebook. Medium. https://towardsdatascience.com/write-markdown-latex-in-the-jupyter-notebook-10985edb91fd#:~:text=The%20Jupyter%20Notebook%20uses%20MathJax,by%20writing%20bet (https://towardsdatascience.com/write-markdown-latex-in-the-jupyter-notebook-10985edb91fd#:~:text=The%20Jupyter%20Notebook%20uses%20MathJax,by%20writing%20bet

Lambert W's Taylor Series. (n.d.). Overleaf, Online LaTeX Editor. https://www.overleaf.com/articles/lambert-ws-taylor-series/dzbvsrzvgkvn (https://www.overleaf.com/articles/lambert-ws-taylor-series/dzbvsrzvgkvn)

Mirzaee, A. (n.d.). Markdown and LaTeX introduction. https://ashki23.github.io/markdown-latex.html (https://ashki23.github.io/markdown-latex.html)

Perveez, S. H. (2023, January 30). What is Git: Features, Command and Workflow in Git. Simplilearn.com. https://www.simplilearn.com/tutorials/git-tutorial/what-is-git#:~:text=What%20is%20Git%3F&text=Git%20is%20a%20DevOps%20tool,together%20on%20 (https://www.simplilearn.com/tutorials/git-tutorial/what-is-git#:~:text=What%20is%20Git%3F&text=Git%20is%20a%20DevOps%20tool,together%20on%20

Perveez, S. H. (2023b, January 30). What is Git: Features, Command and Workflow in Git. Simplilearn.com. https://www.simplilearn.com/tutorials/git-tutorial/what-is-git#:~:text=Git%20is%20a%20DevOps%20tool,together%20on%20non%2Dlinear%20developme

(https://www.simplilearn.com/tutorials/git-tutorial/what-is-git#:~:text=Git%20is%20a%20DevOps%20tool,together%20on%20non%2Dlinear%20developme

Ravoof, S. (2022, November 28). GitLab vs GitHub: Explore Their Major Differences and Similarities. Kinsta®. https://kinsta.com/blog/gitlab-vs-github/#:~:text=Both%20GitLab%20and%20GitHub%20are,%2C%20collaboration%2C%20and%
(https://kinsta.com/blog/gitlab-vs-github/#:~:text=Both%20GitLab%20and%20GitHub%20are,%2C%20collaboration%2C%20and%

Software Developers, Quality Assurance Analysts, and Testers : Occupational Outlook Handbook: : U.S. Bureau of Labor Statistics. (2023, February 6). https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm#:~:text=Software%20developers%20typically%20do%20the,the%20pieces%20wi
(https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm#:~:text=Software%20developers%20typically%20do%20the,the%20pieces%20wi

Software Developers, Quality Assurance Analysts, and Testers : Occupational Outlook Handbook: : U.S. Bureau of Labor Statistics. (2023, February 6). https://www.bls.gov/ooh/computer-and-information-technology/softwaredevelopers.htm#:~:text=Software%20developers%20typically%20do%20the,t
(https://www.bls.gov/ooh/computer-and-information-technology/softwaredevelopers.htm#:~:text=Software%20developers%20typically%20do%20the,t

Thompson, B. (2023, March 4). What is C Programming Language? Basics, Introduction, History. Guru99. https://www.guru99.com/c-programming-language.html (https://www.guru99.com/c-programming-language.html)

TIOBE Index - TIOBE. (2022, June 3). TIOBE. https://www.tiobe.com/tiobe-index/ (https://www.tiobe.com/tiobe-index/)

What is Python? Executive Summary. (n.d.-b). Python.org. https://www.python.org/doc/essays/blurb/ (https://www.python.org/doc/essays/blurb/)

What is the Jupyter Notebook? — Jupyter/IPython Notebook Quick Start Guide 0.1 documentation. (n.d.). https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html (https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html)

Boiko, S., & Boiko, S. (2021, August 17). Python Indexing and Slicing for Lists and other Sequential Types | Railsware Blog. Blog by Railsware | Blog on Engineering, Product Management, Transparency, Culture and Many More. . . https://railsware.com/blog/python-for-machine-learning-indexing-and-slicing-for-lists-tuples-strings-and-other-sequential-types/#:~:text=In%20short%2C%20slicing%20is%20a,add%20its%20support%20as%20well
(https://railsware.com/blog/python-for-machine-learning-indexing-and-slicing-for-lists-tuples-strings-and-other-sequential-types/#:~:text=In%20short%2C%20slicing%20is%20a,add%20its%20support%20as%20well)

GeeksforGeeks. (2018, November 14). Comprehensions in Python. https://www.geeksforgeeks.org/comprehensions-in-python/ (https://www.geeksforgeeks.org/comprehensions-in-python/)

Python Dictionary Comprehension. (n.d.). https://www.programiz.com/python-programming/dictionary-comprehension (https://www.programiz.com/python-programming/dictionary-comprehension)

Gillis, A. S., & Lewis, S. (2021, July 13). object-oriented programming (OOP). App Architecture. https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP (https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP)

Doherty, E. (2015, April 15). What is object-oriented programming? OOP explained in depth. educative.io. https://www.educative.io/blog/object-oriented-programming#building (https://www.educative.io/blog/object-oriented-programming#building)

Methods (OOP) | Brilliant Math & Science Wiki. (n.d.). https://brilliant.org/wiki/methods-oop/ (https://brilliant.org/wiki/methods-oop/)

Cheguri, P. (2022, December 30). Top 10 Applications of Object-Oriented Programming Using Python Language. Analytics Insight. https://www.analyticsinsight.net/top-10-applications-of-object-oriented-programming-using-python-language/ (https://www.analyticsinsight.net/top-10-applications-of-object-oriented-programming-using-python-language/)

Braunschweig, D. (2018, December 15). Encapsulation. Pressbooks. https://press.rebus.community/programmingfundamentals/chapter/encapsulation/ (https://press.rebus.community/programmingfundamentals/chapter/encapsulation/)

Half, R. (2023, February 18). 4 Advantages of Object-Oriented Programming. https://www.roberthalf.com/blog/salaries-and-skills/4-advantages-of-object-oriented-programming (https://www.roberthalf.com/blog/salaries-and-skills/4-advantages-of-object-