Pass task 7.1

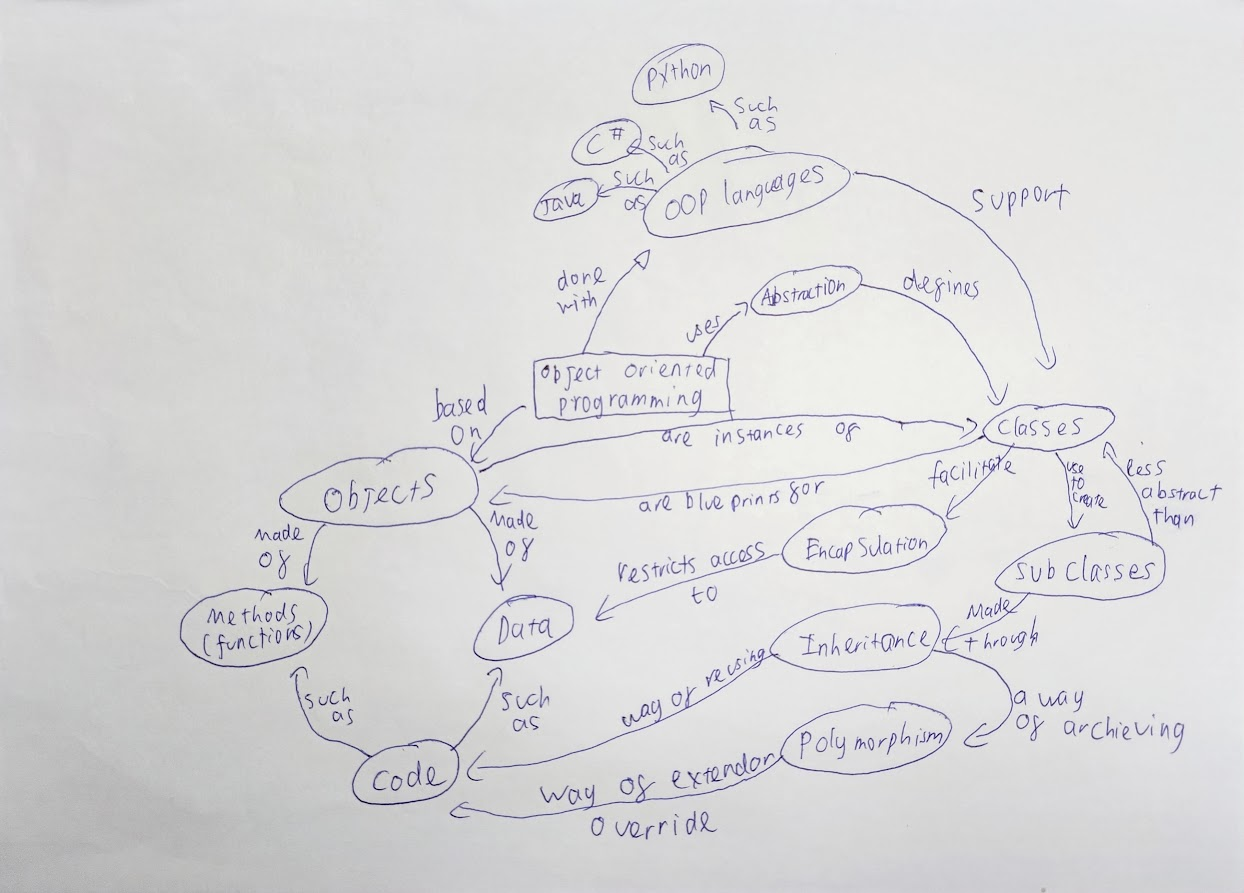
Name: Tran Hoang Hai Anh

StudentID: 104177513

Overview:

A programming paradigm known as object-oriented programming (OOP) divides code into objects, which are instances of classes. OOP principles offer recommendations for creating modular, adaptable, and maintainable software systems.

This is my concept map created by myself:



1. Encapsulation

In order to hide internal information and provide a public interface for interaction, data and methods are bundled together into a class and are said to be encapsulated. While limiting direct access and allowing for no change of internal state, it enables data abstraction. By separating implementation-related details from other concerns and enhancing code maintainability and reusability, encapsulation.

1. Inheritance

By creating a "is-a" link between them, inheritance makes it possible to create new classes that are based on already existing classes. Code reuse is encouraged via inheritance, which also enables the specialized and expanded usage of already implemented functionality. Because similar characteristics and behaviors are inherited from a base class, it makes it easier to create class hierarchies, which encourages modular design and reduces repetition.

1. Polymorphism

Objects of several classes can be used interchangeably because polymorphism enables them to be viewed as instances of a single base class. By offering an interface for dealing with things that are consistent across all sorts of objects, polymorphism makes coding easier. Because methods may be implemented in several ways depending on the behavior of the particular object, it encourages code flexibility and extension.

1. Abstraction

Abstraction emphasizes conveying crucial characteristics and actions while obscuring unimportant information. Developers may use it to make abstract classes or interfaces that define a contract without mentioning implementation specifics. Abstraction gives a high-level perspective of the system and simplifies and makes a system more modular. Through interfaces, it also permits many implementations, encouraging loose coupling and the division of responsibilities.

Conclusion:

Software engineers may write well-organized, modular, and extendable code by using these ideas. OOP promotes code reuse, maintainability, and scalability while providing a strong basis for creating complex systems. It makes it possible for teams to work together productively, assign tasks, and create software in a more structured and efficient way.