Object Oriented Programming
Quiz 1

Name
Dicha Zelianivan Arkana

NIM
2241720002

Class
2i

Department
Information Technology

Study Program
D4 Informatics Engineering

Quiz 1

Class and Object

- What does the word **class** mean in object-oriented programming?

  A **class** is a blueprint for creating objects (a particular data structure) which can contain states and behaviours. In other words, a class is a template or blueprint about the capability of what an object can do.

- How do you define an object of a class in Java programming language?

  To define an object of a class in Java programming language, we can use the following syntax:

  ```
  ClassName objectName = new ClassName();
  ```

  This process is also known as *instantiation*.

- Suppose you have a class `Item` in an inventory system. How would you make a `laptop` object from that class?

  To make a `laptop` object from the `Item` class, we can use the following syntax:

  ```
  Item laptop = new Item();
  ```

Encapsulation

- Explain the concept of encapsulation in object-oriented programming and its importance in the development in an inventory system.

  Encapsulation is a mechanism to hide data from the outer world. It is a way to achieve data hiding in Java because other class will not be able to access the data through the private data members. The outer world can only access / modify them through the provided getter and setter methods.

  It is important to use encapsulation in the development of an inventory system because it can protect the data from unwanted changes. In other words, it prevents an object from having an invalid or inconsistent state.

- In the context of inventory system, mention an example of an attribute that should be encapsulated and explain why.

  An example of an attribute that should be encapsulated is the `stock` attribute. We need to have a setter method for this attribute to prevent the stock from having a negative value.

Class Relationship

- What does it mean by **class relationship** in object-oriented programming?

  Class relationship is a way of describing how objects are related to each other. A class can contain another class as a field. This is known as **composition**. Other type of class relationship is **inheritance**.

- In an inventory system, how would you describe the relation between the class `Item` and `Category`?

  The relation between the class `Item` and `Category` is **composition**. An item can have a category, but a category cannot have an item. It can be illustrated as such:

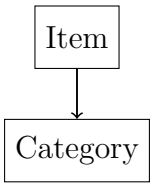

Figure 1: Class relationship between `Item` and `Category`

PBL

- Based on the case study of the inventory system, try to create a single class along with its attributes and methods that describe an entity in the case study. For example, the class `Item`.

```java
public class Item {
    private String name;
    private String description;
    private int stock;

    public Item(String name, String description, int stock, int price) {
        this.name = name;
        this.description = description;
        this.stock = stock;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) throws IllegalArgumentException {
        if (name == null) {
            throw new IllegalArgumentException("Name cannot be null");
        }
```

```
        if (name.length() < 3) {
            throw new IllegalArgumentException("Name must be at least 3 characters long");
        }
        this.name = name;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) throws IllegalArgumentException {
        if (description == null) {
            throw new IllegalArgumentException("Description cannot be null");
        }
        if (description.length() < 10) {
            throw new IllegalArgumentException(
                "Description must be at least 10 characters long"
            );
        }
        this.description = description;
    }

    public int getStock() {
        return stock;
    }

    public void setStock(int stock) {
        if (stock < 0) throw new IllegalArgumentException("Stock cannot be negative");
        this.stock = stock;
    }
}
```

Item
name : String description : String stock : int
getName() : String getDescription() : String getStock() : int setName(name : String) : void setDescription(description : String) : void setStock(stock : int) : void

- How would you use *encapsulation* to protect the class that you've created?

  We can use *encapsulation* to protect the class that we've created by making the attributes private and provide getter and setter methods for them. This is used to guard against invalid or inconsistent state of an object. For example, the field `name` and `description` cannot be null and must be at least 3 and 10 characters long respectively. The field `stock` cannot be negative.

- Give an example of a *class relationship* in the inventory system.

  An example of a *class relationship* in the inventory system is the relationship between the `Item` and `Category` class. An item can have a category, but a category cannot have an item. It can be illustrated as such: