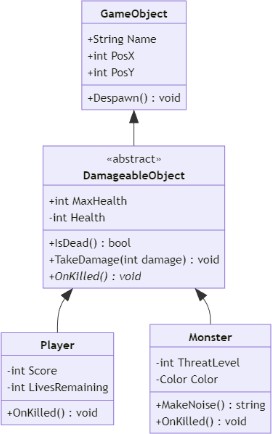
QUIZ QUESTIONS 2

OBJECT-BASED PROGRAMMING PRACTICUM

1. Identify the following Abstract method and Class usage, explain the purpose of the diagram class and create the program code to the demo to display it.



Answer :

1. This system is designed to organize various objects in the game that can be destroyed or interacted with by the player. In this system, there are four main classes: GameObject, DamageableObject, Player, and Monster.

The GameObject class acts as a base class that contains common attributes and methods used by all objects in the game. DamageableObject is an abstract class that is derived from GameObject, with additional features to handle damage and death status of objects. Meanwhile, Player and Monster are derived classes from DamageableObject that have special attributes and functions according to their respective roles.

1. Class Diagram :
2. GameObject

* Atribut:
* Name : String – Name of the object.
* PosX : int – X position of the object.
* PosY : int – Y position of the object.
* Methods:
* Despawn() : Removes an object from the game.

1. DamageableObject (Abstract Class)

* Attributes:
* MaxHealth : int – Maximum amount of health.
* Health : int – Current amount of health.
* Abstract Methods:
* TakeDamage(int damage) : Reduces the Health value.
* OnKilled() : Defines the action when the object dies.
* Non-Abstract Methods:
* IsDead() : Checks whether the Health value is zero or less.

1. Player

* Attributes:
* Score : int – Player score.
* LivesRemaining : int – Number of player lives remaining.
* Methods:
* OnKilled() : Implements the logic when the player runs out of lives.

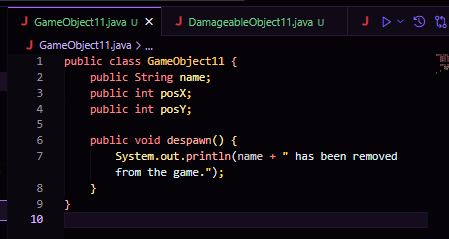
1. Monster

* Attributes:
* ThreatLevel : int – Monster threat level.
* Color : String – Monster color.
* Methods:
* MakeNoise() : Generates monster sound.
* OnKilled() : Implements the logic when the monster dies.

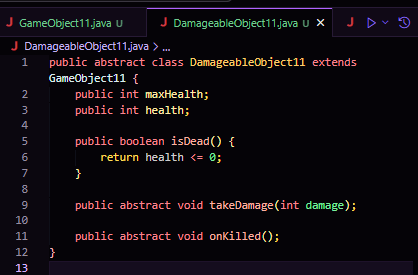
1. Relationship Between Classes:

* GameObject is the parent class for DamageableObject.
* DamageableObject is an abstract class that is the basis for Player and Monster.
* The Player and Monster classes have specific implementations for the OnKilled() abstract method according to their respective functions.

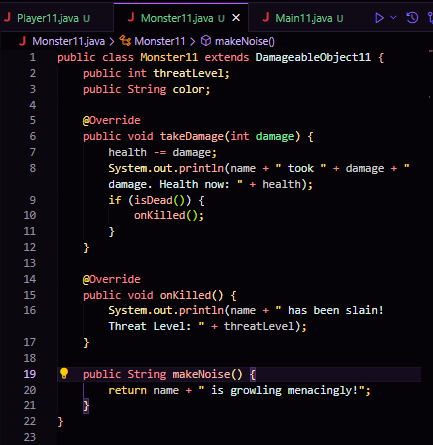
1. Java Program Code
2. GameObject



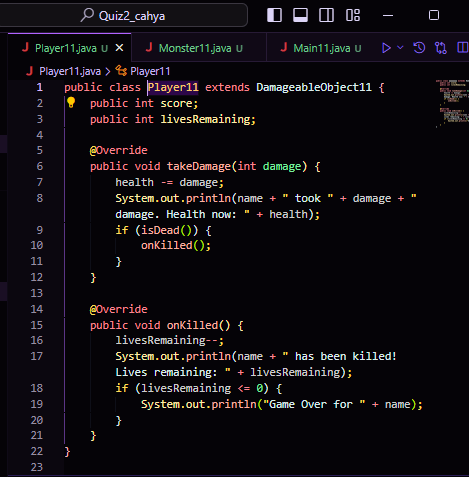
1. DamageableObject



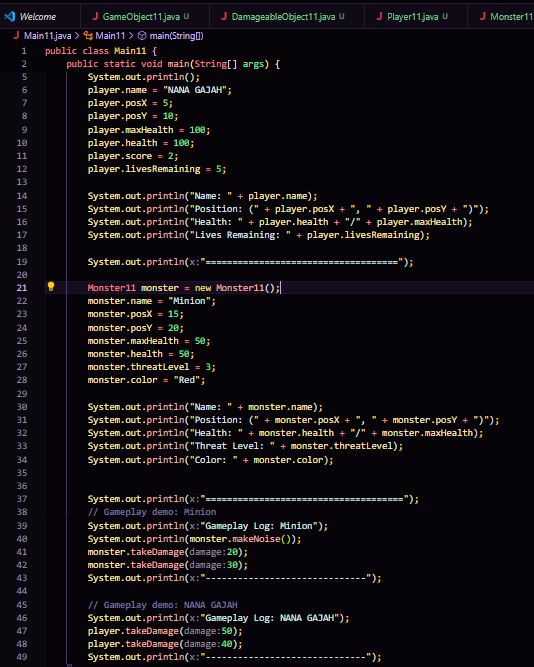
1. Monster



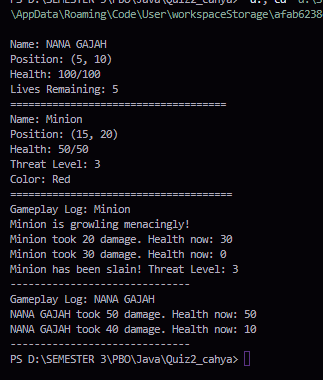
1. Player



1. Main



1. Output



1. A client of yours is a Seller who has a lot of media to accommodate orders from customers, but this Seller has difficulty in creating Order categories, he wants every order to have an order date and there must be a confirmation method for each category which is separated into 3 classes: MailOrder, WebOrder, WhatsappOrder. There is an "order status tracking" contract on the MailOrder and WebOrder classes

Help your client by describing his diagram classes that are easy for him to understand!

Answer :

1. This class diagram explains that Order is a base abstract class for various types of orders, with an orderDate attribute and an abstract confirmOrder() method, while MailOrder, WebOrder, and WhatsappOrder are subclasses that each implement the trackStatus() method (for MailOrder and WebOrder) or just confirmOrder() (for WhatsappOrder).
2. Class Diagram:
3. Order (Abstract Class)

* Attributes:
* orderDate : String – Order date.
* Methods:
* confirmOrder() : Abstract method to confirm an order.

1. MailOrder (Class)

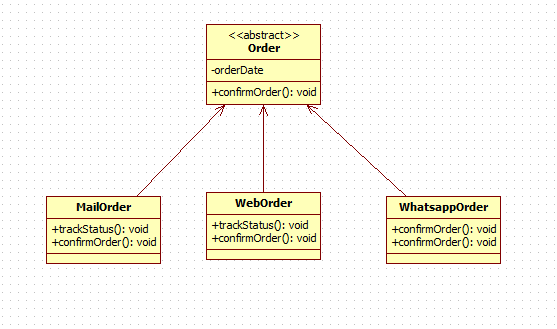
* Atribut:
* (Mewarisi atribut dari Order)
* Methods:
* confirmOrder() : Mengonfirmasi pesanan melalui surat.
* trackStatus() : Melacak status pesanan.

1. WebOrder (Class)

* Atribut:
* (Inherit attributes from Order)
* Methods:
* confirmOrder() : Confirms orders via website.
* trackStatus() : Tracks order status.

1. WhatsappOrder (Class)

* Attributes:
* (Inherit attributes from Order)
* Methods:
* confirmOrder() : Confirms orders via WhatsApp.

1. Class Diagram
2. Explanation:

The Order class is abstract, containing an orderDate attribute and an abstract method confirmOrder(). Its subclasses MailOrder, WebOrder, and WhatsappOrder provide implementations for the confirmOrder()method. Additionally, MailOrder and WebOrder include a trackStatus() method, whereas WhatsappOrder solely implements the confirmOrder() method.

1. Give an example of program code using the concept of polymorphism (Heterogenous Collection, Object Casting, Polymorphic Arguments,

InstanceOf) on 1 theme (for example, choose 1 theme: vehicle or electronic device or animal, etc... You can create any theme to apply the 4 points of polymorphism). Create interrelated java program code.

Answer :

1. This system is designed to manage various types of buildings and calculate their construction costs efficiently. There are three main classes: Building, House, Building, and Mall.

Building is an abstract class that stores general information such as the name and area of ​​the building, and a method for calculating construction costs.

House, Building, and Mall are subclasses that each add more specific information, such as the number of rooms, the number of floors, and the number of shops.

1. Class
2. Building (Abstract Class)

* Attributes :
* name: Name of the building
* area: Area of ​​the building in m²
* Methods:
* calculateCost(): Abstract method for calculating construction costs
* getName(): Gets the name of the building
* getArea(): Gets the area of ​​the building

1. House (Building Subclass)

* Attributes:
* numberofRooms: Number of rooms in the house
* Methods:
* calculateCost(): Calculate the construction cost of a house based on the area
* infoHouse(): Display the number of rooms in the house

1. Building (Building Subclass)

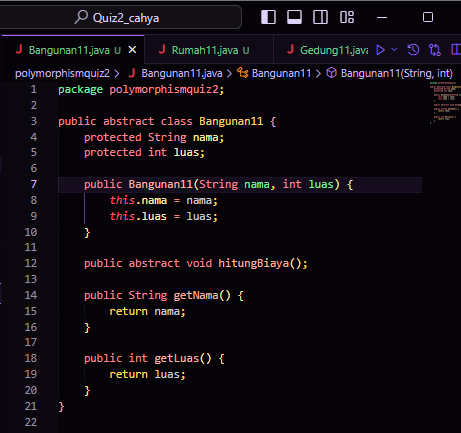
* Attributes:
* numberOfFloors: Number of floors in the building
* Methods:
* calculateCost(): Calculate the construction cost of a building based on the area and floors
* infoGedung(): Display the number of floors in the building

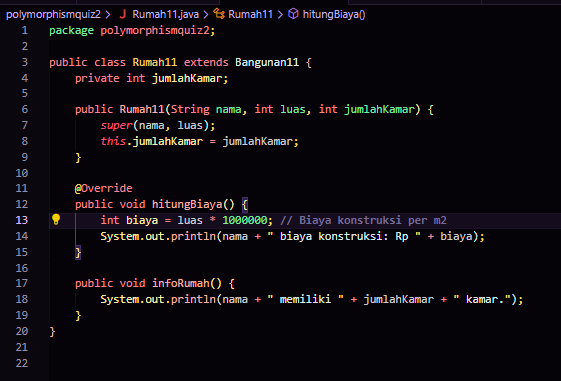
1. Mall (Building Subclass)

* Attributes:
* numberOfStores: Number of stores in the mall
* Methods:
* calculateCost(): Calculate the construction cost of a mall based on the area
* infoMall(): Display the number of mall stores

1. Relationship Between Classes

* Building is an abstract parent class with name and area attributes, and methods hitungBiaya(), getNama(), and getLuas().
* Rumah, Gedung, and Mall are subclasses of Bangunan that add specific attributes and implement the hitungBiaya() method to calculate construction costs according to their type.

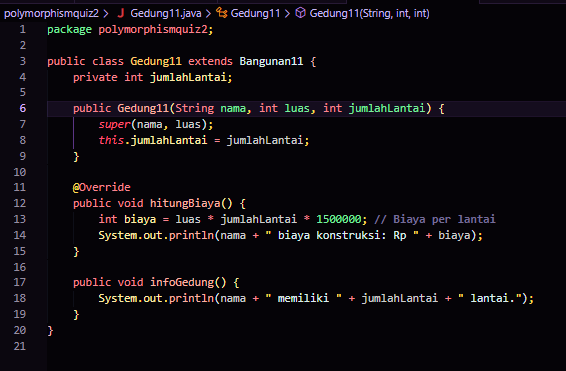
1. Java Program Code
2. Bangunan
3. Rumah



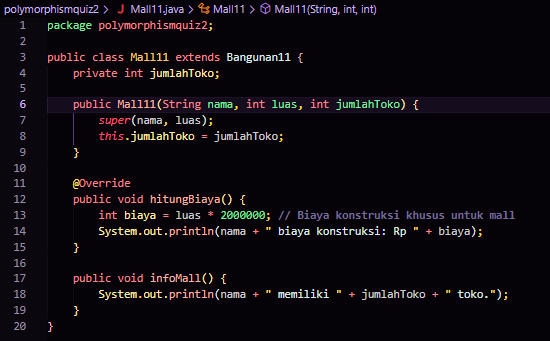
**--**

**--**

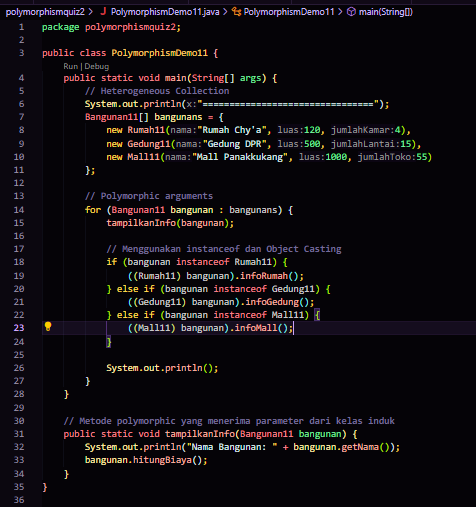
1. Gedung

****

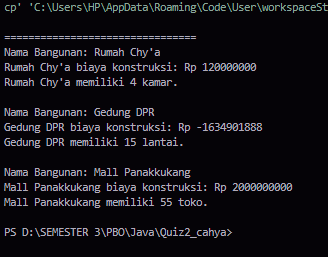
1. Mall

****

1. PolymorphismDemo



1. Output



1. Code Explanation

* Heterogeneous Collection
* Bangunan[] Bangunans stores objects from the subclasses: Houses, Buildings, and Malls.
* Object Casting
* Object type checking is done with instanceof, then the object can be converted (cast) to the appropriate type, such as House, Building, or Mall.
* Polymorphic Arguments
* The method tampilkanInfo accepts a parameter from the parent type Building.
* InstanceOf
* Used to check the object type, ensuring that the object being processed has the appropriate type before being converted to a specific type.

**Good Luck ----**