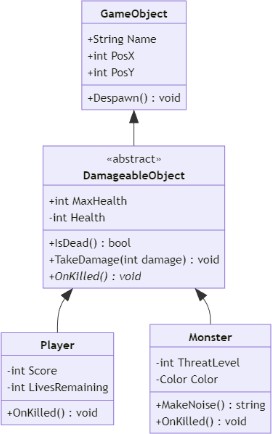
QUIZ QUESTIONS 2

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.



* Analisis Diagram

**Class GameObject:**

kelas ini berfungsi untuk menyimpan atribut umum seperti String Name, int PosX, int PosY

**Class DamageableObject:**

Kelas ini merepresentasikan objek seperti MaxHealth dan Health mengelola status Health dari object, IsDead() mengecek apakah objek sudah mati (Boolean), TakeDamage(int) ini untuk mengurangi nilai health berdasarkan int damage yang diterima, OnKilled(), ini adalah metod abstrak yang akan diimplementasikan oleh child classnya

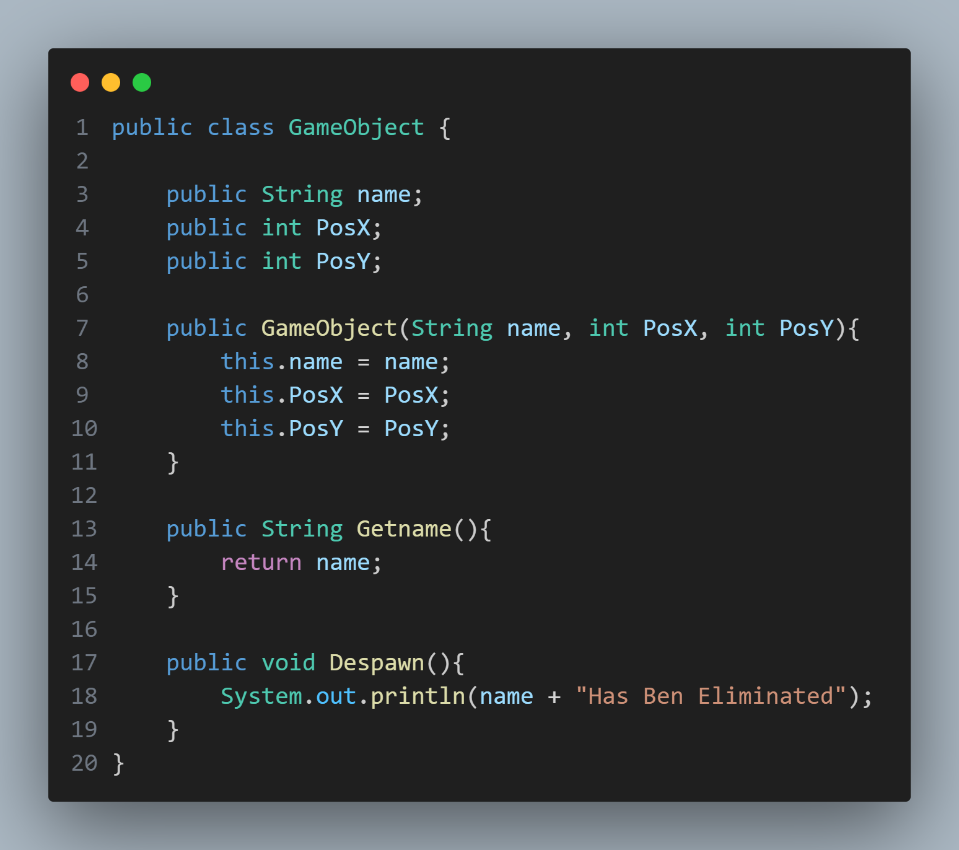
**Class Player:**

Class ini memiliki atribut tambahan seperti Score dan LiverRemaining, pada kelas ini implementasi dari abstrak class yaitu method OnKilled()

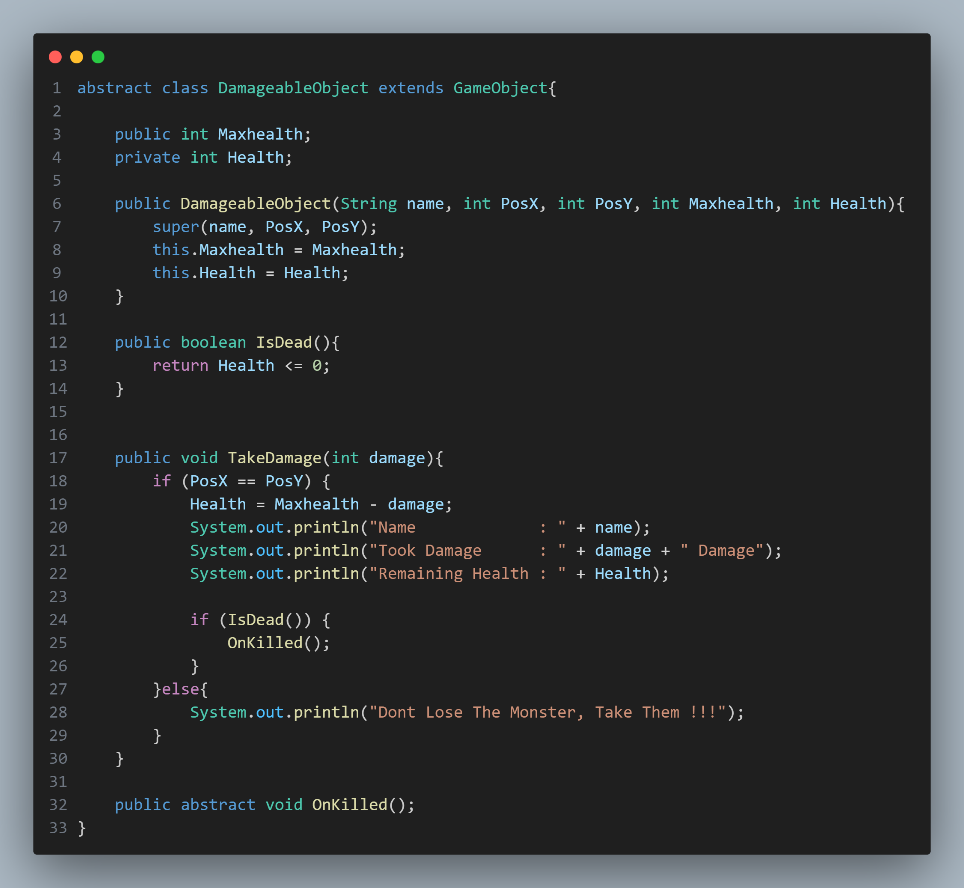
**Class Monster:**

Class ini memiliki atribut tambahan sepert ThreatLevel, dan color, pengimplementasian dari abstak method yaitu OnKilled() dan memiliki method tambahan yaitu MakeNoise

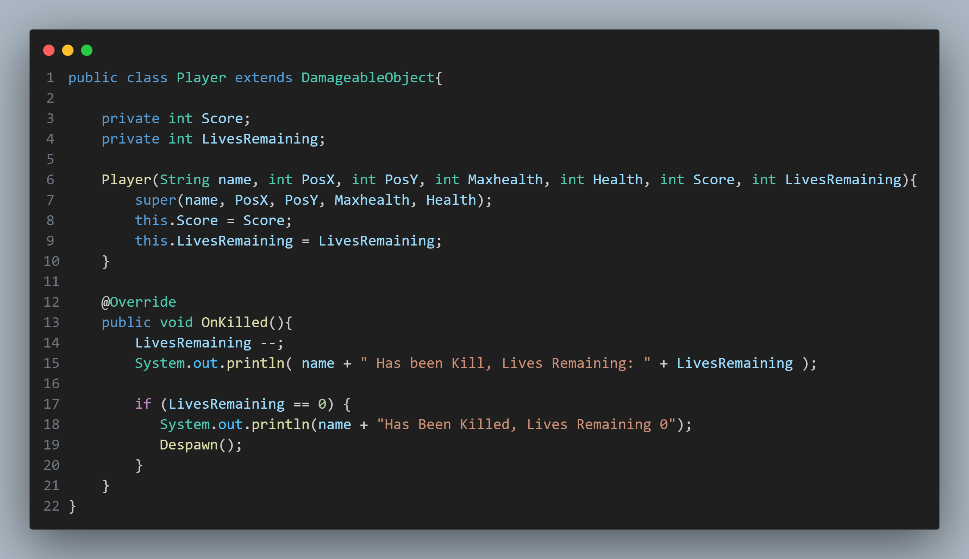
* Implementasian Code:
* **GameObject**

****

* **DamageableObject**

****

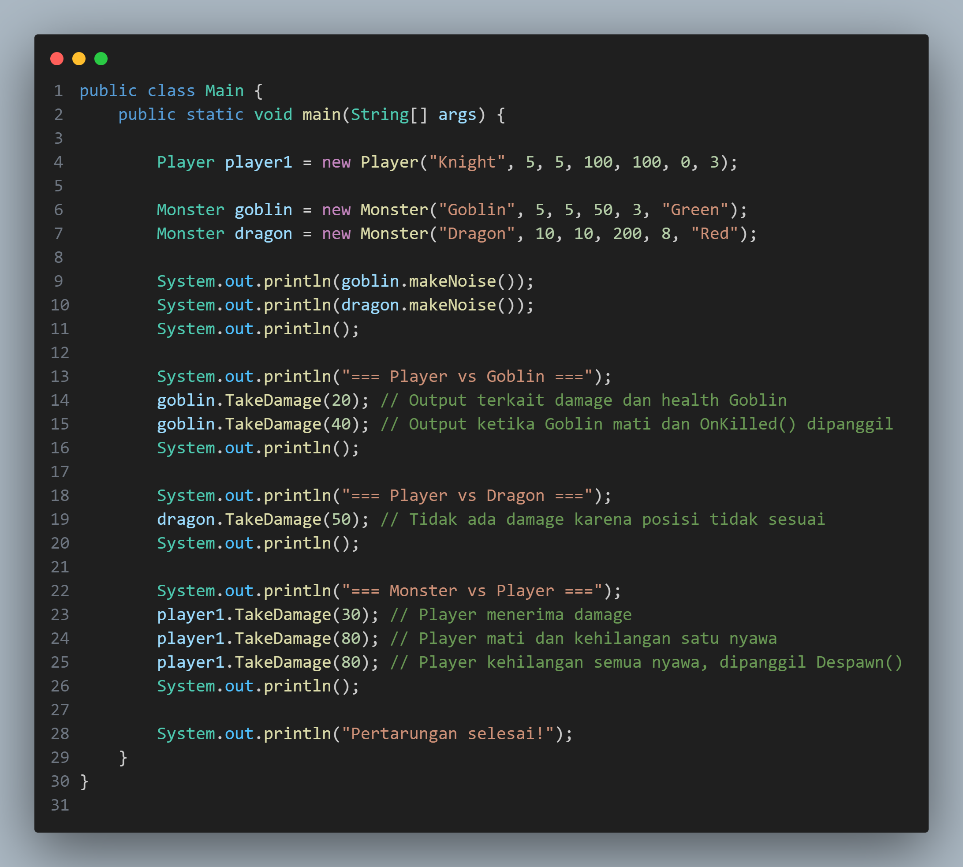
* **Player**

****

* **Monster**

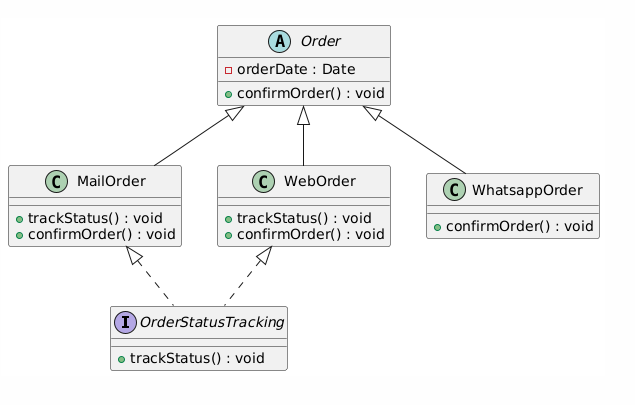
****

* **Main**

****

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!

* 

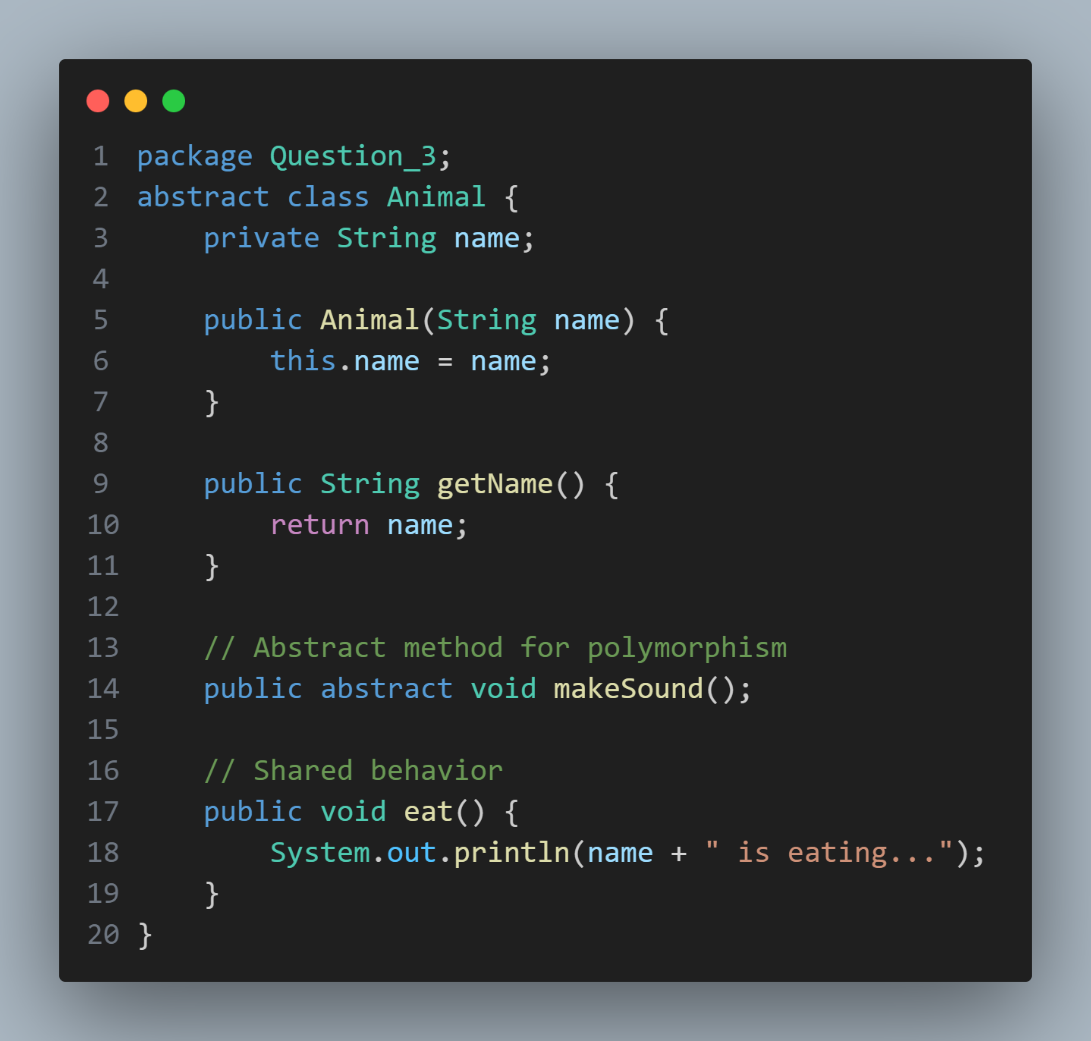
**Explanation of Diagram**

* **Order Class**:
  + Acts as the parent class and enforces a common structure for all order types.
  + Includes a field for orderDate and an abstract method confirmOrder().
* **MailOrder and WebOrder Classes**:
  + Extend the Order class and implement the OrderStatusTracking interface.
  + Provide concrete implementations for confirmOrder() and trackStatus().
* **WhatsappOrder Class**:
  + Extends the Order class but does not implement OrderStatusTracking, as the requirement states that it does not need "order status tracking".
* **OrderStatusTracking Interface**:
  + Ensures that classes implementing it provide functionality for tracking order status.

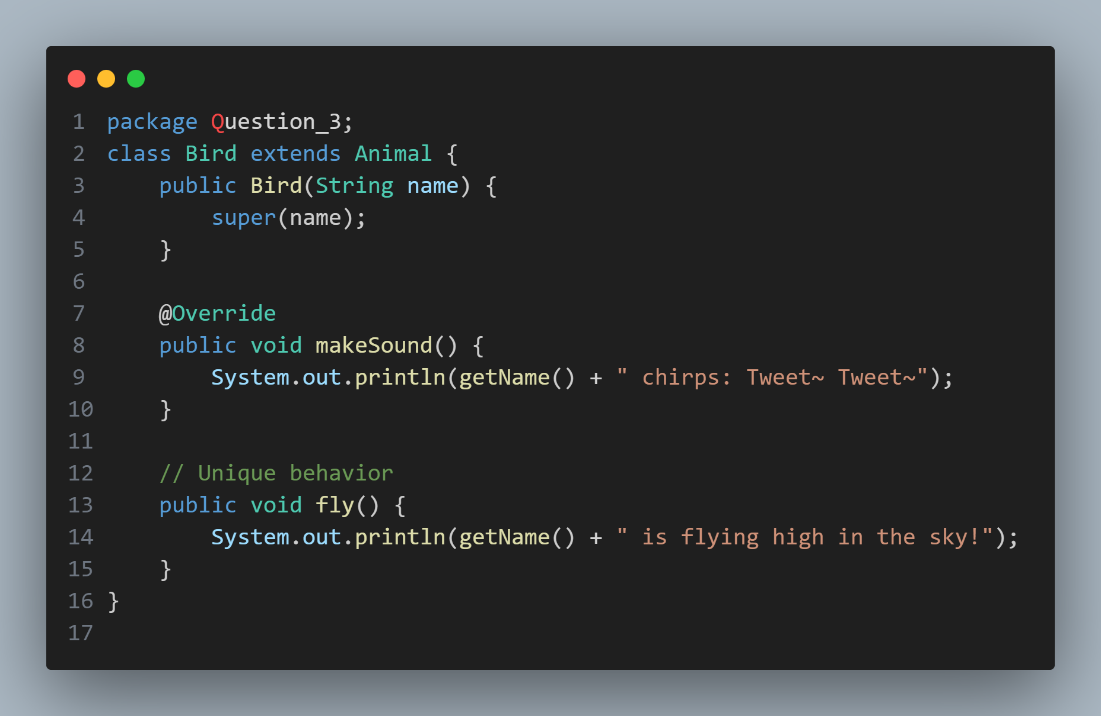
1. Give an example of program code using the concept of polymorphism (Heterogenous Collection, Object Casting, Polymorphic Arguments,

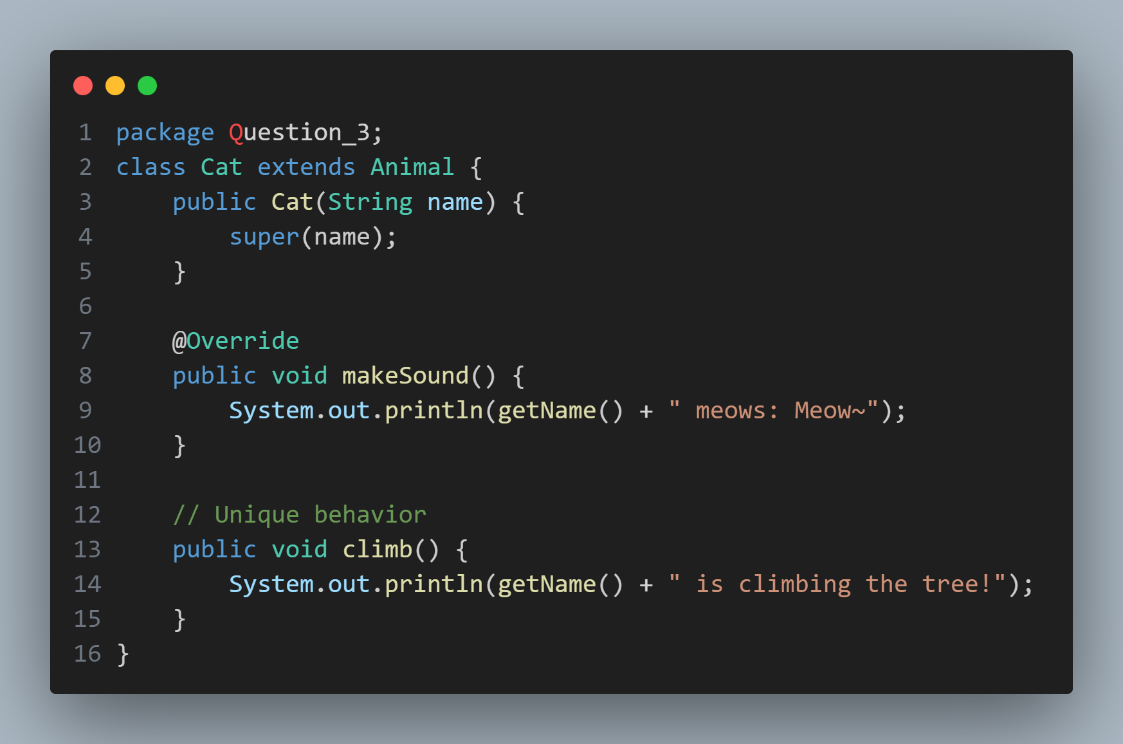
InstanceOf) on 1 theme (for example, choose 1 theme: vehicle or electronic device or animal, etc... You can create any theme to apply the 4 points of polymorphism). Create interrelated java program code.

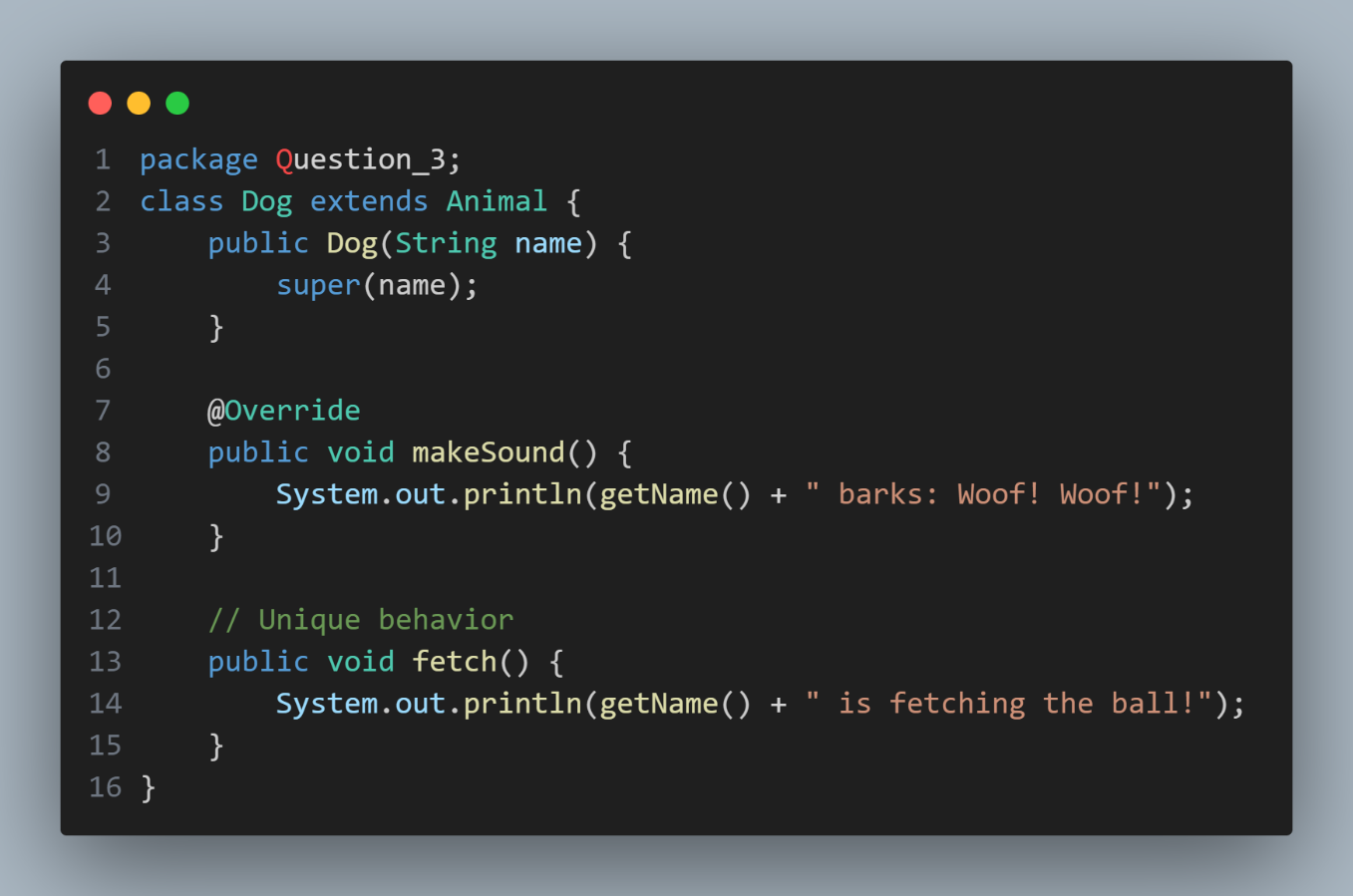
* **Class Animal**

****

* **Class Bird**

****

* **Class Cat**
* **Class Dog**

****

* **Class PolymorphismMain**

****

**---- Good Luck ----**