Inheritance and Abstraction Assignment

1. What is Inheritance in Java?

Answer: Inheritance is the mechanism in java by which one class is allowed to inherit the features of another class by using extends keyword.

1. What is superclass and subclass??

Answer: Super class is also known as parent class whose data members and method are inherited by another Child class.

Subclass is also known as child class who inherits the properties of another parent class

1. How is Inheritance implemented/achieved in Java?

Answer: By using “extends” keyword.

1. What is polymorphism?

Answer: The word “Poly” means many and “morphin” means forms so it means many forms. Polymorphism allows us to perform single action in different ways.

5. Differentiate between method overloading and overriding.

Answer:

|  |  |
| --- | --- |
| Method overloading | Method overriding |
| It is a compile-time polymorphism. | It is a runtime polymorphism. |
| It occurs within the single class. | It is performed between two or more classes with inheritance relationship. |
| It may or may not support inheritance. | It support inheritance. |
| It may have different parameter. | It should have the same parameter. |

6. What is an abstraction explained with an Example?

Answer: Abstraction is a process of hiding the implementation details from the user and showing only functionality of the user.

Example:-

abstract class AeroPlane {

    abstract void takeOff();

    abstract void fly();

}

class CargoPlane extends AeroPlane {

    public void takeOff() {

        System.out.println("CargoPlane requires longer runway");

    }

    public void fly() {

        System.out.println("CargoPlane will fly in lower height");

    }

}

class PassengerPlane extends AeroPlane {

    public void takeOff() {

        System.out.println("PassengerPlane requires medium size runway");

    }

    public void fly() {

        System.out.println("PassengerPlane will fly in medium height");

    }

}

class FighterPlane extends AeroPlane {

    public void takeOff() {

        System.out.println("FifhterPlane reqires short runway");

    }

    public void fly() {

        System.out.println("FighterPlane will fly in higher height");

    }

}

class Airport {

    public void poly(AeroPlane ref) {

        ref.takeOff();

        ref.fly();

        System.out.println("---------------------------");

    }

}

public class Q1 {

    public static void main(String[] args) {

        CargoPlane cp = new CargoPlane();

        PassengerPlane pp = new PassengerPlane();

        FighterPlane fp = new FighterPlane();

        Airport a = new Airport();

        a.poly(fp);

        a.poly(pp);

        a.poly(cp);

    }

}

7. What is the difference between an abstract method and final method in Java? Explain with an example.

|  |  |
| --- | --- |
| **Abstract Method** | **Final Method** |
| It participates in inheritance and it can be override. | It also participates in inheritance but it does not override. |
| It is declared by the abstract keyword. | It is declared by the final keyword. |
| Helps to achieve abstraction. | Helps to restrict other classes from accessing its properties and methods. |

8. What is the final class in Java?

Answer: If a class is marked as final then, the class won’t participate in inheritance, if we try to do then it would through “CompileTime error”.

9. Differentiate between abstraction and encapsulation.

Answer:

|  |  |
| --- | --- |
| Abstraction | Encapsulation |
| Abstraction is the process or method of gaining the information. | While encapsulation is the process or method to contain the information. |
| In abstraction, problems are solved at the design or interface level. | While in encapsulation, problems are solved at the implementation level. |
| Abstraction is the method of hiding the unwanted information. | Whereas encapsulation is a method to hide the data in a single entity or unit along with a method to protect information from outside. |
| We can implement abstraction using abstract class and interfaces. | Whereas encapsulation can be implemented using by access modifier i.e. private, protected and public. |
| Abstraction provides access to specific part of data. | Encapsulation hides data and the user can not access same directly (data hiding. |

10. Difference between Runtime and compile time polymorphism explain with an example.

Answer:

|  |  |
| --- | --- |
| Compile time polymorphism | Runtime polymorphism |
| In Compile time Polymorphism, the call is resolved by the compiler. | In Run time Polymorphism, the call is not resolved by the compiler. |
| It is also known as Static binding, Early binding and overloading as well. | It is also known as Dynamic binding, Late binding and overriding as well. |
| Method overloading is the compile-time polymorphism where more than one methods share the same name with different parameters or signature and different return type. | Method overriding is the runtime polymorphism having the same method with same parameters or signature but associated withcompared, different classes. |
| It is achieved by function overloading and operator overloading. | It is achieved by virtual functions and pointers. |
| It provides fast execution because the method that needs to be executed is known early at the compile time. | It provides slow execution as compare to early binding because the method that needs to be executed is known at the runtime. |