**OOPS Concepts**

1. Explain OOPS Concepts?

OOPS features are:

* Class
* Object
* Encapsulation
* Abstraction
* Inheritance
* Polymorphism

**Class:**

A class is like a master copy or blue print or plan out of which we can derive objects.

**Class Declaration:**

**Class Myclass {**

**//fields or instance variables, constructor, and method declaration**

**}**

**Object:**

An object is anything that really exists in the world.

Syntax: Creating an object from the class

Class\_name object\_name = new constructor();

**Encapsulation:**

Encapsulation is the mechanism where the data (variables) and the code (methods) that act on the data will bind together.

**For ex:** If we take a class, we write the variables and methods inside the class. Thus, class is binding them together, so class is an example for encapsulation.

**Example code:**

Class person {

//variables- data

Private String name;

Private int age;

Public void talk() {

System.out.println (“Hello, I am”+name);

System.out.println (“My age is”+age);

}

}

In the above code, the variables names and age are declared as ‘private ’. There is no way to manipulate them. Only the public method talk() can access them.

**One more example:** Java POJO (plain old object) class is the best example for encapsulation.

**Abstraction:**

Hiding the implementation complexity is called abstraction, but we can access the hidden implementation through interface.

**For ex:** The example is Fan has so many implementations like increasing the speed, decreasing the speed etc but these implementations are hidden to the external system, still we can access this implementation trough interface regulator. Regulator doesn’t have implementation.

**Java Ex:** Iterator is the best example for abstraction.

**Inheritance:**

The feature by which one class acquires the properties of an existing class is called inheritance.

But the main drawback of Java is it doesn’t support multiple Inheritances; hence to overcome this draw back Java provides us the Interface concept.

**PolyMorphism:**

The meaning of Polymorphism is something like one name many forms. The concept of polymorphism can be explained as “one interface, multiple methods”.

In java++ there are two type of polymorphism:

* Static polymorphism (or compile-time polymorphism or overloading)
* Dynamic Polymorphism (or run-time polymorphism or overriding)

**Static Polymorphism:**

* Polymorphism that is exhibited during compile-time is called static polymorphism
* In java Static polymorphism is achieved through Method overloading.
* Example:

|  |
| --- |
| Class Staticpoly {  Void product(int x, int y)  {  Sysout(“product1:”+(x\*y));  }  Void product(int x, int y, int z){  {  Sysout(“product2:”+(x\*y\*z));  }  Public static void main(String args[])  {  Staticpoly Obj=new Staticpoly();  Obj.product(5,6);  Obj.product(5,6,7);  }  } |

**Dynamic polymorphism:**

* Polymorphism that is exhibited during run-time is called dynamic polymorphism
* In java Dynamic polymorphism is achieved through Method overriding and inheritance.
* Example:

|  |
| --- |
| Class A{  Static void calc(double x)  {  Sysout(“Square”+(x\*x))  }  } |
| Class B extends A{  Static void calc(double x)  {  double y=5;  Sysout(“area:”+(x\*y));  }  } |
| Public class Dynamicpoly {  Public static void main(String args[]){  A a=new B()  a.calc(5);  //output:25  B b=new B()  b.calc(6);  //output: 30  }  } |

1. Does java support multiple inheritances? Why?

Java doesn’t support multiple inheritances but it provides a way through which it can enact it.

Consider the scenario is C++:

|  |
| --- |
| Class A{  public void add(){  // some text  }  }  Class B{  public void add(){  // some text  }  }  Class C extends A,B{  public static void main(String arg[]){  C obj = new C();  obj.add(); // problem, compiler gets confused and cant  Decide to call Class A or B method.  } |
| This problem is called **Diamond problem.**  This problem in java is taken care with the use of interfaces  In Java similar problem would look like: |
| interface A{  add();  }  interface B{  add();  }  class C implements A,B{  add(){  // doesn’t matter which interface it belong to  }  } |

1. How is polymorphism achieved in java?

Inheritance, Overloading and Overriding are used to achieve Polymorphism in java

1. What is meant by static binding?

Static binding is a binding in which the class association is made during compile time. This is also called as early binding.

1. What is meant by Dynamic binding?

Dynamic binding is a binding in which the class association is not made until the object is created at execution time. It is also called as late binding.

1. Explain the different forms of Polymorphism?

From a practical programming viewpoint, polymorphism exists in three distinct forms in Java:

* Method overloading
* Method overriding through inheritance
* Method overriding through the Java interface