* **OOPs (Object Oriented Programing System):-**

There are four types in it----

1. Inheritance
2. Polymorphism
3. Abstraction
4. Encapsulation
5. **Inheritance:**

Inheritance it is oops principle where one class acquired property of other class.

The class from which property acquired is called super class or parent class.

The class in which property are delivered is called sub class or child class.

For inheritance we used extend keyword.

There are 5 types of inheritance in java but we can use only 3 because other two are not supported in java.

1. Single level inheritance
2. Multi-level inheritance
3. Hierarchical inheritance
4. **Single level inheritance:**

When a class extends another one class then we called it single level inheritance.

Egg. A=parent class, B=child class ( class B extends A)

1. **Multi-level inheritance:**

It is like chain of inheritance, means one sub class acquired the property of parent class then again another sub class acquired property of that sub class.

Egg. A=parent class, B=child class, C=child class ( class B extends A then class C extends B)

1. **Hierarchical inheritance:**

When two or more child classes extends single parent class it is known as Hierarchical inheritance.

Egg. A=parent class, B=child class, C=child class ( class B extends A then class C extends A)

1. **Polymorphism:**

Polymorphism is the concept where an object behaves differently in different situations.

And it also allows us to perform single action in different ways.

There are two types in polymorphism---

1. Compile time polymorphism
2. Run time polymorphism
3. **Compile time polymorphism: (Static Binding)**

Whenever an object is bound with its functionality at compile time, this is known as compile time polymorphism.

We can achieve it by method overloading.

\*\* Method overloading perform only in same class

Overloading means same method name but different arguments or parameters.

In method overloading we can change no. of parameters, change data type of the parameter and also we can change order of parameters.

1. **Run time polymorphism: (Dynamic Binding) (yaat aapn refral variable create krto)**

Whenever an object is bound with the functionality at run time, this is known as runt time polymorphism.

We can achieve it by method overriding.

\*\* Method overriding means same method name same arguments but class is different.

* Difference between Method overloading and Method overriding?

|  |  |
| --- | --- |
| **Method Overloading (Static Binding)** | **Method Overriding (Dynamic Binding)** |
| 1). Method overloading is a compile time polymorphism means it happens at compile time | 1). Method overriding is a run-time polymorphism means it happens at run-time. |
| 2). The binding of overloaded method is static | 2). The binding of overridden method is Dynamic |
| 3). It is occur within the class. | 3). While method overriding performed in two classes with inheritance relationship. |
| 4). In method overloading, return type can or cannot be same, but we must have change the parameter. | 4). In method overriding, return type must be same or covariant. |

1. **Abstraction:**

It means hiding the implementation and showing the service that has been provided.

Here we hide certain details and show only essential information to the user.

There are two ways to achieve abstraction----

1. By Abstract class
2. By Interface class

**Abstract class:-**

It contains abstract and concrete method.

Abstract method does not have body but the concrete method have body.

It is partially abstracted hence abstraction can be achieved from 0 to 100%.

It is a restricted class that cannot be used to create objects.

So we create reference of it.

Overriding can be done here.

We use extend keyword to inherited from a class.

**Interface:-**

Interface contains only abstract method so abstraction can be achieved 100%.

We create reference in it.

Here we use implements keyword for implementation.

Upto java 1.7 interface allows only abstract method but from 1.8 it allows default and static method.

* Difference between Abstract class and Interface?

|  |  |
| --- | --- |
| **Abstract class** | **Interface** |
| 1). Abstract class can have abstract and non-abstract methods. | 1). Interface can have only abstract methods. |
| 2). Abstract class does not support multiple inheritance. | 2). Interface supports multiple inheritance |
| 3). The abstract keyword is used to declare abstract class | 3). The interface keyword used to declare interface. |
| 4). Abstract class can be extended by using keyword “extends” | 4). Interface can be implemented by using keyword “implements” |
| 5). In abstract class we achieve 0 to 100% abstraction | 5). In interface we achieve 100% abstraction |
| 6). Abstract class can have class members like private, protected | 6). Members of java interface are public by default. |

1. **Encapsulation:**

It is a process of wrapping methods and variables in a single unit.

Encapsulation can be achieved by declaring all the variable in the class as private

And declaring public getter and setter method for viewing and modifying the data.

It is way to achieve data hiding in java. It is used for banking transactions as data hiding is done for security purpose.