Overloading & Overriding

CDAC PATNA

- **Overloading –**

Two methods are said to be overloaded if and only if , both methods having same name but different argument(either change type of argument / no. of argument/ order of argument)

- **Overriding -**

When we redefine the method of parent class in child class with same signature(i.e method name and argument) then it is said to be overriding**.**

- Parent class method which is overridden is called Overridden method
- Child class method which is overriding is called overriding method

Property	Overloading	Overriding
Method name	Must be same	Must be same
Argument type	Must be different	Must be same(including order)
Method signature	Must be different	Must be same
Return type	No restriction	Can be different
Also known as	Static polymorphism/compile time polymorphism	Dynamic polymorphism/ run time polymorphism
Inheritance	May or may not be required	Must be required
private, static, final method	Can be overload	Can not be overridden

CDAC PATNA

Interfaces

- Inside interface every method is always abstract whether we are declaring or not. Hence, interface is considered as 100% pure abstract class
- Syntax of declaring interface:-

```
interface interface_name
{
    data member;
     methods();
}
```

Where we can use extends or implements keyword ?

- A class can extend only one class at a time
- An interface can extend any no. of interface simultaneously
- A class can implement any no. of interface at a time
- A class can extend one class and can implement any no. of interfaces simultaneously.

Interface variable :-

Ex- interface I1

```
{
    int x=10;
}
```

public- to make this variable available to every implementation class

static- without creating object, implementation class has to access this variable

final- If one implementation class changes value then remaining implementation class will be affected .So, that interface variable is always final

Difference between interface and abstract class:-

Interface	Abstract class
If we don't know any thing about implementation then we should go for interface	If we know implementation but not completely(partial implementation) then we should go for abstract class .
Inside interface every method is always public& abstract , whether we are declaring or not . Hence, it is 100% abstract class	Every method present inside abstract class need not be public & abstract.
Every variable present inside interface is always public statis final	Every variable present inside abstract class need not to be public statis final
For interface variable compulsory we should perform initialization at the time of declaration (otherwise we get error).	For abstract class variables we are not required to perform initialization at the time of declaration

THANK YOU