

Definition of abstract class and abstract methods :

A class that does not provide complete implementation (partial implementation) is defined as an abstract class.

An abstract method is a common method which is used to provide easiness to the programmer because the programmer faces complexity to remember the method name.

An abstract method observation is very simple because every abstract method contains abstract keyword, abstract method does not contain any method body and at the end there must be a terminator i.e ; (semicolon)

In java, whenever action is common but implementations are different then we should use abstract method, Generally we declare abstract method in the super class and its implementation must be provided in the sub classes.

if a class contains at least one method as an abstract method then we should compulsory declare that class as an abstract class.

Once a class is declared as an abstract class we can't create an object for that class.

*All the abstract methods declared in the super class must be overridden in the sub classes otherwise the sub class will become as an abstract class hence object can't be created for the sub class as well.

In an abstract class we can write all abstract methods or all concrete methods or combination of both the method.

It is used to achieve partial abstraction that means by using abstract classes we can achieve partial abstraction(0-100%).

*An abstract class may or may not have abstract method but an abstract method must have abstract class.

Note :- We can't declare an abstract method as final, private and static (illegal combination of modifiers)

We can't declare an abstract class as a final.

//Programs :

```
abstract class Shape
{
    public abstract void draw();
}

class Square extends Shape
{
    @Override
    public void draw()
    {
        System.out.println("Drawing Square");
    }
}

class Circle extends Shape
{
    @Override
    public void draw()
    {
        System.out.println("Drawing Circle");
    }
}

public class AbstractDemo1
{
    public static void main(String[] args)
    {
        Shape shape = null;

        shape = new Square(); shape.draw();
        shape = new Circle(); shape.draw();
    }
}

abstract class Bike
{
    protected int speed = 80;

    public Bike()
    {
        System.out.println("No Argument Constructor");
    }

    public void getBikeDetails()
    {
        System.out.println("Bike has two wheels..");
    }

    public abstract void run();
}

class Honda extends Bike
{
    @Override
    public void run()
    {
        System.out.println("Honda Bike is running");
    }
}

public class IQ
{
    public static void main(String[] args)
    {
        Bike bike = new Honda();
        System.out.println("Bike Speed is :"+bike.speed);
        bike.getBikeDetails();
        bike.run();
    }
}
```

Note : abstract class constructor will be executed with the help of sub class object using super()

IQ :

What is the advantage of writing instance variable(Object properties) and constructor inside an abstract class, If we can't create an object for abstract class ?

Yes, We can't create an object for abstract class but still we can write instance variable and constructor that means abstract class also contains properties and behavior but these properties will initialize through **sub class object using super()** so we can write object properties as well as constructor.

An abstract class may contain only concrete methods [Abstraction level is : 0]

```
abstract class Hello
{
    public void m1()
    {
    }
}

public class AbstractDemo2
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}

abstract class Test
{
    public abstract void m1();

    public void m2()
    {
    }
}

public class AbstractDemo3
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}
```