**Method return type**

**Primitive type :**

byte

short

int

long

float

double

char

boolean

int meth() {

return 5; // return statement

}

Example :

**class Animal1 {**

**int age;**

**int display(int n) {**

**age=n;**

**return age;**

**}**

**}**

**public class Demo1 {**

**public static void main(String[] args) {**

**Animal1 aa=new Animal1();**

**int value=aa.display(20);**

**System.*out*.println("the value is : "+value);**

**}**

**}**

**Constructor**

**Constructor is used to initialize the object.**

**Constructor name and class name should be same.**

**Constructor will call at the time of creating an object.**

**Constructor don’t have any return type not even void.**

**Constructor mostly used to initialize the instance variable.**

**There are two types of constructor**

* **Default constructor**
* **Parameterized constructor**

**Example :**

**class Car {**

**int model;**

**Car() { //default constructor**

**model=10;**

**}**

**Car(int a,int b) { // parameterized constructor**

**model=a+b;**

**}**

**void display() {**

**System.*out*.println("value of model is : "+model);**

**}**

**}**

**public class ConstDemo {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**Car cc=new Car();**

**Car cc1=new Car(5,20);**

**cc.display();**

**cc1.display();**

**}**

**}**

**Method Overloading :**

With in the class we can have more than one method with same name and with different

Parameter is called method overloading.

Based upon the argument that we passed to the parameter it decide which method to call.

It comes with compile time polymorpshim

**Example :**

**class Machine {**

**int tyre(int a) {**

**return a;**

**}**

**int tyre(int a,int b) {**

**return a+b;**

**}**

**int tyre(int a,int b,int c) {**

**return a+b+c;**

**}**

**}**

**public class MethodOverloading {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**Machine mm=new Machine();**

**System.*out*.println(mm.tyre(10));**

**System.*out*.println(mm.tyre(10,20));**

**System.*out*.println(mm.tyre(10,20,30));**

**}**

**}**