The object-oriented programming is basically a computer programming design methodology or technique where store data and their functionality both together. OOP is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures or methods (often known as behaviors).

Everything in Java is associated with classes and objects, along with its attributes and methods. For example: in real life, a car is an object. The car has attributes, such as weight and color, and methods, such as drive and brake.

The Object –oriented programming has some basic structure or component there are following.

1) Class
2) Data member variables
3) Encapsulation.
4) Access modifier
5) Constructors
6) Setter and getter method
7) Object
8) Abstraction
9) Inheritance
10) Polymorphism

Class

Class is a user-defined datatype that has its own data members and member functions whereas an object is an instance of class by which we can access the data members and member functions of the class. A class is used to organize information or data so a programmer can reuse the elements in multiple instances.

In object-oriented programming, a class is a blueprint for creating objects (a particular data structure), providing initial values for state (member variables or attributes), and implementations of behavior (member functions or methods). a class is a template definition of the methods and variable. The user-defined objects are created using the class keyword.

Data Member Variable

an object-oriented programming, a member variable (sometimes called a member field, attribute and properties) is a variable that is associated with a specific object, and accessible for all its methods (member functions).

Example-1

```
public class Human{
    String Name;
    String FatherName;
    float Height;
    float Waight;
}
```

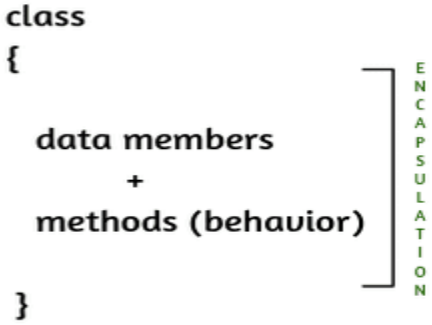
Data member variable's

Example-2

```
public class Car{
    String CompanyName;
    int   Model;
    int EnginNo;
    int RegNo;
}
```

Data member variable's

Encapsulation is a mechanism that allows us to bind data and functions of a class into an entity. It protects data and functions from outside interference and misuse. Therefore, it also provides security and This also helps to achieve data hiding. A class is the best example of encapsulation.



Access modifier

In Java, access modifiers are used to set the accessibility (visibility) of classes, interfaces, variables, methods, constructors, data members, and the setter/getter methods. There are four type of access modifiers in java language.

Modifier	Description
Default	declarations are visible only within the package (belong to all class of package)
Private	declarations are visible within the class only. It cannot be accessed from outside the class.
Protected	The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.
Public	The access level of a **public** modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

The meaning of Encapsulation, is to make sure that "sensitive" data is hidden from users. To achieve this, you must: **declare class variables/attributes as private example this is**

Example-1

```
public class Human{
    private String Name;
    private String FatherName;
    private float Height;
    private  float Waight;
}
```

Data member variable's with encapsulation

Example-2

```
public class Car{
    public String CompanyName;
    public int   Model;
    public int EnginNo;
    public  int RegNo;
}
```

Data member variable's without encapsulation

Access modifier it could be classes, data member variables, methods and constructors.

Constructor

A constructor in Java is a special method that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes:

There are three rules defined for the constructor.

1. Constructor name must be the same as its class name
2. A Constructor must have no explicit return type
3. A Java constructor cannot be abstract, static, final, and synchronized

It is called constructor because it constructs the values at the time of object creation. It is not necessary to write a constructor for a class. It is because java compiler automatically creates a default constructor with default value depend of data type as where if your class doesn't have any.

How to create object of class with constructor? According to define above class of Human and Car. Example is

```
public class TestClass
{

    public static void main(String[] args)
    {                    objectName        constructor
        Human  HumObj1 = new Human();
        Car       CarObj1 = new Car();
    }
}
```

```
public class Human{
    private String Name;
    private String FatherName;
    private float Height;
    private float Weight;

     public Human()    // this is constructor for initialize
    {
         Name=Null;
        FatherName=null;
        Height = 0.0f;
        Weight = 0.0f;
    }
}
```

Types of Constructor

1) Default constructor
2) Non-arg (no parameterize or no argument) Constructor
3) Parameterize constructor

Default Constructor

If we not write constructor, then java compiler automatically creates a default constructor with default value depend of data type as where.

Non-arg (non-parameterize or no-argument) Constructor

The constructor is called when an object of a class is created. It can be used to set initial values for object attributes without any argument or parameters. Example this is

```
public class Human{
      private String Name;
      private String FatherName;
      private float Height;
      private  float Weight;
         public Human()    // this is constructor for initialize
      {
            Name=”nonthing”;
           FatherName=”nothing”;
           Height = 0.0f;
           Weight = 0.0f;
      }
}
```

```
public class Human{
      private String Name;
      private String FatherName;
      private float Height;
      private  float Weight;
        public Human()    // this is constructor for initialize
      {
            Name=”Muhammad Asif”;
           FatherName=”Muhammad Imran”;
           Height 5.6f;
           Weight = 65.34f;
      }
}
```

Parameterize Constructor

The constructor is called when an object of a class is created. It can be used to set initial values for object attributes with any argument or parameters. Example this is

```
public class Human{
      private String Name;
      private String FatherName;
      private float Height;
      private  float Weight;

       public Human(String n,String Fn,float h, float w)     // this is constructor for initialize with parameterize
      {
            Name=n;
           FatherName=Fn;
           Height = h;
           Weight = w;
      }
}
```

Getter and Setter Methods

In Java, getter and setter are two conventional methods that are used for retrieving and updating value of a variable. Getter and Setter are methods used to protect your data and make your code more secure and it is to access and update the value of a **private** variable.

Getter Methods

The Getter returns the value (assessors), it returns the value of data type **int**, String, double, **float**, etc. For the convenience of the program, getter starts with the word "get" followed by the variable name.

Setter Methods

While Setter sets or updates the value (matadors). It sets the value for any variable which is used in the programs of a class. and starts with the word "set" followed by the variable name. Getter and Setter make the programmer convenient in setting and getting the value for a particular data type. In both getter and setter, the first letter of the variable should be capital.

```
public class Human{
      private String Name;
      private String FatherName;
      private float Height;
      private  float Weight;

       public Human(String n,String Fn,float h, float w)     // this is constructor for initialize with parameterize
      {
            Name=n;
           FatherName=Fn;
           Height = h;
           Weight = w;
      }
  public String getName()   // getter method
 {
           return Name;
 }
  public void setName(String newName)   // setter method
 {
           this.Name = newName;
 }

}
```

Example explained
The get method returns the value of the variable name.

The set method takes a parameter (newName) and assigns it to the name variable. The ***this*** keyword is used to refer to the current object. However, as the name variable is declared as **private**, we cannot access it from outside this class:

Object
A Java object is a member (also called an instance) of a Java class. Each object has an identity, a behavior and a state. The state of an object is stored in fields (variables), while methods (functions) display the object's behavior. Objects are created at runtime from templates, which are also known as classes.

```
public class Human{
    private String Name;
    private String FatherName;
    private float Height;
    private  float Weight;
        public Human()    // this is constructor for initialize
    {
        Name="nonthing";
        FatherName="nothing";
        Height = 0.0f;
        Weight = 0.0f;
    }

    public Human(String n,String Fn,float h, float w)    // this is constructor for initialize with parameterize
    {
        Name=n;
        FatherName=Fn;
        Height = h;
        Weight = w;
    }
```

```
public class Car{
    private String CompanyName;
    private String CarName;
    private int chassisNo;
    private  int EnginNo;
        public Car{ ()    // this is constructor for initialize
    {
        CompanyName="nonthing";
        CarName="nothing";
        chassisNo = 0;
        EnginNo = 0;
    }

    public Car (String n,String cn,int ch, int eng)    // this is constructor for initialize with parameterize
    {
        CompanyName=n;
        CarName=cn;
        ChassisNo = h;
        EnginNo = w;
    }
```

In Java, an object is created using the keyword "**new**". Like this is
Human HumObj1 = new Human();
Or
Human HumObj2 = new Human("Nadeem","Farhan",5.6f,60.45f)
or
Car CarObj1 = new Car();
or
Car CarObj2 = new Car("Suzuki","Alto",2012,1872653,89363)

Java objects are very similar to the objects we can observe in the real world. A Human, a lighter, a pen, or a car are all objects. They are characterized by two features:

1) State
2) Behavior

For example, a Human state includes its color, height, gender, and age, while its behavior is sleeping, walking and Language for speaking.

Class	Object
Class is the blueprint of an object. It is used to declare and create objects.	Object is an instance of class.
No memory is allocated when a class is declared.	Memory is allocated as soon as an object is created.
A class is a group of similar objects.	Object is a real-world entity such as book, car, etc.
Class is a logical entity.	Object is a physical entity.
Class can only be declared once.	Object can be created many times as per requirement.
Example of class can be car.	Objects of the class car can be BMW, Mercedes, Suzuki, etc.

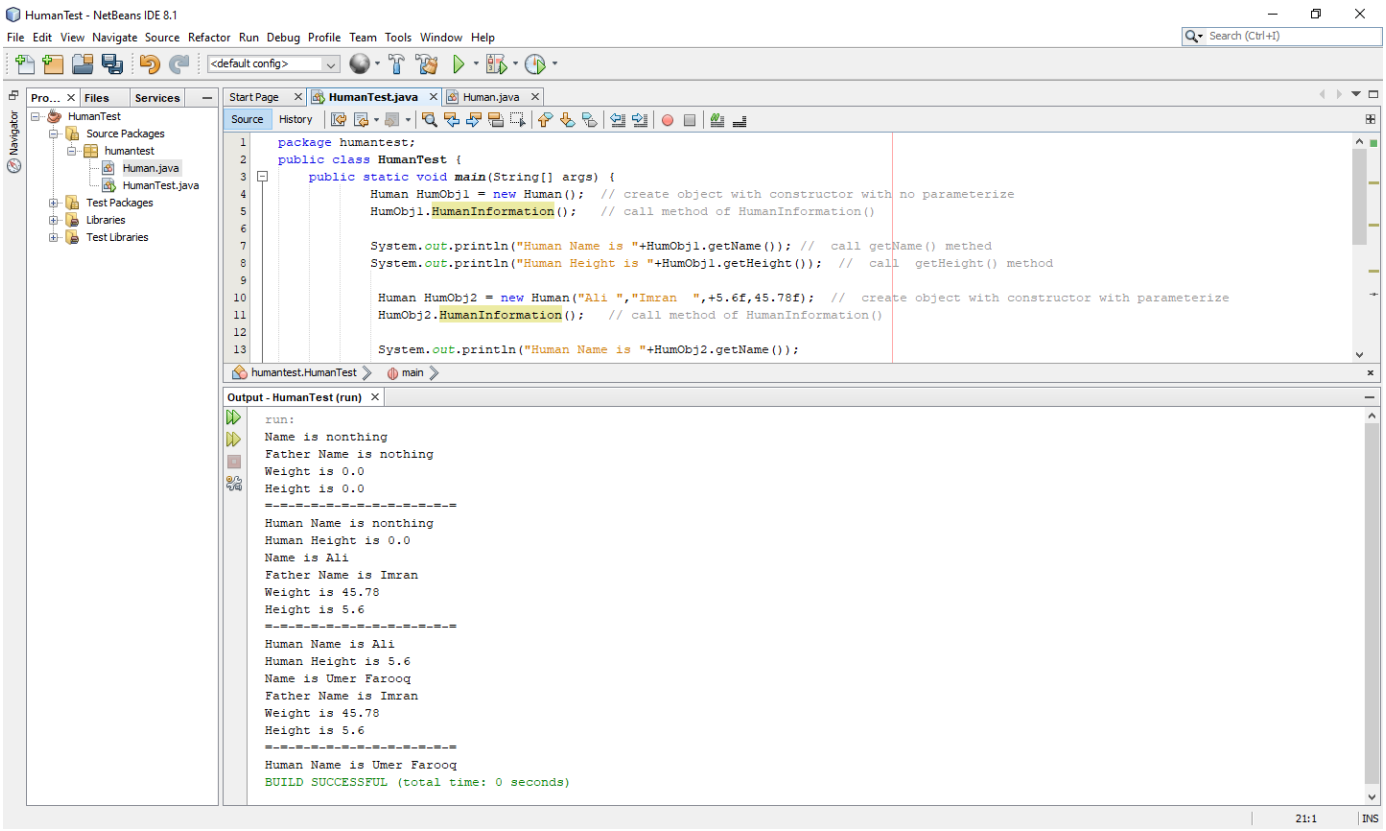
Difference between constructor and method in Java

There are many differences between constructors and methods. They are given below.

Java Constructor	Java Method
A constructor is used to initialize the state of an object.	A method is used to expose the behavior of an object.
A constructor must not have a return type.	A method must have a return type.
The constructor is invoked implicitly.	The method is invoked explicitly.
The Java compiler provides a default constructor if you don't have any constructor in a class.	The method is not provided by the compiler in any case.
The constructor name must be same as the class name.	The method name may or may not be same as the class name.

```
public class Human{
    private String Name;
    private String FatherName;
    private float Height;
    private  float Weight;
     public Human()    // this is constructor for initialize
    {
         Name="nonthing";
        FatherName="nothing";
        Height = 0.0f;
        Weight = 0.0f;
    }

    public Human(String n,String Fn,float h, float w)     //
this is constructor for initialize with parameterize
    {
         Name=n;
        FatherName=Fn;
        Height = h;
        Weight = w;
    }
    public String getName()   // getter method
    {
            return Name;
    }
    public void setName(String newName)   // setter method
    {
            this.Name = newName;
    }
    public void HumanInformation()
    {
     System.out.println("Name is "+Name);
     System.out.println("Father Name is "+FatherName);
     System.out.println("Weight is "+Weight);
     System.out.println("Name is "+Height);
    System.out.println("**************************");
     }

    public float getHeight()   // getter method
    {
            return Height;
    }

}
```

```
public class HumanTest
{    public static void main(String [] args)
    {
        Human HumObj1 = new Human();  // create object with constructor with no parameterize
        HumObj1.HumanInformation();   // call method of HumanInformation()
        System.out.println("Human Name is "+HumObj1.getName()); //  call getName() methed
       System.out.println("Human Name is "+HumObj1.getHeight());  //  call  getHeight() method

         Human HumObj2 = new Human("Ali ","Imran  ",+5.6f,45.78f);  //  create object with constructor with parameterize
         HumObj2.HumanInformation();   // call method of HumanInformation()
        System.out.println("Human Name is "+HumObj2.getName());
       System.out.println("Human Name is "+HumObj2.getHeight());
   // now we change Name by setName() method
      HumObj2.setName("Umer Farooq");
      HumObj2.HumanInformation();   // call method of HumanInformation()
       System.out.println("Human Name is "+HumObj2.getName());
   }}
```


HumanTest - NetBeans IDE 8.1
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Pro... Files Services
HumanTest
Source Packages
humantest
Human.java
HumanTest.java
Test Packages
Libraries
Test Libraries
Start Page
HumanTest.java
Human.java
Source History
1 package humantest;
2 public class HumanTest {
3 public static void main(String[] args) {
4 Human HumObj1 = new Human(); // create object with constructor with no parameterize
5 HumObj1.HumanInformation(); // call method of HumanInformation()
6
7 System.out.println("Human Name is "+HumObj1.getName()); // call getName() methed
8 System.out.println("Human Height is "+HumObj1.getHeight()); // call getHeight() method
9
10 Human HumObj2 = new Human("Ali ","Imran ",+5.6f,45.78f); // create object with constructor with parameterize
11 HumObj2.HumanInformation(); // call method of HumanInformation()
12
13 System.out.println("Human Name is "+HumObj2.getName());
humantest.HumanTest
main
Output - HumanTest (run)
run:
Name is nonthing
Father Name is nothing
Weight is 0.0
Height is 0.0
=-=-=-=-=-=-=-=-=-=-=-=
Human Name is nonthing
Human Height is 0.0
Name is Ali
Father Name is Imran
Weight is 45.78
Height is 5.6
=-=-=-=-=-=-=-=-=-=-=-=
Human Name is Ali
Human Height is 5.6
Name is Umer Farooq
Father Name is Imran
Weight is 45.78
Height is 5.6
=-=-=-=-=-=-=-=-=-=-=-=
Human Name is Umer Farooq
BUILD SUCCESSFUL (total time: 0 seconds)
21:1
INS

Exercise

Theory Questions

1. Define Object Oriented Programming.
2. What is Access modifier and their types?
3. What do you mean by Encapsulation?
4. How you can differentiate between the object and class?
5. What is constructor and how many types of constructor.
6. What is difference between the method and constructor.

Practical Questions

1. Create Computer class with parameterize constructors.
2. Create mobile class with constructor and display mobile information using method.
3. Make a student class with ***RollNo,Name, FatherName, and Department*** with constructor and create three different object.

Objective MCQ's

1) The object-oriented programming is basically a computer programming design_____
   a) methodology
   b) language
   c) module
   d) way

2) The OOP where store _____ and their _____ both together.
   a) class and object
   b) class and constructor.
   c) Data and Functionality.
   d) Field and attribute

3) In object-oriented programming, a _____ is a blueprint for a particular data structure.
   a) object
   b) class
   c) variable
   d) Function

4) Encapsulation is a mechanism that allows us to _____
   a) binding
   b) hiding
   c) binding and hiding
   d) nothing all

5) Constructor name must be the _____ as its class name
   a) same
   b) different
   c) constructor
   d) nothing all

6) A Java _____ is an instance of a Java class
   a) constructor
   b) class
   c) method
   d) object

7) In Java, an object is created using the keyword "____"
   a) if
   b) create
   c) new
   d) class

8) The _____ method sets or updates the value (matadors) data member variable.
   a) getter
   b) update
   c) setter
   d) nothing all

9) data member variables It cannot be accessed from outside the class.
   a) private
   b) public
   c) default
   d) protected

10) data member variables Declarations are visible only within the package (belong to all class of package)
   a) public
   b) private
   c) protected
   d) default

11) A method must have a ____ type.
   a) void
   b) int
   c) float
   d) return any data

12) It can be used to set initial values for object attributes with any argument or parameters.
   a) Constructor
   b) Default constructor
   c) Parameterize constructor
   d) Nothing all