OOPs (Object-Oriented Programming System)

- **Object** means a real-world entity such as a pen, chair, table, computer, watch, etc.
- **Object-oriented programming is a methodology or paradigm used** to design a program using classes and objects.
- The main aim of object-oriented programming is to implement real-world entities, for example, objects, classes, abstraction, inheritance, polymorphism, etc.
- It simplifies the software development and maintenance.

Note:

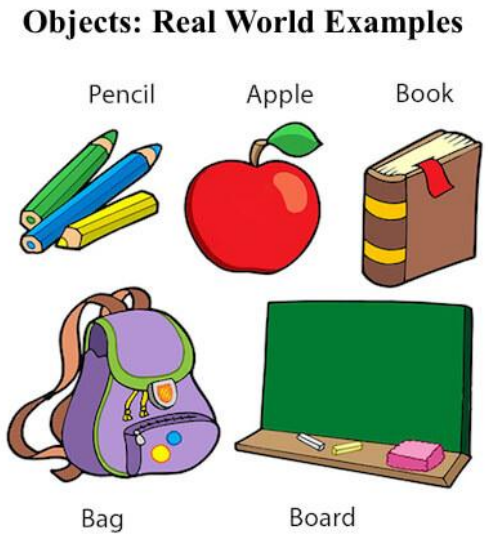
- **Simula** is considered the first object-oriented programming language. The programming paradigm where everything is represented as an object is known as a truly object-oriented programming language.
- **Smalltalk** is considered the first truly object-oriented programming language.
- The popular object-oriented languages are Java, C#, PHP, Python, C++, etc.

Object:

An object is an instance of a class. A class is a template or blueprint from which objects are created. So, an object is the instance(result) of a class.

Object Definitions:

- An object is *a real-world entity*.
- An object is *a runtime entity*.
- The object is *an instance of a class*.
- Object is the basic unit of object-oriented programming.
- An object is a variable of type class.
- It stores data and class methods.
- Example: Alto, Zen Estilo, Swift etc are the objects of class Car.



Creation of Objects:

- Once the class is created, one or more objects can be created from the class as objects are instances of the class.
- Objects are declared as:

  Syntax: class-name object-name= new class-name();

  Example: Student S1=new Student();

The **new** keyword is used to allocate memory at runtime. All objects get memory in the Heap memory area.

Class:

- Class is a collection of data members and member functions.
  - Member data specifies the type of data to be stored.
  - Member function acts as a mediator between user and data. They manipulate data.

- Class is a user-defined data type.
- The class implements encapsulation and abstraction.
- A class is a group of objects which have common properties.
- It is a template or blueprint from which objects are created. It is a logical entity. It can't be physical.

A class in Java can contain:

- Fields-data members
- Methods/functions
- Constructors/special member function
- Blocks
- Nested class and interface

Class Declaration/ Class Prototyping:

- A class is specified with a name after the keyword class.
- The body of the class begins with the curly bracket '{' and is followed by the code containing the data members and functions. Then the class is closed with curly bracket '}' and concluded with a semi-colon (;)- Optional.
- There are different access specifiers for defining the data and functions present inside a class.
- In Java, the class can be declared as follows:
- Syntax:

```
class Classname
{
list of data members;
constructors
list of member function;
etc..
} ; (optional)
```

Methods or Functions

- A Method/function is a group of statements that performs any coherent task.
- A function contains a set of statements that are bundled within a set of curly brackets and is identified by a name called function name.
- It can be classified into two categories:
  - Predefined or In-Built functions
  - User-defined function.
- The following three things have to be done to use a function
  - **Function Prototyping**
    Syntax:
    Return type function name (list of ordered formal arguments);
    void main(String args[])
    int addition(int x,int y) // x and y are formal arguments
  - **Function Definition/Body**
    Syntax:
    Return type function name (list of formal arguments)
    {
    Set of statements;//function body
    }

  - **Function Calling**
    Syntax:
    Object name. function name (list of ordered actual arguments);
    s.addition(3,5);//3 and 5 are actual arguments

Declaration styles of method:

1. int addition(int x, int y) –takes something, returns something
2. void addition(int x,int y)- takes something, returns nothing
3. int addition( )- takes nothing returns something
4. void addition() - takes noting, returns nothing

Access modifiers:

There are two types of modifiers in Java: access modifiers and non-access modifiers.
The access modifiers in Java specify the accessibility or scope of a field, method, constructor, or class. We can change the access level of fields, constructors, methods, and classes by applying the access modifier to it.
There are four types of access modifiers/specifiers in the Java programming language:

- Private (Access within class)
- protected (Access within class and derived class) A and B child class
- Package (Access within the directory/folder)- It is by default in Java
- Public (Access from anywhere)

There are many non-access modifiers, such as static, abstract, synchronized, native, volatile, transient, etc. **TODO**

Java Program to illustrate how to define a class and fields(data members).

```java
class Demo
 {
  int roll;
  String name;
  double marks;
public static void main(String args[])
{
Demo d1 =new Demo();
System.out.println(d1.roll);
System.out.println(d1.name);
System.out.println(d1.marks);

   }
 }
```

C:\Windows\System32\cmd.exe

```
F:\Java Code>javac Demo.java

F:\Java Code>java Demo
0
null
0.0
```

Note: In real-time development, we create classes and use them from another class. It is a better approach than the previous one. Let's see a simple example where we have a main() method in another class.

main outside the class

```java
class Demo1
{
 int roll;
 String name;
 double marks;
 }

 class TestDemo1
 {
   public static void main(String args[])
   {
   Demo1 d1 =new Demo1();
   System.out.println(d1.roll);
   System.out.println(d1.name);
   System.out.println(d1.marks);
  }
}
```

If we save this program as Demo1.java,it will show a compile-time error.

C:\Windows\System32\cmd.exe

```
F:\Java Code>javac Demo1.java

F:\Java Code>java Demo1
Error: Main method not found in class Demo1, please define the main method as:
   public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
```

We can have multiple classes in different Java files or a single Java file. If you define multiple classes in a single Java source file, it is a good idea to save the file name with the class name which has the main() method.

C:\Windows\System32\cmd.exe

```
F:\Java Code>javac TestDemo1.java

F:\Java Code>java TestDemo1
0
null
0.0
```

There are three ways to initialise the object in Java.

- By reference variable
- By method
- By constructor

Initialize object by reference variable

Initialising an object means storing data in the object. Let's see a simple example where we are going to initialise the object through a reference variable.

```java
class Demo2
 {
  int roll;
  String name;
  double marks;
  }

  class TestDemo2
  {
    public static void main(String args[])
    {
    Demo1 d1 =new Demo1();
    //roll=20; will show compile time error
    d1.roll=20;
    d1.name="Saurabh";
    d1.marks=97.6754;

    //System.out.println(roll); will show error
    System.out.println(d1.roll);
    System.out.println(d1.name);
    System.out.println(d1.marks);
   }
 }
```

Output:

C:\Windows\System32\cmd.exe

```
F:\Java Code>javac TestDemo2.java

F:\Java Code>java TestDemo2
20
Saurabh
97.6754
```

Initialization through method

In this example, we are creating the two objects of the Student class and initializing the value of these objects by invoking the method.

```
class Student
 {
  int roll;
  String name;
  double marks;
  public void setData(int r, String s, double m) // mutator
   {
     roll=r;
     name=s;
     marks=m;
   }
   public void getData() // accessor
   {
    System.out.println("Roll number of student:"+roll);
    System.out.println("Name of student:"+name);
    System.out.println("Marks of student:"+marks);
   }
 }
class TestStudent
 {
  public static void main(String args[])
  {
   Student S1=new Student();//object create
   S1.setData(49,"Rahul",840.67);
   S1.getData();
   System.out.println();//Inserting new line
   Student S2=new Student();
   S2.setData(54,"Shruti",785.89);
   S2.getData();
  }
 }
```

C:\Windows\System32\cmd.exe

```
F:\Java Code>javac TestStudent.java

F:\Java Code>java TestStudent
Roll number of student:49
Name of student:Rahul
Marks of student:840.67

Roll number of student:54
Name of student:Shruti
Marks of student:785.89
```

Initialisation through a constructor: We will discuss this later.

Class Work

Write a program using a function that prints the area of a rectangle(Take input from the user through the Scanner class).

```
import java.util.Scanner;
 class Rectangle
 {
  int l,b,area;

  public void insertData()
  {
     Scanner s=new Scanner(System.in);
     System.out.println("Enter length");
     l=s.nextInt();
     System.out.println("Enter Width");
     b=s.nextInt();
  }
  public void printArea()
  {
  area=l*b;
  System.out.println("Area is :"+area);
  }
  }

  class TestRectangle
  {
  public static void main(String args[])
   {
   Rectangle r=new Rectangle();
   r.insertData();
   r.printArea();
   }
 }
```

```
C:\Windows\System32\cmd.exe

F:\Java Code>javac TestRectangle.java

F:\Java Code>java TestRectangle
Enter length
34
Enter Width
45
Area is :1530
```

Java Program will demonstrate how to use a banking system, where we deposit and withdraw amounts from our account. Display the transaction details of the customer account.

```
class Account1{
int acc_no;
String name;
double amount;
//Method to initialize the object
void insert(int a,String n,double amt){
acc_no=a;
name=n;
amount=amt;
}
//deposit method
void deposit(double amt){
amount=amount+amt;
System.out.println(amt+" deposited");
}
//withdraw method
void withdraw(double amt){
if(amount<amt){
System.out.println("Insufficient Balance");
}else{
amount=amount-amt;
System.out.println(amt+" withdrawn");
```

```
}
}
//method to check the balance of the account
void checkBalance()
{
System.out.println("Balance is: "+amount);}
//method to display the values of an object
void display()
{System.out.println("Account No is:"+ acc_no);
System.out.println("Customer Name is:"+name);
}
}
//Creating a test class to deposit and withdraw amount

class TestAccount1{
public static void main(String[] args){
Account1 a1=new Account1();
a1.insert(123456,"Saurabh",50000);
a1.display();
a1.checkBalance();
a1.deposit(5000);
a1.checkBalance();
a1.withdraw(10000);
a1.checkBalance();
}}
```

C:\Windows\System32\cmd.exe

```
F:\Java Code 2020>javac TestAccount1.java

F:\Java Code 2020>java TestAccount1
Account No is:123456
Customer Name is:Saurabh
Balance is: 50000.0
5000.0 deposited
Balance is: 55000.0
10000.0 withdrawn
Balance is: 45000.0
```

Using Scanner class:

```
import java.util.*;
class Account{
long a;
String n;
double amt;
Scanner s=new Scanner(System.in);
//Method to initialize object
void insert(){
System.out.println("Enter acc_no");
long a=s.nextLong();
System.out.println("Enter name");
String n =s.next();
System.out.println("Enter amount");
double amt=s.nextDouble();
this.a=a;
this.n=n;
this.amt=amt;
}
//deposit method
void deposit(){
System.out.println("Enter amount to be deposited");
double damount=s.nextDouble();
amt=damount+amt;
System.out.println(damount+" deposited");
}
//withdraw method
void withdraw(){
System.out.println("Enter amount to be withdrawn");
double wamount=s.nextDouble();
if(amt<wamount){
System.out.println("Insufficient Balance");
}else{
amt=amt-wamount;
System.out.println(wamount+" withdrawn");
}
}
//method to check the balance of the account
void checkBalance(){System.out.println("Balance is: "+amt);}
//method to display the values of an object
void display()
{System.out.println("Account No is:"+ a);
System.out.println("Customer Name is:"+n);
}
}
```

//Creating a test class to deposit and withdraw amount

```
class TestAccount{
public static void main(String[] args){
Account a1=new Account();
a1.insert();
a1.display();
a1.checkBalance();
a1.deposit();
a1.checkBalance();
a1.withdraw();
a1.checkBalance();
}}
```

C:\Windows\System32\cmd.exe

```
F:\Java Code 2020>java TestAccount
Enter acc_no
12345
Enter name
Saurabh
Enter amount
50000
Account No is:12345
Customer Name is:Saurabh
Balance is: 50000.0
Enter amount to be deposited
5000
5000.0 deposited
Balance is: 55000.0
Enter amount to be withdrawn
10000
10000.0 withdrawn
Balance is: 45000.0
```