# Ch-1 - Introduction To Java

## * Java Introduction :

Java is a Programming language.

Java is a high level and object-oriented programming language.

-> Features :

1. Simple : Java is very easy to learn and simple syntax and easy understand.
2. PlatForm Independent : Java is platform independent because it is different from other Language.
3. Secured : Java can develop virus-free systems.
4. Robust : Java is case-sencetive Language.
5. Portable : Java does not require any implemenatation.

6 High-Performance : Java is faster than other programming languages.

7 Dynamic : Java supports dynamic loading of classes.

8 Multi-threaded : Java programs deal with many tasks at a once using multi-threaded.

-> Applications :

1 Desktop GUI application:

Java is provides APIs like Swing, AWT and JavaFX provide way to develop GUI application.

2 Mobile Application:

Java is a cross-platform framework that is used to build applications.

3 Web Application:

Java is perfect for developing

web application because its abillity to interact with a large number of system.

4 Business Application:

Java helps us to develop robust application for business requirements.

5 Cloud Application:

Java provides a solution for IT infrastructure at an affordable cost.

6 Scientific Application:

Java has enhaned security features which is the best option for scientific application

* Explain JDK, JRE and JVM.

=> JVM : Java Virtual Machine.

JVM is provides a specification that provides runtime environment.

JVM convert the other languages program into the java bytecode.

JVM can performs Loads code, verifires code and executes code.

=> JRE : Java Runtime Environment

Java Runtime environment is a set of software tools which are used to develop Java application.

=> JDK : Java Development kit

Java Development kit is a software development environment which is develop Java application and Java applet.

The JDK contains a private Java Virtual Machine.

* Java Variable:

Primitive :

```
int a = 10;
boolean a = True;
double a = 10.000;
float f = 10f;
char abc = "a";
```

Non-primitive:

```
String a = "K";
Array
```

* Java Control Statements:

1 Decision Making.

(i) If statements

- Simple if
- if-else
- if-else-if ladder
- Nested if

(ii) Switch Statements

2 Loop Statements.

- For loop

- While loop
- Do while loop
- For - each loop

3 Jump statement.

(i) Break

(ii) Continue

* Explain Java Method and Method overloading, ~~Overridding~~.

=> Method:

Method is a collection of instructions that performs a specific task.

A method is a block of code that perform a specific task in a program.

Using method we do not require to write code again.

-> There are Two types of method:

1 User defined method:

User create its own method for a user requirement.

2 Standard Library method:

There are built in method in java that are available for user to use.

-> Syntax:

```
Return type  Method Name ( Parameter or Null )
{
    // body
}
```

-> Method Calling Syntax:

```
Method name ( Parameter or Null );
```

Ex.

```
class java
{

  int sum()
  {
    int a = 5;
    int b = 6;

    int sum = a+b;

  }

public static void main(String
      args[])
{
  int result;
    java obj1 = new java();

int result = obj1.sum();

System.out.println("Sum is: "
      + result);
  }
}
```

=> Method with Return Type:

Return keyword is used to complete execution of a method.

Return keyword is used to exit from the method.

~~Rutu~~
Return Keyword is declare in a method body.

Syntax:

```
Return Method ( )
  type   Name
{
    // body

  return (Value) ;

}
```

Ex.

```
Class java
{
  int sum ( ) int a , int b )
  {
    return a + b ;
  }
```

```
public static void main(String
        args[])
{
    int x = 5;
    int y = 10;

    sum(x, y);

    System.out.println("Sum is ");
}

}
```

=> Method with Parameters:

Parameters act as variable inside the method.

Parameters are decalre after the method name and inside the parentheses.

Syntax:

```
return Method (Parameter
  type   type
{
    // body

}
```

Ex. Class java

```
{
    int sum (int a, int b)
    {
        return a+b;
    }

    public static void main
            (String args[])
    {
        int x = 5;
        int y = 10;

        sum(x, y);

        System.out.println("sum is")
    }
}
```

=> Method Overloading:

If method has same name and different parameter it is called method overloading.

In method overloading, method have same name.

Method Overloading can be done using this two method.

1) Changing the number of arguments.

2) Changing the data type.

Using method overloading, we can use same name method with different argument.

Method overloading is also known as Compile-time Polymorphism or Early binding.

Method overloading increases the readability of the program.

Ex.

```
class java
{
  int sum(int a, int b)
  {
    return a+b;
  }
  int sum(int a, int b, int c)
  {
    return a+b+c;
  }
```

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

    sum(5, 6);
    sum(5, 6, 7);
}
}
```

* Explain Array in Java.

=> An array is a collection of similar type of element which has contiguous memory.

Array is a linear data structure in which we store similar type of data

In array, we can stored only a fixed set of element.

Array is index-based data structure, First element of the array is stored at the 0th index.

Using array, we can get any data located at an index position.

Syntax:

data type Array[] = new data type [size];

There are two types of Array:

1) Single Dimensional Array

2) Multi Dimensional Array

1 Single Dimensional Array.

One dimensional array represents one row or one column of array element.

Syntax:

data type array[] = new data type [size];

Ex. class java

```
{
  public static void main
        (String args[])
  {
    int a[] = {1,2,3,4,5};
    For(int i =0; i<a.length; i++)
    {
      System.out.println(a[i]+" ");
    }
  }
}
```

2 Multidimensional Array:

Multidimensional array can be represent multiple row and column of array element.

Syntax:

```
data [][] array = new data
type      name      type
                    [size]
                    [size];
```

Ex. class java
{
public static void main
(String args[])

{
int [][] a = {{1,2},{3,4}};

For(i=0;i<2;i++)
{
For(j=0;j<2;j++)
{
System.out.println
(a[i][j]);
}
}
}
}