* Introduction to OOP :-

-> OOP is invented to remove the flaws in procedural language.
-> Object Oriented language gives importance to data i.e we can access data at an specific manner.
-> But in procedural we can't restrict the usage of data all the time.
-> the main part in OOP is "Object". An object is a blue print of a class.
-> we create an object for particular class to access the methods & variables in that class.
-> the data of an object can be accessed only by the function associated with that object.

program :-

```
class ObjectDemo
{
    public int x = 10;
    public float y = 0.5f;
    public static void main(String args[])
    {
        ObjectDemo obj = new ObjectDemo();
        obj.x;
        System.out.println(Obj.y);
    }
}
```

o/p :- 0.5

* principles of OOP :-

1) Objects
2) Classes
3) Data Abstraction & Encapsulation
4) Inheritance.

5. polymorphism
6. Dynamic binding
7. Message passing.

Object :-

→ Objects are the basic run time entities in an oop.

→ Objects contains data & code to manipulate that data.

classes :-

→ The entire set of data & code of an object can be made by a datatype with the help of class.

→ Objects are nothing but variables of the type class.

→ If we define a class then we create any no. of objects which belongs to that class.

Data Abstraction :-

It means it only gives the declaration but not definitions of the particular class.

Encapsulation :-

It is a programming mechanism that binds together code and that data it manipulates.

→ It keeps the both safe from outside interference and misuse.

Inheritance :-

→ It is the process in which one object can aquire the properties of another object.

polymorphism :-

It means many forms i.e which allows one interface to access a general class of actions.

Difference b/w POP vs OOP

POP	OOP
→ program is divided into small parts called "functions"	→ "objects"
→ It follows top-down approach	→ It follows bottom up approach.
→ No Access specifiers	→ Access specifiers are private, public, protected etc
→ Adding new data & function is not easy.	→ Easy.
→ It doesn't have any proper way for hiding data so less secure.	→ It has so it is more secure.
→ function is more imp than data	→ Data is more imp than function.
→ Unreal world P.L	→ real word P.L.

* History & evolution of java :-

→ java was invented by James Gosling.

→ It is designed for interactive television as it was too advanced technology at that time.

→ first it was started by Greenteam. [java teammembers] James Gosling, Mike sheridan & patrick Naughton initiated java language project in june 1991

→ Initially, it was designed as small, but later used in electronic appliances like set-top boxes.

→ Now java is used in Internet programming, Mobile devices, games etc.

→ At begining it was called as "Greentalk" and later "Oak". It was a part of Green project.

→ Many java versions have been released till now. But all of them the version java SE 10 is stable.

* program structure of java :-

It consists of following sections

1) Documentation
2) package statement
3) Import "
4) Interface "
5) class Definition
6) main method class.

Documentation :- We can write comments in this section. these are helpful for programmer for better understanding the operation of the program.

package :- A package is a group of classes that are defined by a name. i.e If you want to declare many classes within one element then you can declare it within a package.

→ It is also an optional; without package we can also run a program without getting errors.

Syntax :- package packagename;

packagename should be lowercase letters.

Import statements :- If you want to use a class of another package then we can directly import that particular package within the class. By using the ~~keyword import~~ keyword import.

Interface statement :- Interface are like a class which includes a group of method declarations.

→ It is also an optional section.

→ We can use this when we want to implement multiple inheritances within a program.

Class Definition :- It is one of the important section. We have to define the classes in this particular section.

Main method class :- Every java stand-alone program requires main method as the starting point of the program.

→ this is an essential part of java

→ there are many classes in a program but there will be one class which defines main method.

Program :-

```
// Sample Java program - Documentation section
class DemoJava // class Definition
{
    public static void main(String args[]) // Main method
    {
        System.out.print("prasanna");
    }
}
```

* Java Virtual Machine :-

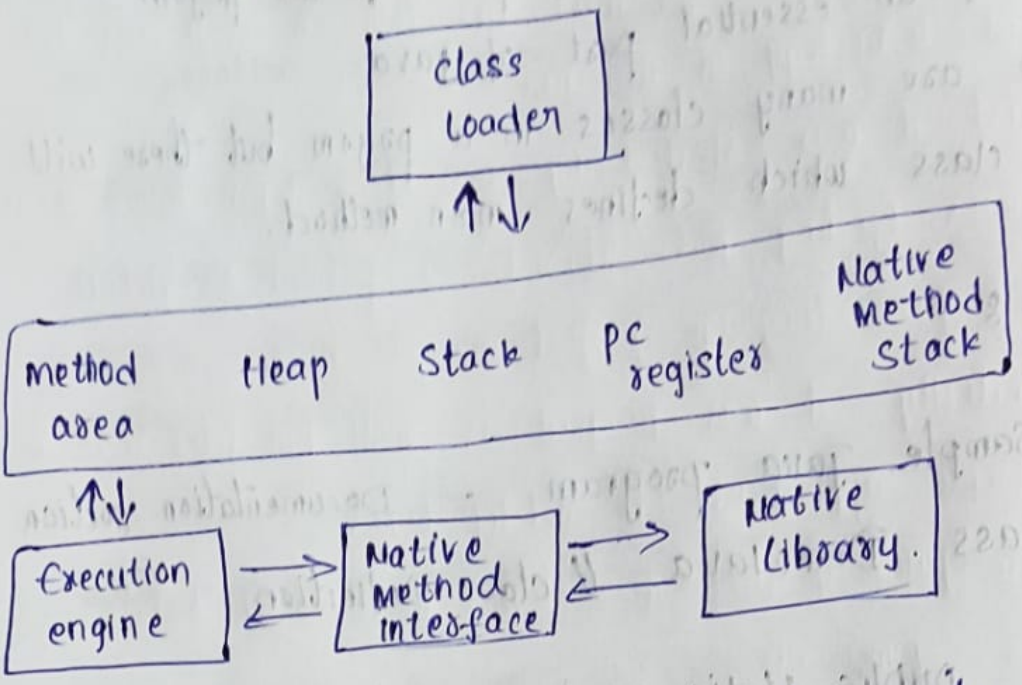
→ JVM provides run time environment to execute java bytecode.

→ We compile .java files to obtain .class files which contains bytecode.

→ JVM control execution of every java prgrm.

→ It also enables automated exception handling.

JVM Architecture :-



Class loader :- It loads the class for execution

Method area :- Stores pre-class structure as Constant pool

Heap :- It is used for allocation of Objects

Stack :- It is used to store the variables.

PC register :- It holds the address of JVM instruction currently being executed

Native method stack :- It is used to store machine code

Executive engine :- It Controls the execution of instructions in methods of classes

Native method interface :- It gives interface b/w native code & Java code during execution

Native library :- It Contains the required files for execution of native class.

* Buzzwords :-

The primary objective of java programming language creation was to make it portable, simple and secure.

Some of the features (or) Buzewords in java are :

1) Simple
2) Object-Oriented
3) portable
4) Platform Independent
5) Robust
6) Interpreted
7) secured
8) Multithreaded
9) Distributed
10) Dynamic.

Simple :- It is very easy to learn and its syntax is simple, clean and easy to understand.

Ex :-
```
public class Name
{
    public static void main(String args[])
    {
        System.out.println("My name is prassu");
    }
}
```

Object-Oriented :- It is an object-oriented programming language i.e everything in java y an object.

Q 8. class Animal
{
    void eat()
    {
        System.out.println("Eating");
    }
    public static void main(String args[])
    {
        Animal a = new Animal();
        a.eat();
    }
}

portable :- Java is portable because it facilitates you to carry the java bytecode to any platform. It doesn't required any implementation.

platform independent :- it is a platform independent i.e it can be executed on multiple platforms.

They are 2 types of platforms:

1) Software-based
2) Hardware-

But java is a software based platform that runs on top of other hardware-based platforms. It has 2 components

1) Run time Environment
2) API (Application programming interface)

Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform independent code because it can be run on multiple platforms i.e (WORA)

Ex :-
```
public class Point
{
    public static void main(String args[])
    {
        System.out.println("Prasanna");
    }
}
```

this code will be Converted into bytecode and that will be executed on any platform that's why it is platform independent.

Robust :- It means java have strong handling on Errors & Exceptions.

Ex :-
```
public class JavaExceptionDemo
{
    Public static void main(String args[])
    {
        try
        {
            int data = 100/0;
        }
        Catch(ArithmeticException e)
        {
            System.out.println(e);
        }
        System.out.println("rest of the code");
    }
}
```

o/p :- Exception in thread main java.lang.ArithmeticException: / by zero
rest of code

Secured :- Java is best known for its security. It is secured because -

- No explicit pointer
- Java programs run inside a virtual machine sandbox.

Multithreaded :- A thread is like a separate program, executing concurrently. The main advantage of multi-threading is that it doesn't occupy memory for each thread. It shares a common memory area

Distributed :- java is distributed because it facilitates users to Create distributed applications in java. RMI (Remote method invocation) & EJB (Enterprise java Bean) are used for creating distributed applications. This feature of java makes us able to access files by calling the methods from any machine on the internet.

Dynamic :- it is a dynamic language. It supports the dynamic loading of classes. It means classes are loaded on demand

Architecture-neutral :- java is architecture-neutral because there is no implementation dependent features.

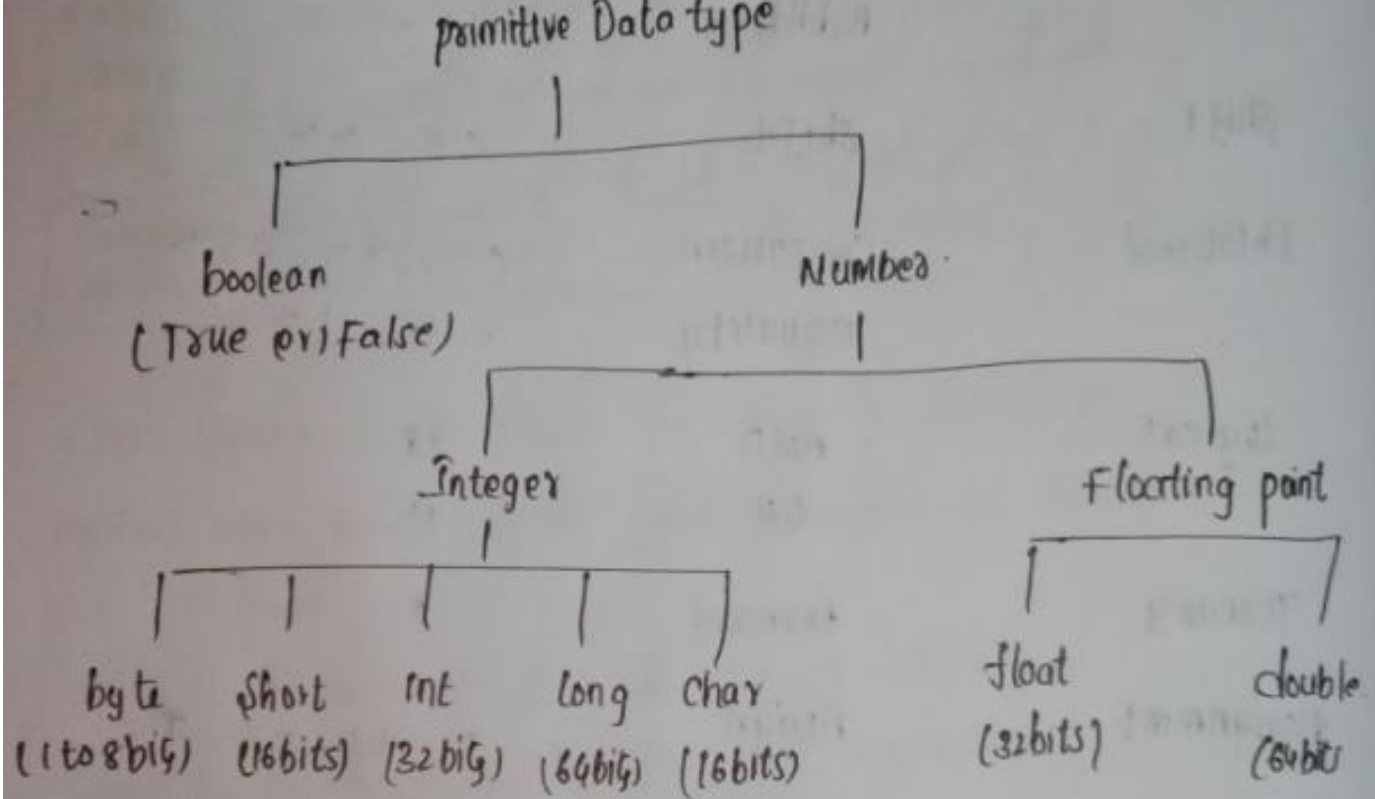
Ex:-
```
import java.util.Scanner;
class PrimitiveDemo
{
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Byte value : ");
        byte b = sc.nextByte();
        System.out.println("Default value: " + b);
    }
}
```

Variables :- which holds the data or value.

Declaration :- Access specifier Datatype name;

Ex:- public int x;

Intialization :- Access specifier Datatype name = value;



Type Conversion :-

→ Converting one primitive datatype into another is known as type Conversion (or) type casting

Two types

1) Implicit :- Converting a lower datatype into higher datatype. It should be done automatically

byte → short → int → long → float → double

2) Explicit - Converting a higher datatype into lower datatype

→ It should not be done automatically. we need to Convert it by using cast operator "( )"

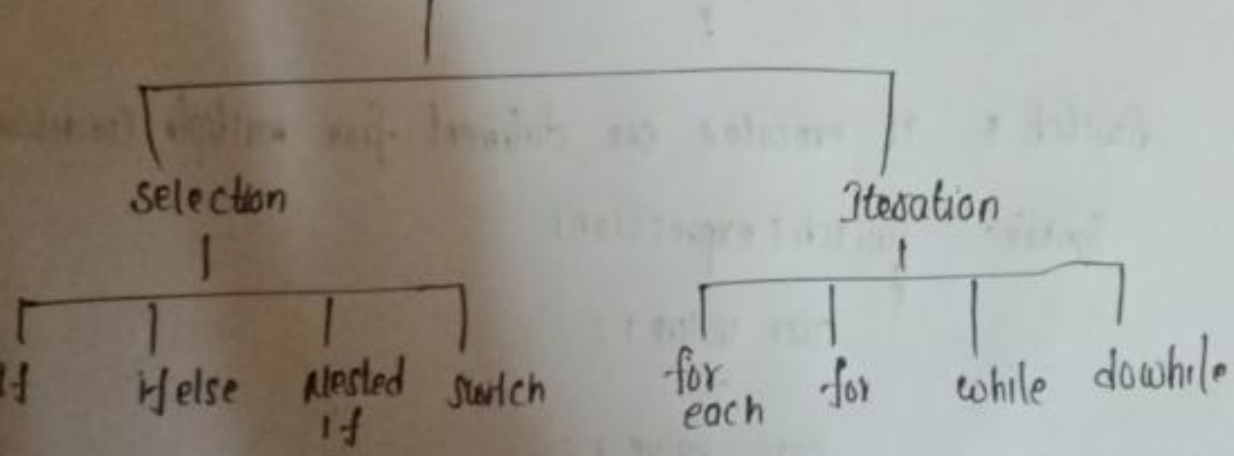
double → float → long → int → short → byte

Why we need ?

programmers need to check the Compatibility of the data type they are assigning to another data type in advance

→ By using casting we can change data type but not data

* Control statements :-



· Syntax :- 1) if (condition)
{
≡
}

2) if (condition)
{
≡
}

```
else
{
    ==
}
```

In this case, if the condition then it print the statements in if block otherwise else block.

Nested if :- It represents the if block within another if block. Here the inner if block condition executes only when outer if block condition is true.

Syntax :-
```
if (condition)
{
    ==
    if (condition)
    {
        ==
    }
}
```

Switch :- It executes one statement from multiple conditions.

Syntax :-
```
switch (expression)
{
    case value 1 :
        ==
    case value 2 :
        ==
    break ; // optional
    default :
}
```

Why we need loops ?

- To execute the multiple statements in a single statement

- for each :- It run only collection of items

Syntax :- for (type variable : collection)

* Operators :-

Operator type	category	precedence
unary	postfix	exp++, exp--
	prefix	++exp, --exp
Arithmetic	Multiplicative	*, /, %
	Additive	+, -
Shift	shift	<<, >>
Relational	Comparison	<=, >=, <, >
	equality	==, !=
logical	AND	&&
	OR	||
Ternary	ternary	?
Assignment	Assign	=, +=, -=, *=
Bitwise	AND	&
	inclusive OR	|
	exclusive OR	^

Arrays:

→ It is a Collection of homogeneous elements with same datatype in a Contiguous memory.

Syntax :- datatype ArrayName[size];

→ Instanstation means creating a memory for array object.

Instanstation :- int a[] = new int[10];

→ Intialization :- int a[] = {1,2,3,4};

NOTE :- New is also known as Operator which is used to create a memory for objects.