Basic Java-8

① Feature of Java 8 ?

Ans ① ForEach():

② Lambda Expression

③ Functional Interface

④ Default Method and Static Method

⑤ Parrallel Sort.

⑥ Java Stream

→ ForEach(): foreach() method are used to iterate the element. It is a default method of Iterable method. It takes a Single parametre we can Pass lambd Expression is an Argument.

```
Interface ABCI {

    Public Void add();

}
```

```
Public Class ABC {
Public Void Static Main (String[] args) {
ArrayList<String> A = New ArrayList();
    A.add("Shubhanshu");
    A.add("Saxena");
    A.add("Bhopal");
A.forEach (Li -> System.out.Println(Li));
```

→ lambda Expression :-

lambda Expression provid the Simplest and Shortest way to define annonymous function.
lambda Expression define the body of Functional Interface
lambd Expression is define with the new operator from Java-8 i.e Arrow Operator (->).

→ Functional Interface :-

Functional Interface is an Interface that have only one abstract Method and number of defaut () and Static ().
Functional Interface is define with the help of @FunctionalInterface.
@FunctionalInterface is ensure that the they Contain ony One abstrart method.

→ Defaut () and Static() :-

Before JDK 1.8, Interface have only one abstract method but after JDK 1.8 Interface have One abstract Method as well as Multiple Default Method and Static Method too. So that developer have oppourtnity to add multiple method with out affecting the in chasses that implement this Interfaces.

```
Interface FunInt {

    Public Void Say ();

    Public static Void show();

    Public Default Void display(){ }
```

Java Stream :-

for sequential and parallel operation with collection Usually with huge amount of data, to perform filter/map/reduce like operation, Java 8 ~~Provide~~ introduce a New feature Stream API that exist in java.util.stream Package which provide bulk operation on Collection such as Sequential or Patted Parallel Map-Reduce Transformation

it Provide two special method to generate stream :

① stream() : Return Squentianal stream on collection

② parallelStream : Return Parrallel stream on Collection.

→ Parallel Sort

To short Sort array elements parallel Java Provide New method parallelSort() in Java.util.Arrays Package

```
Int[] arr = {2,3,5,1,0,8,6,4,8}
Arrays.parallelSort(arr)
```

Java-9 feature :-

1. JShell (REPL)
2. Private Method in interfaces
3. Factory Method for Immutable List, Set, Map and Map Entry
4. <> (Diamond) operator Enhancement
5. (_) UnderScore Keyword.
6. JLink (Java Linker / Custom JRE)

→ JShell (REPL) : JShell is a Read-Evalute Print loop → Java added JShell from JDK-9, by that we can immidiatly evalute the Result and makes the changes as needed.

→ Private() in interfaces :-

For sharing the Common Code between two default method or non-abstract method Java-9 introduce private method that Can not be overridden in implementation Class classes and Can not be abstract.

→ Factory Method for Immutable for List, Set, map and map Entry :-

Java 9 provide us a New feature Called factory() that Creates Immutable or you Can say an Unmodified Small Collection of List, Set and Map.

→ <> (Diamond) operator Enhancement :-

Now in Java-9, we Can use diamond operator with annonymous classes that were not allowed in Java 7

→ Underscore (_) Keyword :-

From Java 9 we cannot use Underscore (_) as a indentifier. Now it becomes a Keyword in Java 9. if we add it as indenfier it will throw a Compile-time error.

6) Jlink (Java Linker / Custom JRE)

Now from Java 9, we can Create our own Custom JRE to execute Small program on our System. Till Now we have to use default JRE that is provide by Java

If we Run a Small Program on Java like hello World we need only 4 to 5 classes to Run a program but default JRE runs 4300+ classes to Run a Simple Program.

Key Component of Java :-

1. JVM
2. JRE
3. JDK
4. Libraries

JVM :- JVM is the Virtual Machine that run the Java bytecodes.
when Java Program (*.Java) is Compiled it generates class (.class) file that Contain the bytecodes understandable by JVM.

JRE - Java Runtime Environment JRE Consist of JVM and Java Core libraries, it Contains necessary functionalities to execute Java application or programs.

JDK :- JDK is a Superset of JRE, and Contain Every thing that is in JRE moreover tools such as Compiler and debugger necessary for developing applets and application.

Feature of Java :-

1. Platform Independent
2. object oriented Programming lang.
3. Strongly-typed programming lang.
4. Automatic meomery management
5. Interpreted and compiled language.

Garbage Collector :-

JVM automatically releases the Meomory of object which is not referred by any other object

Class Loader :-

Java class loader is a part of JRE. it dynamically loads the Java classes into JVM.

① Bootstrap class loader :-

It loads Core Java libraries located in the <JAVA-HOME>/Jre/lib directory. it is part of JVM and Written in Native Code

② Extensions class loader :-

It loads the classes from extension directories

③ System class loader :-

It loads classes from Java.class.Path, which maps the classpath enviroment Variable.

① Bootstrap class loader :-

It loads Core Java libraries located in the <JAVA_HOME>/Jre/lib directory. it is part of JVM and Written in Native Code.

② Extensions class loader :-

It loads the classes from extension directories

③ System class loader :-

It loads classes from Java.class.Path, which maps the classpath enviroment Variable.

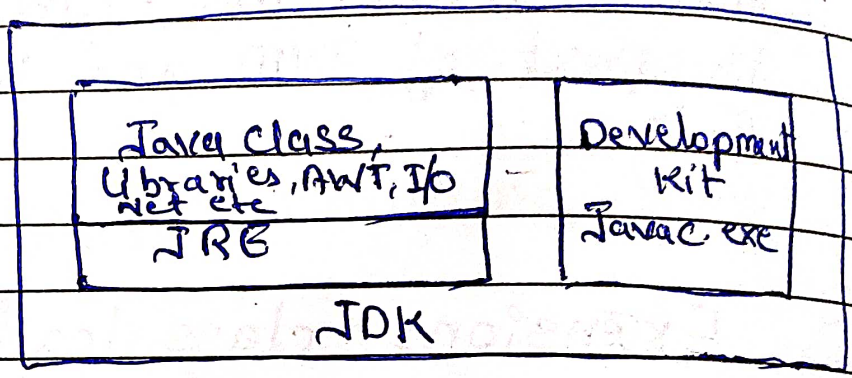
① Key Component of Java :-

① JDK

② JVM

③ Libraries

④ JRE



① JVM (Java Virtual Machine) :-

JVM is a Virtual Machine that Run the Java bytecodes. when Java program (.Java) is Compiled it generates (.class) file that Contain byteCodes Understandable by JVM.

② JRE (Java Runtime Enviroment) :-

JRE Consist of JVM and Java Core libraries. it Contains necessary functionality to execute Java application or Program.

(1) JDK (Java Development Kit) :-

JDK is a Superset of JRE, and Contain everything that is in JRE. more over tools such as Compiler and debbuger necessary for developing applets and application.

(2) Feature of Java :-

① Platform Independent

② object oriented Programming language.

③ strongly typed Programmin langeage.

④ Automatic Meomery Management.

⑤ Key feature of Java program it to Compile once and run anywhere.

(3) Garbage Collection ?

Ans JVM automatically release the meomory of object which is Not being leffered by any other object. or Garbage Collection destroy orphan object and release their meomery.

→ but Developer still have to ensure that orphan they are not holding reference of Unwanted object otherwise Garbage Collector can not Release them.

→ Keeping orphan object references in program is called memory leak.

③ ClassPATH :-

classPATH is a PATH of directory or list of directories that contains Java.class file
Java Compiler at Runtime look these directories to find classes required to load in the memory to execute a Java Program

classpath can be set by two ways

① Set by classPATH environment Variable.

② Specified at Command line by -cp or -classpath attribute

4) Class Loader :-

It dynamically loads Java classes into JVM.

There are 3 type of class loader:

① Bootstrap class loader :-

It loads Core Java libraries in the <JAVA_HOME>/Jre/Lib directory. It is a part of JVM and written in Native Code

② Extention class loader :-

It loads classes from extention directories located by <JAVA_HOME>/JRE/Lib/Ext.

③ system class loader :-

It loads classes from java.class.path, which maps to the classPATH enviroment Veriable

5) How you make custom class loader?

Ans By extends Java.lang.classloader.

Class Loader :-

It dynamically loads Java classes into JVM.

There are 3 type of class loader:

① Bootstrap class loader :-

It loads Core Java libraries in the <JAVA_HOME>/Jre/Lib directory. It is a part of JVM and written in Native Code

② Extention class loader :-

It loads classes from extention directories located by <JAVA_HOME>/JRE/Lib/Ext.

③ system class loader :-

It loads classes from java.class.path, which maps to the classPATH Enviroment Variable

⑤ How you make Custom Class loader?

Ans By extends Java.lang.classloader.

6) What is Variables ?

Ans Variable is a name of a meomery location.

There are Three types of Variable in Java

① Local Variable

② Instance Variable (non-static field)

③ class Variable (static field)

→ local Variable :-

A Variable that is declared inside the method called local Variable.

→ Instance Variable

A Variable that is declared inside the class but outside the method.

o-> Int. Instance Variable can have access modifier

o-> It can be defined as final or transient.

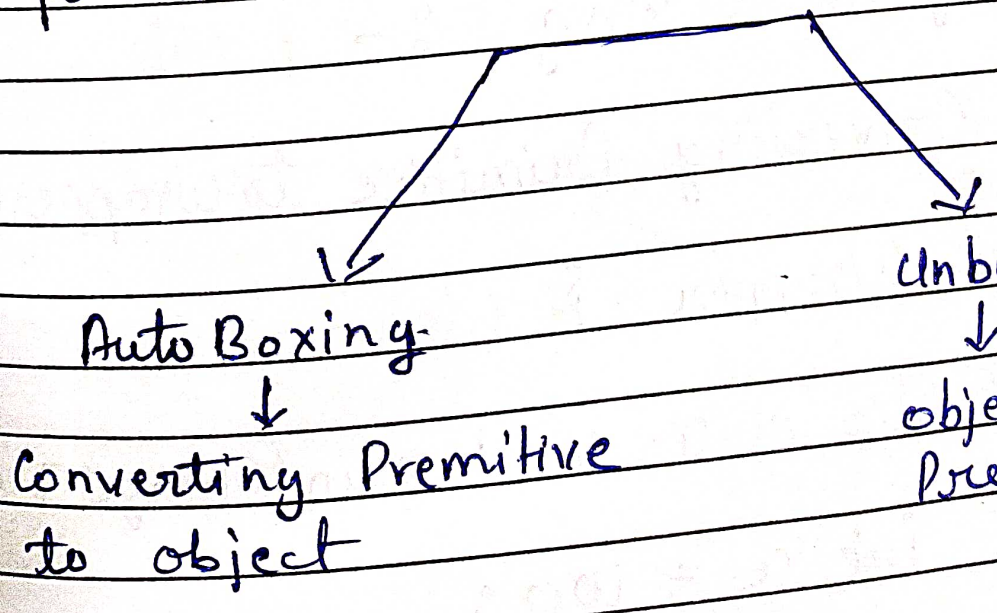
-> class Variable :-

class Variable is a field declared with the Static modifier. This tells JVM to allocate memory only once in a life of static Variable

o-> class Variable can be accessed with object or class Nam

7) Wrapper classes in Java

Ans Wrapper classes are used to convert primitive into object Vice-versa



o→ It can be defined as final or transient.

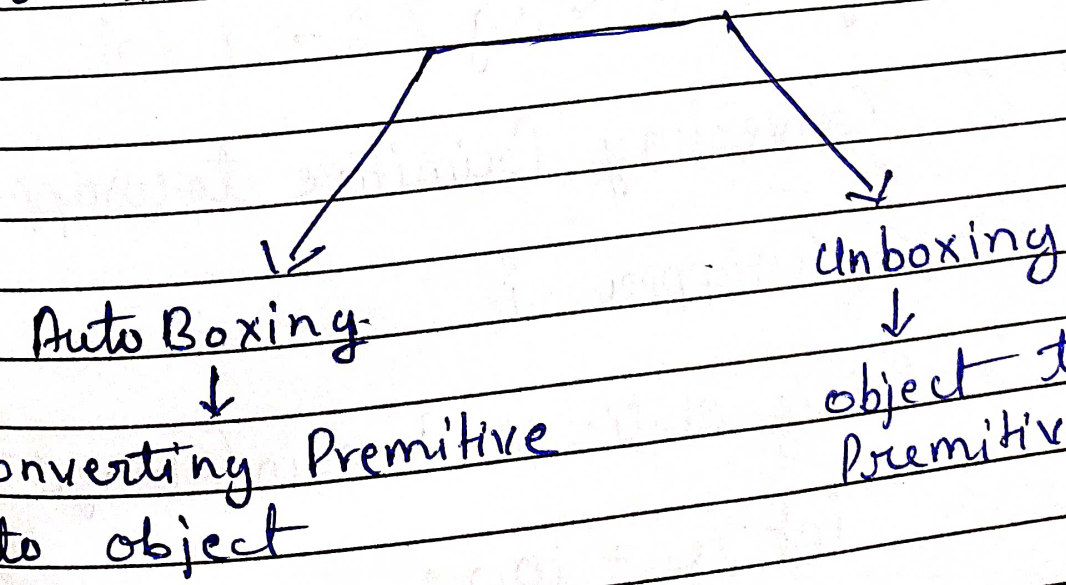
→ class Variable :-

class Variable is a field declared with the Static modifier. This tells JVM to allocate memory only once in a life of static Variable

o→ class Variable can be accessed with object or class Nam

7) Wrapper classes in Java

Ans Wrapper classes are used to convert primitive into object vice-versa



Primitive type	Wrapper class
boolean	Boolean
char	character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double

Example of Autoboxing :-

```
// Converting Primitive to wrapper(object).
class Wrapper {
    Public static void main(String[] args) {
        int a = 100;
```

```
Integer b = Integer.valueof (a) ;
        // Integer.Valueof done by Compiler
        // automatically

Integer c =  a ; // this is Autoboxing
        }
```

Example of Unboxing :-

```
Public class Wrapper {
// Converting Wrapper to Primitive
Integer I = New Integer (100) ;
int j = I.intValue() ;   (or)
int K = I ;
SOP (K) ;
```

Short Cut of Debug :-

1. Ctrl + F11 - Run
2. F11 Debug
3. F6 Step over [Next line with execute Current line]

F5 Step into (method)
F7 Step out
F8 Jump into nearest brk point.

ctrl+shift+I Inspect.

```
Public class ABC {
Public Static Void Main (String[] args) {
    int a = 10
    int b = 5;
if (a>b) {
  Sysout ("a is greater")
      }
else {
  sop ("b is greater")
      }
}
```

```
Public class Anish {
    Public Static Void Main (String[] args) {
        int a=5 ;   int b=10 ;   int c=20
    if (a>b) {
        if(a>c) {
            SOP ("a is greater") ;
        }
    }
    else if(b>c) { SOP ("c is greate") ;
        }}
    else if (b>c)
    {
        SOP ("b is greate") ;
    }
    else
    { SOP ("c is greate") ;
    }
    }
```

Write a Program to generate 5 random number blw 1 to 100

```
Public class ABC {
    P S V M (String[] args) {
        for (i=1; i<=5; i++) {
            SOP (" " + (int)(Math.random() * 100));
```

WAP to find factorial of given number?

```
Public class ABC {
    P S V M (String[] args) {
        int no = 5 , fact = 1.
        for (int i=1; i<=no; i++)
        {
            fact = fact x i;
        }
        Sop (fact);
    }
}
```

	Integer.ParseInt()	Integer.valueOf()
1)	it only take a String as a parameter.	It can take a String as well as Integer as parameter
2)	it return a primitive int value	It return an Integer object.

2) Should we use ParseInt or ValueOf ?

Ans if you need just native type, use ParseInt. If we need an object, use ValueOf.

3) what is ParseInt method in Java ?

ParseInt function converts its first argument to a String, Parse that String then Return an Integer or NaN.

Java Convert String to int,

We Can Convert String to an int in Java Using Integer.ParseInt() method.
To Convert String into Integer, we Can Use Integer.ValueOF() which returns the instance of Integer class.

Convert String to Primitive int.

```
Public class ABC {
public static Void main (String[] args) {

    String s = "200";

    int i = Integer.ParseInt(s);

    Sop (i);        {OP= 200}
    }
}
```

Convert String into Integer obj

```
String s = "200";

Integer i = New integer.ValueOF(s)

SOP (i)
```

Java Convert String to int,

We can Convert String to an int in Java using Integer.ParseInt() method.
To Convert String into Integer, we can Use Integer.ValueOF() which returns the instance of Integer class.

Convert String to Primitive int.

```
Public class ABC {
public static Void main (String[] args) {
    String s = "200";
    int i = Integer.ParseInt(S);
    Sop (i);        {OP= 200}
    }
}
```

Convert String into Integer obj

```
String s = "200";
Integer i = New integer.ValueOF(S)
SoP (i)
```

Java - 8

① Foreach method

```
ArrayList<String> Str = new ArrayList();

Str.add("S");
Str.add("H");
Str.add("U");

Str.forEach(i -> System.out.println(i));
```

② Lambda Expression

It provide Simplest and shortest way to define annonymous function. It ~~define~~ define the body of functional interface define by New operator that is (->) arrow operator

```
Interface Abc {
P.S.V.M(String[] args) {
    Int Calculate(int x, int y);
}
class Abc1 {

    Abc c = (x,y) -> (x+y);
    int result = c.Calculate(a,b)

        SOP(result);
    }
```

Java - 8

① Foreach method

```
ArrayList<String> Str = new ArrayList();

Str.add("S");
Str.add("H");
Str.add("U");

Str.forEach(i -> System.out.println(i));
```

② Lambda Expression

It provide Simplest and shortest way to define annonymous function. It ~~define the~~ It define the body of functional interface define by New operator that is (->) arrow operator

```
Interface Abc {
P.S V M(String[] args) {
    Int Calculate(int x, int y);
                }
class  Abc1 {

    Abc c = (x,y) -> (x+y);
    int result = c.Calculate(a,b)

        SOP (result);
    }
```

(3) Functional Interface :-

An interface that Contain only 1 abstract method and No. of static and default() @FI defined

① Reverse a Number

```
class ABC {

Public static void main (String[] args) {

int a = 123;
int r = 0;

while (a>0) {

r= a% 10

System.Out.Println (r);

a = a/10;

}

}

}
```

DATE
PAGE

⑤ Random 5 number b/w 1 to 100

```
class ABC {
Public static void main(String[] args) {
for(int i=1 ; i<=5 ; i++) {
System.Out.Println ((int)(Math.random()*100));
}
}
}
```

③ Reverse String

```
Class ABC {
    Public static void main (String[] args) {
    String S = "MADAM";
    String rev = "";
for (int i = s.length()-1; i >= 0 ; i--) {
    rev = rev + S.charAt(i);
}
SOP (rev);
}
}
```

```
if S.equal(rev) {
        SOP (Palindrone)
        }
else {
        SOP (Not Palindrone)
    }
}
}
```

4) Swap ~~two~~ three int using two int?

```
class ABC {
    Public static Void main (String[] args) {
        int a = 5 ;
        int b = 10 ;

    SOP ( a = a+b ) ;
    SOP ( a = a-b ) ;
    SOP ( b = a-b ) ;

    }
}
```

⑤ Find Factorial of given number

```
Public class ABC {

Public static Void main (String[] args) {

int no = 5 ; Fact = 1

for(int i=1 ; i<=no ; i++) {

Fact = Fact * i

}

SOP (Fact) ;

}

}
```

4) ParseInt Program

```
// Convert String to an int Primitive in Java
// Using Integer.ParseInt() method ;

Public class ABC {

    Public static void main (String [] args) {

        String s = "10";
        int h = 5;

    int i = Integer.ParseInt (s);

        SOP (s+h) ;     outPut :- 105
        SOP (i+h) ;     outPut :- 15

    } }
```

5) ValueOF() Program

```
// to Convert string to Integer class obj
// we use ValueOf() ;
```

```
Public class ABC {
Public static void main (String[] args) {
String S = "1034" ;
Integer I = Integer.ValueOF(S) ;
SOP(I);
} }
```

6) What is Wrapper class

Ans to Convert primitive into object vice-versa

① autoboxing ② Unboxing

↓ ↓

Convert Primitive to object. Covert object to Primitive

```
Public class ABC {
Public static Void main(String[] args) {
int a = 5;
Integer i = a or
Integer i = Integer.Valueof(a) // by Jvm
} // autoboxing;
```

```
Public class ABC {
    Public static void main (String[] args) {
        Integer I = New Integer(100);
        int a = I;    or
        int a = I.intValue();  // by JVM
    }
} // Unboxing.
```

8) type Casting :- type Casting is Used to Convert one data type to another data type

1) Primitive data type Casting
2) Reference data type Casting

Primitive data type Castiong :-
Implict type Casting :-
Convert lower data type to higher data type it automatic Converted by JVM

byte → short → char - int - long - float - double
Widening →

```
class ABC {
Public static void main(String[] args) {
char c = 's';
double d = c;  Implict Conversion
} }
```

Explicit type Casting :-

Convert higher data to lower datea

double → Float → long → int → char - short - byte

Norrowing

```
Public class ABC {
Public static void main (String [] args ) {
double d = 12.3
long l = (long) d;
```

9) Reference data type Casting

```
Public class ABC { ... }
Public class XM2 extends ABC {
    ABC a = New XM2 ;
ABC a = New ABC() ; //widening

XM2 x = (XM2) S ; Narrowing
```