UNIT – 1

JAVA Programming

★ **JAVA :–** JAVA is a programming language and a platform. Java is a high – level language, object oriented (not pure object-oriented) programming language and secure programming language.

1.	Paradigm	Object-oriented, multi-paradigm class based, oops based, reflective etc.
2.	Design by	James Gosling
3.	Developer	Sun Microsystems
4	Appeared year	23 May, 1995

★ **History of JAVA :–** James Gosling, Mike Sheridan and Patrick Naughton initiated the JAVA language project in June 1991. JAVA was originally designed for interactive television but it was advanced for the digital cable television industry that time. Firstly JAVA was known by OAK, later this language was known by JAVA. This team was known by 'green'.

without need of source code.

★ Principle of JAVA :-

1- It must be simple and object-oriented (not pure) programming language.
2- It must be robust and secure.
3- It must be architectured and portable.
4- It must be dynamic, threaded and interpreted.
5- JAVA is an auto-garbage collection programming language.

★ Version of JAVA :-

The latest version of Java is Java 12 or JDK 12 released on March, 19 2019.
There are three main Java platforms : Java SE (Standard Edition), Java ME (Mobile edition), Java EE (Enterprice edition).

★ Various fields where we use JAVA -

There are many places where Java is used in real world, starting from commercial e-commerce website to android apps, from scientific application to financial applications like electronic trading system from games like - Mine craft to desktop applications like - Eclipse, Netbeans and intellij

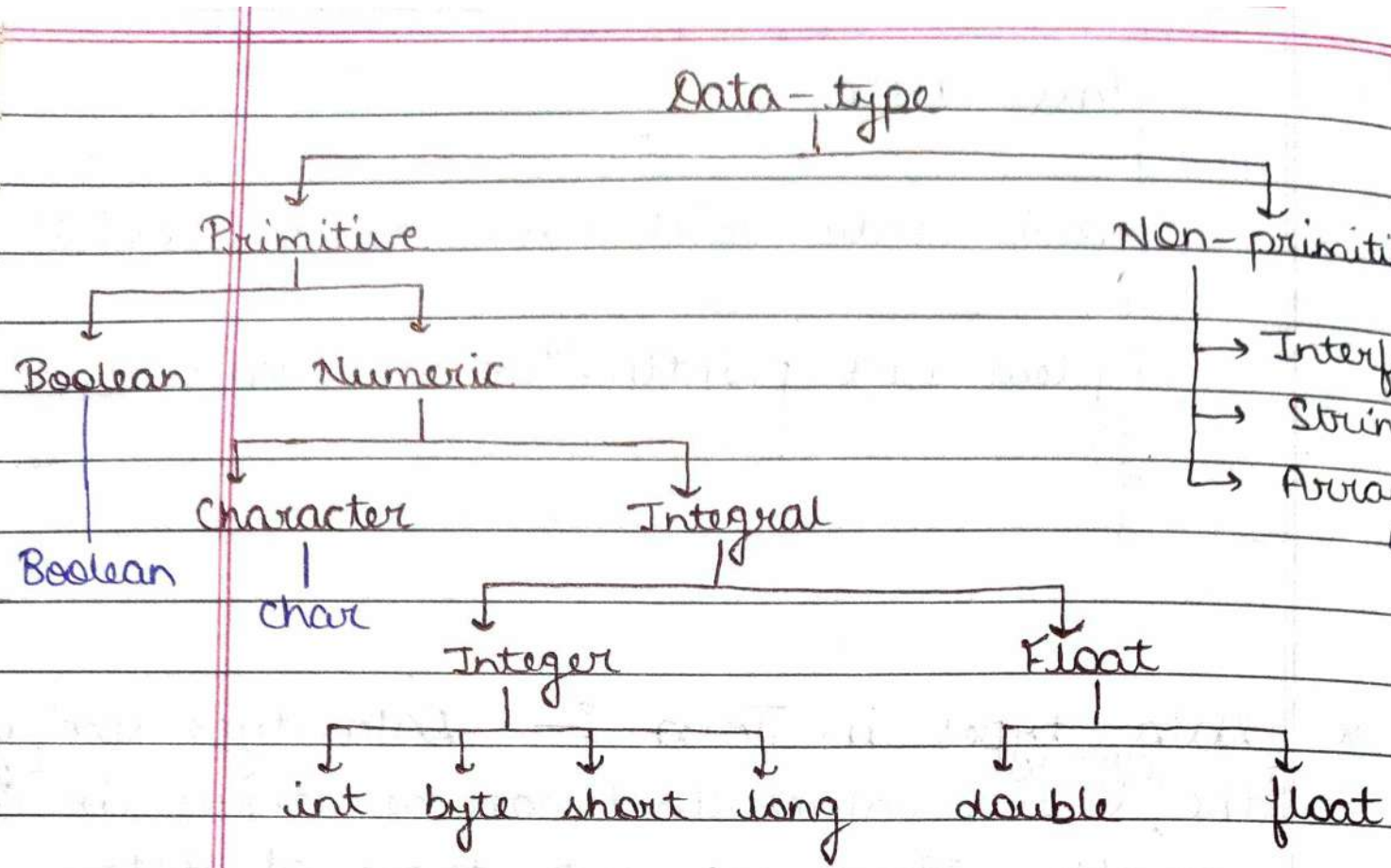
```
class Hello
{
public static void main(String args[])
{
System.out.println("welcome Java");
}
}
```

* Data types in Java :- Data type specify the different sizes, values that can be stored in variable. There are two types of data-types in Java -

1- Primitive
2- Non-primitive

1. Primitive data-type - Primitive data-type include boolean, char, byte, short, integer, double, long, float etc.

2. Non-primitive - Non-primitive data type includes interface, classes and array.



Data type	Default value	Size
Boolean	false	1 bit
char	'\u0000'	2 byte
byte	0	1 byte
short	0	2 byte
int	0	4 byte
long	0l	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

Long	-9223372036854775808 to 9223372036854775807
Float	1.4e-045 to 3.4e+038
Double float	4.9e-324 to 1.8e+308
Char	0 to 65536

```
                                    → user define
public class Example Data Type
{
public static void main(String args[])
{
int myNum = 10;
float myFloat = 5.99f;
char myChar = 'D';
boolean myBool = true;
String myText = "Hello Java";
System.out.println(myNum);
System.out.println(myFloat);
System.out.println(myChar);
System.out.println(myBool);
  System.out.println(myText);
}
}
```

* **Control Statement (Structure) :-**

A Java program normally execute from top to bottom but if we want to control flow of execution of the program, based on logic and values we use control statements.

```
public class ABC
{
public static void main(String args[])
 {
  condition (logic)
  condition (body)
 }
}
```

Flow control or flow of the control is the order in which instructions, statement and function call being executed or evaluated when a program is running. There are 3 types of control statement in Java -

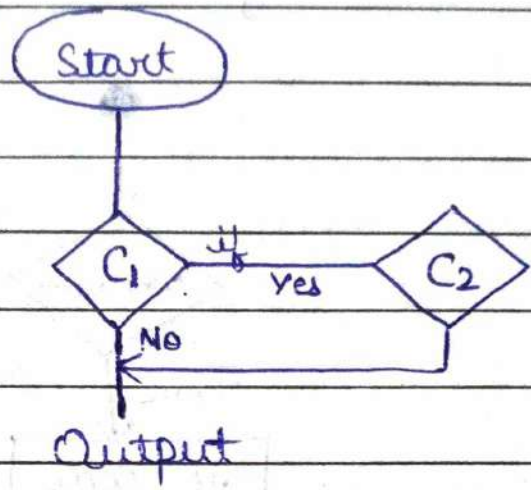
1. Selection statement - if, if--else, Nested if,
2. Iteration statement
3. Jump statement

1- Selection statement -
Ex- if, if--- else, Nested if, Nested if-- else, Nested if--- else if, Switch.

2- Iteration statement -
for, while, do while

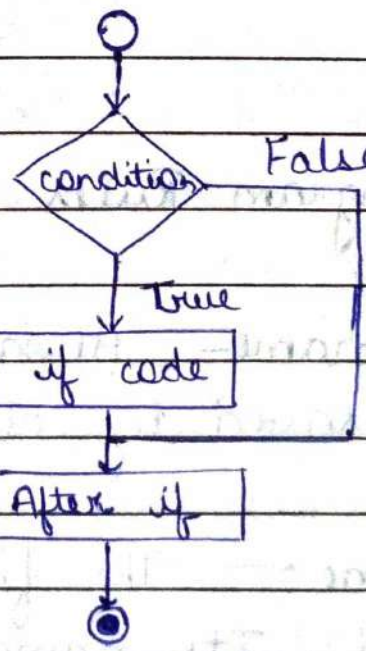
3. Jump statement
break, continue, return

allow us to control the flow of program execution on the basis of the outcome of an expression (ex- condition).



(i) Java if Statement — The Java if statement tests the condition. It executes the 'if' block if condition is true.

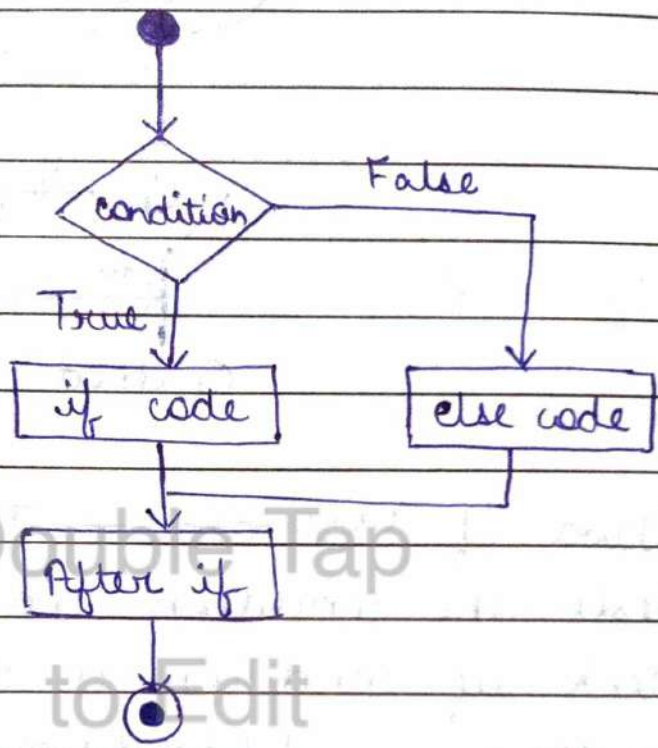
Syntax— if (condition) {
}



(ii) Java if-else Statement — The Java if-else statement also tests the condition. It

the 'if' block if condition is true otherwise 'else' block is executed.

Syntax – if (condition) {
} else {
}



★ Java Program Rules and Conditions :-

1. Program name – Always Java program will be saved by class name.

2. Class name – The first letter should be capitalised. Its name first alphabet will be in uppercase and others in lowercase.
ex example – AmitKumar (No space)

3. Object name — Always object will defin with the 'new' keyword.

4. Method name — The first letter should be in lowercase then normal rule follows
Ex- amitKumarSingh( )

5. Variable name — Its also first letter will be in lowercase and in the las it stopped by semicolon (;).

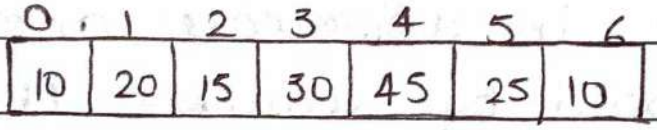
6. Comments in program — If we want definition or declaration any method, variable, object, class etc they are explained with the help of comments.

7. Constants — Java constants are created by marking on variable static and final. They should be named using uppercase letters with underscore character for seperation.
Ex- HIGH_NUMBER
MIN_NUMBER

★ Java array :- Normally an array is a collection of similar type of elements

element of a similar data type. It is a data - structure where we store same data and we can only store fixed sets of data.

Array in Java is indexed based which mean the first element of the array stored at the position of 0.



Array

Advantages of Array -

1. Code optimization
2. Random access

Disadvantages of Array -

1. Size is fixed
2. Only similar-type of data can store.

* Types of array in Java - There are two types of array -

1- Single - dimensional array
2- Multidimensional array

1- Single - dimensional array -

Syntax - dataType[ ] arr; (or)
dataType arr[ ];

Instantiation of an array in Java -

arrayRefVar = new datatype[size];

Example of Java array -

```
class Testarray
{
public static void main(String args[])
{
int a[] = new int[5];
a[0] = 10;
a[1] = 20;
a[2] = 70;
a[3] = 40;
a[4] = 50;
for(int i=0; i<a.length; i++)
System.out.println(a[i]);
}
}
```

Output -

10
20
70
40
50

2- Multidimensional array - In such case, data is stored in row and column based index (also known as matrix form).

Syntax —

```
dataType [][] arrayRefVar;   or
dataType arrayRefVar[][];    or
dataType [] arrayRefVar[];
```

Example to instantiate Multidimesional array:

```
int[][] arr = new int[3][3];
```

Example to initialize—

```
1. arr[0][0] = 1;
2. arr[0][1] = 2;
3. arr[1][1] = 5;
4. arr[2][1] = 8;
5. arr[2][2] = 9;
```

Example of Multidimensional Java array—

```
class Testarray3
{
public static void main(String args[]),
{
int arr[][] = {{1,2,3},{2,4,5},{4,4,5}};
for (int i=0; i<3 ; i++)  {
for (int j=0 ; j<3 ; j++)  {
System.out.print(arr[i][j]+" ");
}
System.out.println();
}
}}
```

Output —

1 2 3

2 4 5

4 4 5

★ Vector :- Vector class contain in java.util package. The vector class can be used to create generic dynamic array, that hold object of any type or number.

Ex-

```
import java.util.Vector;
public class A
{
public static void main(String args[])
{
Vector v = new Vector();
v.add("Hello Java");
v.add("15");
v.add("10.55");
for(int i=0; i<v.size(); i++)
{
System.out.println(v.get(i));
}
}
}
```

Output- Hello Java
15
10.55

★ Declaration of vector — There are two ways to declare a vector —

1- Declaring without size —

```
Vector list = new Vector();
```

2- Declaring with size -

```
Vector list = new Vector(5);
```

* String :- String is a sequence of characters. String is an object that represent the sequence of character. It belongs to 'java.lang' package. In java, string is immutable.

Ways to create string object - There are 2 ways to create string object in java -

1- By string literals

```
String s = "abc"
```

2- By new keyword

```
String s = new String("Hello")
```

* Methods of String -

1- charAt()

Returns the character at the specified index.

2- concat()

Concatenates the specified string to the end of this string.

3- intern()

Returns a canonical representation for the string object.

4- compareTo()
Compares this string to another object.

5- length()
Returns the length of this string.

6- replace()
Returns a new string.

7- replaceAll()

8- replaceAll()

9- replaceFirst()

10- startsWith()
Test if this string starts with the specified prefix.

11- toLowerCase()

12- toUpperCase()

13- equal()
Compares this string to the specified object

14- trim()
Returns a copy of the string, with leading and trailing whitespace omitted

Example of intern —

```
class a
{
public static void main(String args[])
 {
String a1 = new String("Hello")
String a2 = "Hello";
String a3 = a1.intern(); // it will same as stack memo address data a2
System.out.println(a1==a2)
}
}
```

Output — Hello.

- Immutable — Once string object is created its data or state cannot be changed but a new string object is created.

★ Types of string :-

1. String
2. String buffer
3. String builder

1- String — In Java, string is basically an object that represents sequence of char values. An array of characters works same as Java string.

For ex-

```
char[] ch = {'j','a','v','a'};
String s = new String(ch);
```

is same as -

```
String s = "java";
```

Java String class provides a lot of methods to perform operations on string such as compare(), concat(), equals(), split(), length(), replace(), intern() etc.

2- Java StringBuffer Class - Java StringBuffer class is used to create mutable (modifiable) string. The StringBuffer class in java is same as String class except it is mutable i.e. it can be changed.

StringBuffer may have characters and substrings inserted in the middle or appended to the end.

Some of the most used methods in StringBuffer are - length() and capacity, append(), insert(), reverse(), delete() and deleteCharAt(), replace(), trim to size() etc.

3- Java StringBuilder class - StringBuffer is identical to StringBuffer except for one important difference that it is not synchronized, which means it is not thread safe.

(ii) Because StringBuilder methods are not synchronized.

StringBuilder Constructors –

1. StringBuilder()
2. StringBuilder(int size)
3. StringBuilder(String str)

Methods of StringBuffer class – append(), insert(), replace(), delete(), reverse(), capacity(), length() etc.

★ **Life Cycle of a Thread :-**

A thread can be in one of the five states. A thread lies only in one of the shown states at any instant –

1. New
2. Runnable
3. Running
4. Non-Runnable (Blocked)
5. Terminated

According to sun, there is only 4 states in thread life cycle – new, runnable, non-runnable and terminated. There is no running state. The life cycle of a thread in java is controlled by JVM.

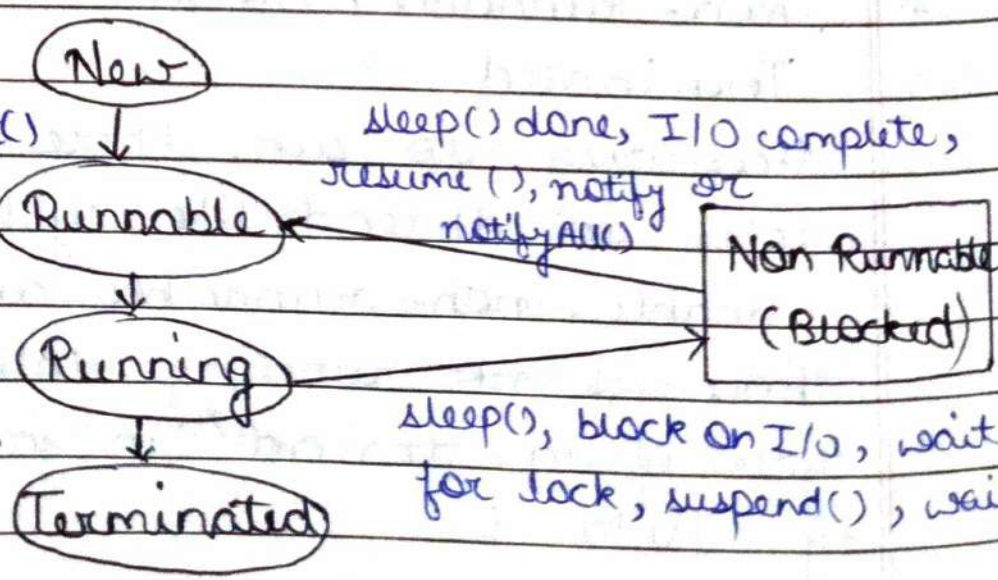
1- New — The thread is in new state if you create an instance of Thread class but before the invocation of start() method.

2- Runnable — The thread is in runnable state after invocation of start(), but the thread scheduler has not selected it to be the running thread.

3- Running — The thread is in running state if the thread scheduler has selected it.

4- Non-Runnable (Blocked) — This is the state when thread is still alive, but is currently not eligible to run.

5- Terminated — A thread is in terminated or dead state when its run() method exits.



Life-cycle of a thread

* Multithreading :- Multithreading in java is a process of executing multiple threads simultaneously. Thread is basically a lightweight sub-process, a smallest unit of processing. Multiprocessing and multithreading both are used to achieve multitasking.

Threads share a common memory area. They don't allocate seperate memory area so saves memory, and context-switching between the threads takes less time than process.

Advantages of Java Multithreading -

1. It doesn't block the user because threads are indepent and you can perform multiple operations at same time.
2. You can perform many operations together so it saves time.
3. Threads are independent so it doesn't affect other threads if exception occur in a single thread.

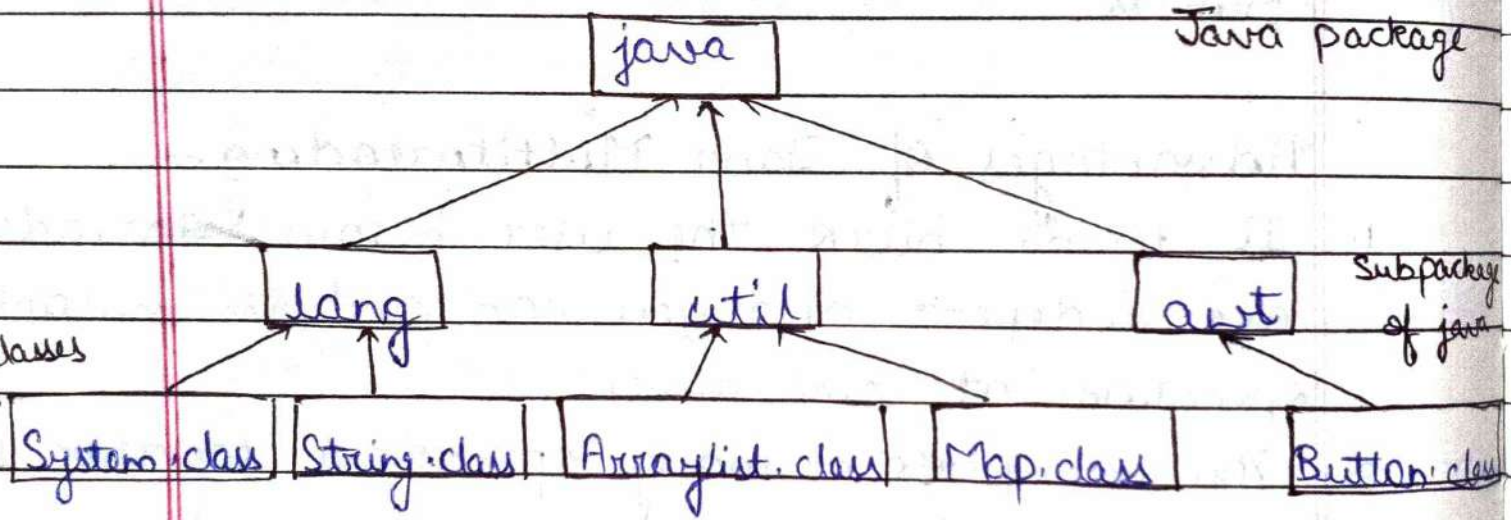
* Java Package — A java package is a group of similar types of classes, interfaces and sub-packages.

Package in java can be categorized in two form, built-in package and user-defined package. There are many built-in

swing, net, io, util, sql etc.

Advantages of Java Package-

1. Java package is used to categorize the classes and interfaces so that they can be easily maintained.
2. Java package provides access protection.
3. Java package removes naming collision.



Example of java package-

The 'package' keyword is used to create a package in java.

```
package mypack;
public class Simple {
public static void main(String args[]){
System.out.println("Welcome to package");
}
}
```

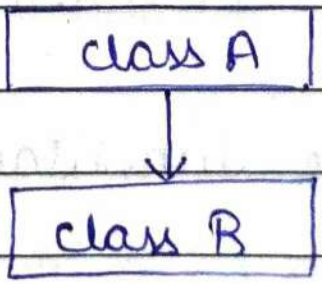
★ Inheritance :- Inheritance in java is a mechanism in which one object acquire all properties and behaviour of parent object. The idea behind inheritance in java is that you can create new classes that are build upon existing classes.

You can inherit parent class data with the help of 'extend' keyword. With the help of inheritance you can reuse methods and reduce the code in your program.

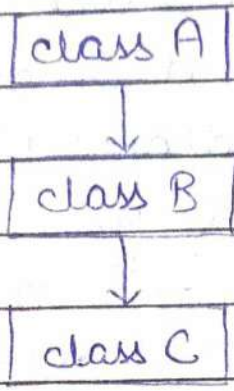
Types of inheritance — There are 4 type of inheritance in java —

1- Single inheritance
2- Multi-level inheritance
3- Hierarchical inheritance
4- Hybrid inheritance

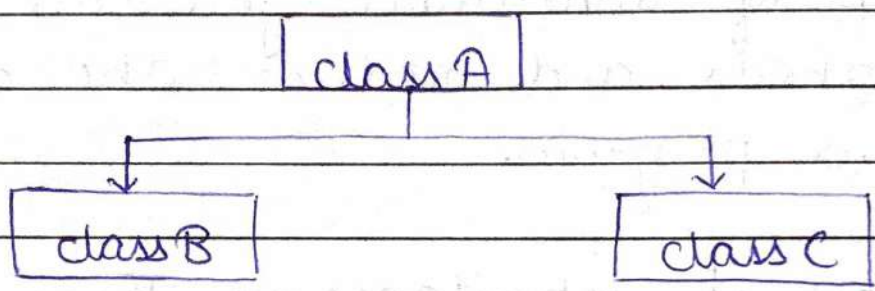
1- Single inheritance



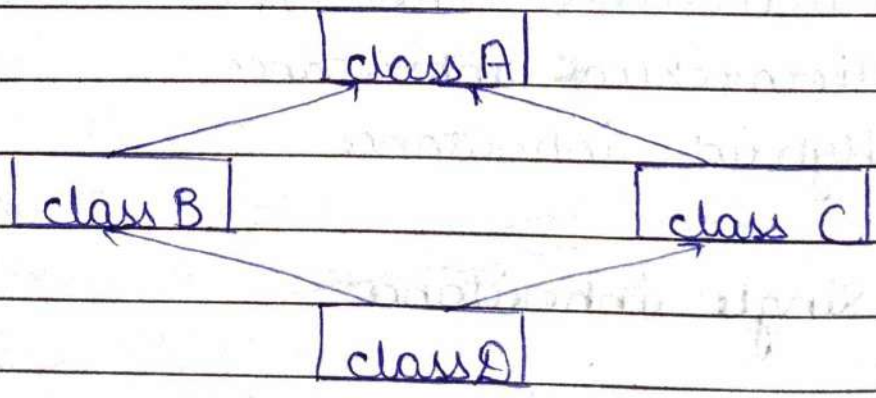
2- Multi-level inheritance —



3- Hierarchical inheritance —



4- Hybrid inheritance —



* Why multiple inheritance is not supported in java?

To reduce the complexity and simplify the language, multiple inheritance is not supported in java.

Consider a scenario where A, B, C are three

classes. The C class inherits A and B classes. If A and B classes have same method and you call it from child class object, there will be ambiguity to call method of A or B class.

Since compile time errors are better than runtime errors, java renders compile time error if you inherit 2 classes. So whether you have same method or different, there will be compile time error now.

★ Exception Handling in Java :-

The exception handling in java is one of the powerful mechanism to handle the runtime errors so that normal flow of the application can be maintained.

Exception handling is a mechanism to ~~maintain the and~~ handle the runtime errors such as ClassNotFound, IO, SQL, Remote etc.

Exception — Exception is an abnormal condition. In java, exception is an event that distrupts the normal flow of the program. It is an object which is thrown at runtime.

Advantage of Exception Handling —
The core advantage of exception handling is to maintain the normal flow of the application. Exception normally disrupt the normal flow of the application that is why we use exception handling. Lets take an example—

```
statement 1;
statement 2;
statement 3;
statement 4;
statement 5;
statement 6;
```

Suppose there is 6 statements in your program and there occurs an exception at statement 3, rest of the code will not be executed i.e. statement 4 to 6 will not run. If we perform exception handling, rest of the exception will be executed. That is why we use exception handling in java.

* Types of Exception — There are mainly two types of exceptions - checked and unchecked where error is considered as unchecked exception. The sun microsystem says that there are 3 types of exceptions —

1. Checked exception

2. Unchecked exception
3. Error

Checked exception – The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions e.g.- IOException, SQLException etc.
Checked exceptions are checked at compile-time.

Unchecked exception – The classes that extend RuntimeException are known as unchecked exceptions e.g.– ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc.
Unchecked exceptions are checked at runtime.

Error– Error is irrecoverable e.g. – OutOfMemoryError, VirtualMachineError, AssertionError etc.

★ Java Exception Handling Keywords – There are 5 keywords used in java exception handling-
try
catch
finally
throw
throws

1. Java Try block- Java try block is used to enclose the code that might throw an exception. It must be within the method. Java try block must be followed by either catch or finally block.

Syntax of java try - catch

```
try
{
    // code that may throw exception
} catch(Exception_class_Name ref){}
```

Syntax of try- finally block

```
try {
    // code that may throw exception
} finally {}
```

Java catch block - Java catch block is used to handle the Exception. It must be used after the try block only. You can use multiple catch block with a single try.

Java finally block- Java finally block is a block that is used to execute important code such as closing, connection, stream etc.

Java finally block is always executed whether exception is handled or not.

Java finally block must be followed by try or catch block.

Java throw keyword- The java throw keyword is used to explicitly throw an exception

exception in java by throw keyword. The throw keyword is mainly used to throw custom exception. The syntax of java throw keyword is given below -

throw exception;

Ex- throw new IOException("Sorry device error");

Java throws keyword- The java throws keyword is used to declare an exception. It gives an information to the programmer that there may occur an exception.

Syntax of throws

return_type method_name() throws exception_class_name {

// method code

}

* Difference between throw and throws -

	throw	throws
1.	Java throw keyword is used to explicitly throw an exception.	Java throws keyword is used to declare an exception.
2.	Checked exception can not be propagated using throw only.	Checked exception can be propagated with throws.
3.	Throw is followed by an instance.	Throws is followed by class.

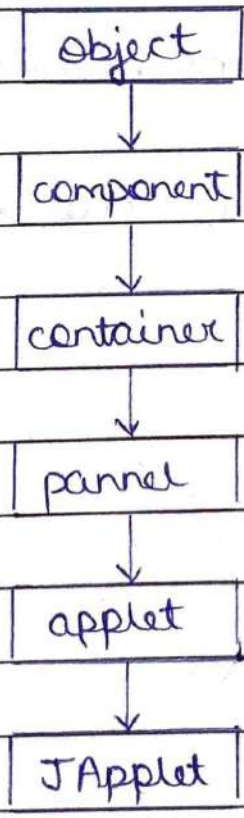
4	Throw is used within the method. You cannot throw multiple exceptions.	Throws is used with the method signature. You can declare multiple exceptions eg- public void method() throws IOException, SQLException

UNIT - 2

★ **Java applet :-** Java applet cannot be executed independently, Applet can be executed inside a java container or web browser.

Java applet is a special type of program that is embedded in the webpage to generate dynamic content. It runs inside the browser and work at client set.

Hierarchy of applet -



★ Life-cycle of applet -

1. Applet initialized [public void init();]
2. Applet started [public void start();]

3. Applet painted [public void paint();]
4. Applet stopped [public void stop();]
Applet destroyed [public void destroy();]

- Ways to run applet program – There are 2 methods for run the applet program–

1- With applet viewer
2- With web browser

With applet viewer –

```
import java.applet.Applet;
import java.awt.graphics;
public class Abc extends Applet
 {
public void paint(Graphics g)
 {
gdraw String("Hello applet" 150,150);
 }
 }
```

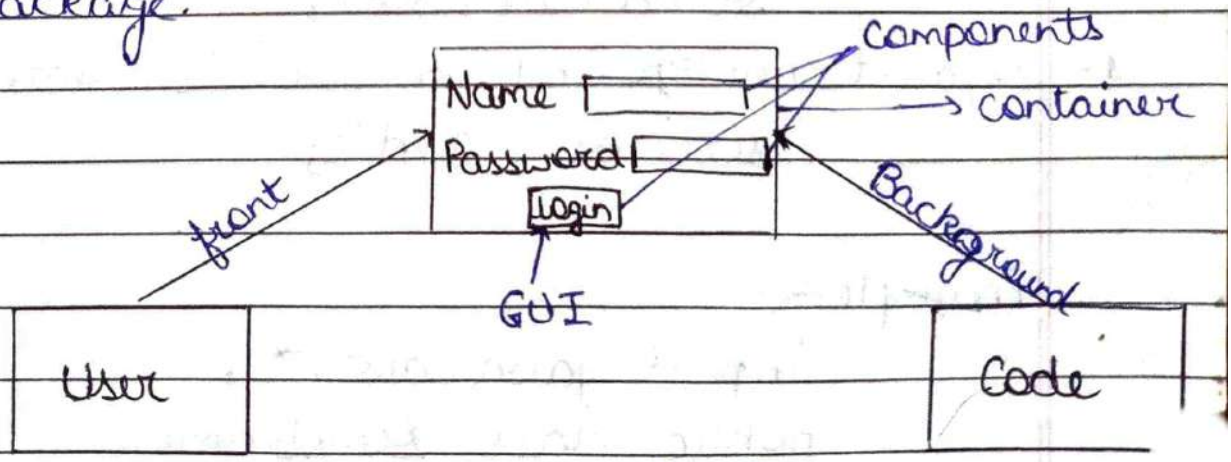
With web browser –

```
<html>
<head>
<title>
</title>
</head>
<body>
<applet code="a class">
</applet>
</body>
</html>
```

- Commonly used methods in applet –

```
public abstract void drawString();
public abstract void drawRec();
public abstract void drawOval();
public abstract void drawLine();
public abstract void drawArc();
public abstract void setColor();
public abstract void setFont();
public abstract void fillRec();
```

★ AWT :- (Abstract Window Toolkit) :- Java AWT is an API (Application Program Interface) to develop GUI (Graphical User Interface) or window based applications in Java. It is introduced in 1995. Java applet contain heavy weight component because its components are using the resources of OS and AWT components are platform dependent. Java AWT belongs in java.awt package.



(AWT)

- Creating frame in AWT — There are two methods for creating frame-

1. By creating frame object
Ex- Frame f = new Frame();

2. By extending frame class
Ex-

```
class A extends Frame
{
A()
{
code
}
}
```

- Default settings of a frame —

1- Always created frame is invisible.
setVisible(true);

2- Default frame cannot be close.
ctrl + C

3- By default the size of the frame is (0,0).
setSize(200, 300);

4- Default background colour is white.
setColor("Red");

Example-

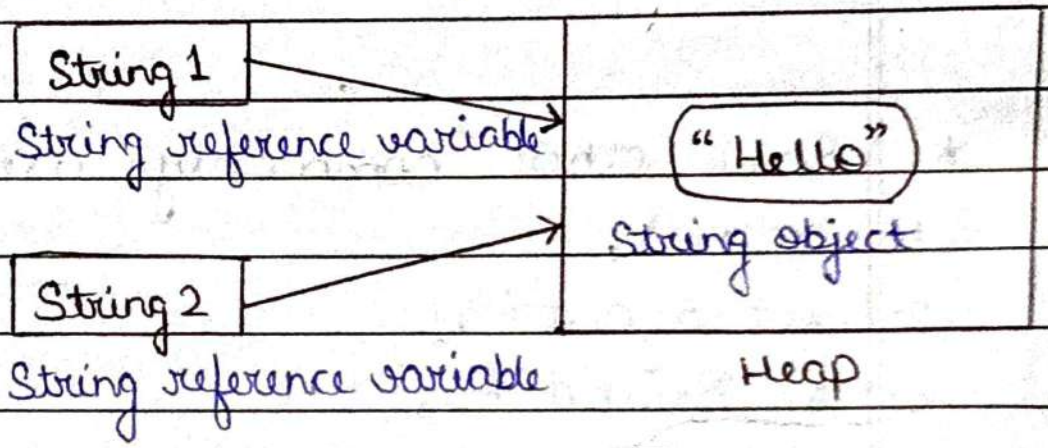
```
import java.awt.*;
public class Demoframe
{
```

```
public static void main (String args[])
{
Frame f = new Frame();
f. setVisible (true);
f. setSize (250, 350);
f. setTitle (" My frame demo ");
}
}
```

* Some other commonly used method in AWT

1- add();
2- setLayout();
3- setBounds();

★ **String Handling :-** The basic aim of string handling concept is storing the string data in the main memory (RAM), manipulating the data of the string, retrieving the part of the string etc.



★ **Main method in java :-**

In java programs, the point where the program starts its execution or simply the entry point of java programs is the main() method.

Most common syntax of main() method -

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

public static void main(String args[])

- public → Access Specifier
- static → Return type
- void → Return type
- main(String args[]) → Array of string type

1. Public — It is an Access modifier, which specifies from where and who can access the method. Making the main() method public makes it globally available. It is made public so that JVM can invoke it from outside the class as it is not present in the current class.

Example—

```
private static void main(String[] args)
```

Error : Main method not found in the class, please define the main method.

2. Static — It is a 'keyword' which is when associated with a method, makes it a class related method. The main() method is static so that JVM can invoke it without instatiating the class. This also saves the unnecessary wastage of memory which would have been used by the object declared only for calling the main() method by the JVM.

Example—

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

Error: Main method is not static in class A, please define the main method as:
public static void main(String[] args)

3. Void – It is a keyword and used to specify that a method doesn't return anything. As soon as the main() method terminates, the java program terminates too. Hence, it doesn't make any sense to return from main() method as JVM can't do anything with the return value of it.

Example –

public static int main(String[] args)

Error: Main method not found in the class, please define the main method as –
public static void main(String[] args)

4. main – It is the name of Java main method. It is the identifier that the JVM looks for as the starting point of the java program. It's not a keyword.

Example –

public static void myMain(String[] args)

Error: Main method not found in the class, please define the main method as –
public static void main(String[] args)

5. String[] args — It stores java command line arguments and is an array of type java.lang.String class. Here, the name of the String array is args but it is not fixed and user can use any name in place of it.

* Interface of AWT :- There are 3 steps of interface used in AWT.

1- UI Elements — These are the core visual elements. The user eventually sees and interact with.

2- Behaviour — These are events which occur when the user interact with UI elements.

3- Layout — They define how UI element should be organised on the screen and provide the final look and feel to the GUI.

* AWT UI Elements commonly used controls :-

1. Label — A label object is a component for placing text in the container.

2. Button — The button class is used to create a labled button that has platform independent. Some application result in some action are performed when the button clicked.

```
import java.awt.*;
public class ButtonExample
{
```

```
public static void main (String args[])
{
Frame f = new Frame("Button Example");
Button b = new Button(" Click here");
b.setBounds (50,100, 80,30);
f.add(b);
f.setSize(200, 450);
f.setVisible (true);
}
}
```

Output—

* AWT Label :- AWT The object of label is a class for use placing the text in a container. It is used to display a single line read only text. The text can be changed by an application developer but a user cannot edit its display.

```
import java.awt.*;
public class LabelExample
{
public static void main (String args[])
{
Frame f = new Frame ("Label Example")
Label l1, l2;
```

```
l1 = new Label("First table demo");
l1.setBounds(100, 200, 100, 30);
l2 = new Label("Second table demo");
l2.setBounds(40, 60, 20, 80);
f.add(l1);
f.add(l2);
f.setSize(400, 500);
f.setVisible(true);
}
}
```

* List in Java AWT — The object of list class represents a list of text item. By the help of list user can choose either one item or multiple item.

Example—

```
import java.awt.*;
public class ListExample
{
ListExample() ← constructor
{
Frame f = new Frame();
List l = new List(4);
l.setBounds(100, 100, 75, 70);
l.add("1");
```

```
l.add("l2");
l.add("l3");
l.add("l4");
f.add(l);
f.setSize(400,500);
f.setVisible(true);
}
public static void main(String args[]);
{
new ListExample();
}
}
```

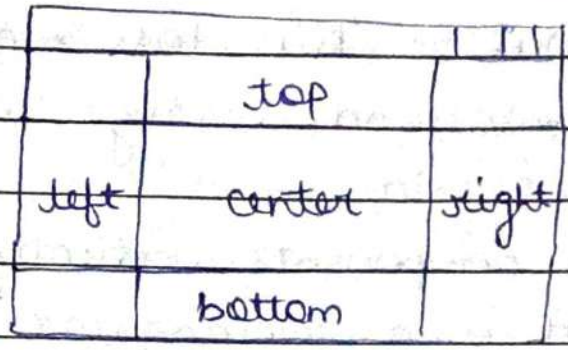
* Combo box: Choice box :- The object of choice class is used to show pop-up menu of choices. Choice selected by the user is shown on the top of the menu. At a time you can select or choice only one. Ex:

```
import java.awt.*;
public class choiceDemo
{
choiceDemo()
Frame f = new Frame();
choice c = new choice();
```

```
c.setBounds = (100,150,75,85);
c.add ("Delhi");
c.add ("Noida");
c.add ("Gurugram");
c.add ("Saharanpur");
f.add(c);
f.setSize(450,500);
f.setLayout(all);
f.setVisible(true);
}
public static void main(String args[])
{
new choiceDemo()
}
}
```

* Layout manager in Java AWT :- Java AWT provide various layout manager to position and control in a frame. The properties like - size, shape and arrangement of one layout to another layout. Some different type of layouts are discussed below -

arrange the component to fit the five region : top, bottom, left, right, center.

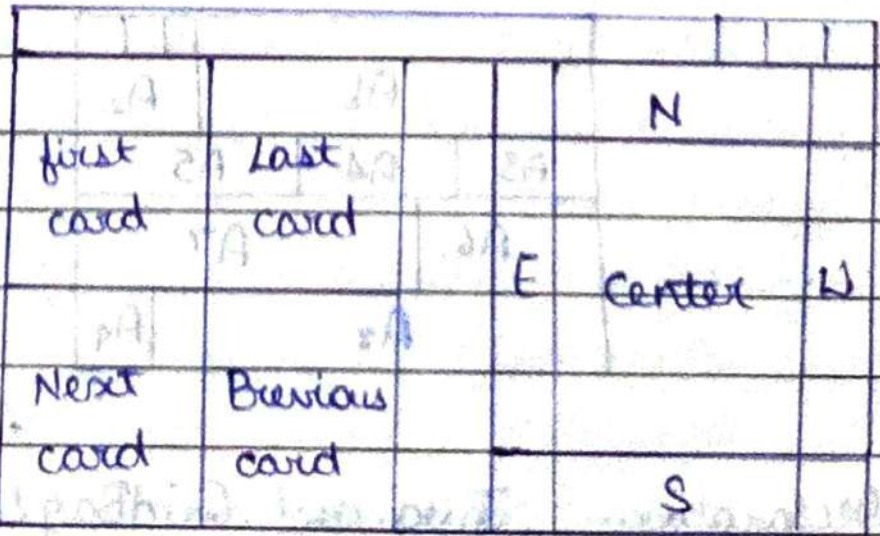


It is the default layout for all windows

Declaration — Java.awt.BorderLayout;

Classes — public class BorderLayout extends Object implements LayoutManager2, Serializable

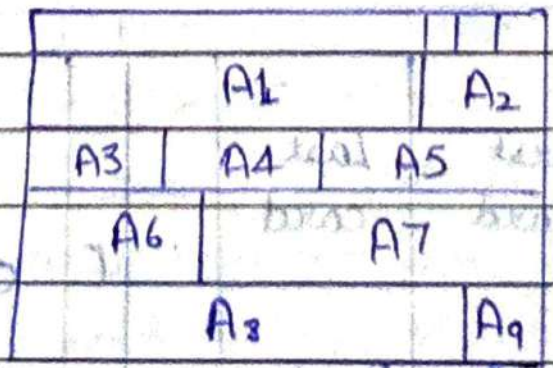
2. Card layout — This class manage two or more components that share same display space. It arrange each component in the container as a card and only one card is visible at a time.



declaration – Java.awt.CardLayout,
classes – public class CardLayout extends object
implements LayoutManager 2, Serializable

3. Grid layout – This class may all components of same size and arrange in the form of rectangular grid.
It aligns components vertically, horizontaly and formed in a rectangular grid.

4. Grid bag layout – The object of grid bag layout, vertically, horizontally along their base line without requiring components in same size.

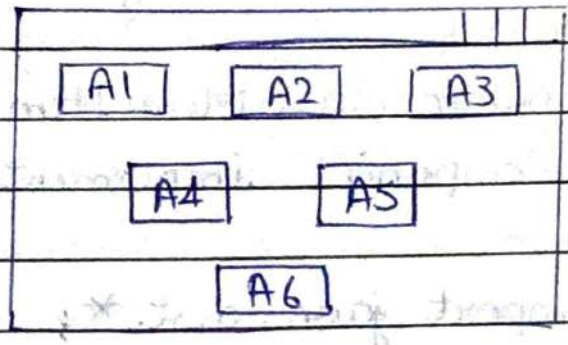


Declaration – Java.awt.GridBagLayout

class —

public class GridBagLayout extends Objects implements LayoutManager, Serializable

5- Flow layout — This class put component in a row or at the prefered size. Flow layout create left to right buttons.



declaration — Java.awt.FlowLayout

class —

★ Adapter classes in Java AWT :—

Or Event Listener / Action Listener :—

In java AWT adapter classes handle the other resources like — key, mouse, focus, window, Event perform in our program when we use the adapter classes. There are four types of adapter classes —

1- Focus adapter ;

2- Key adapter ;

3- Mouse adapter

4- Window adapter

class — Java.awt.event.*;

* **Menu item and menu** :- The object of menu item class add, labled, menu item on menu. The items used in a menu must belong to menuitem or any sub-class.
The object of menu class is pull down menu component which display in menu-bar.

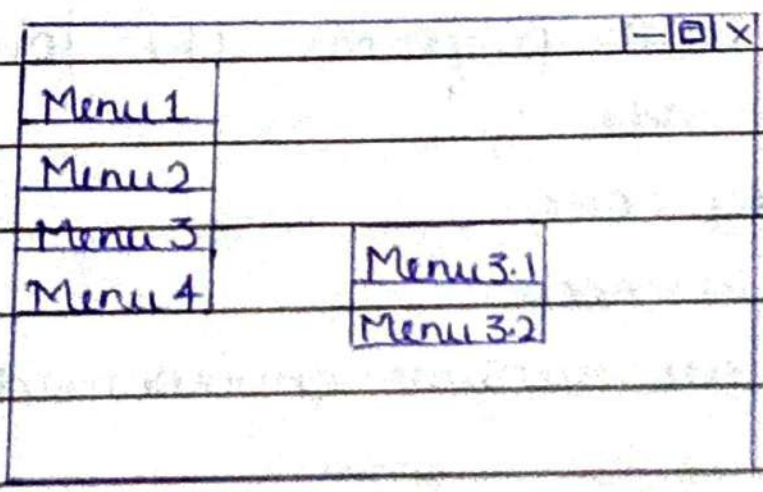
Class — public class MenuItem extends Menu Component implements Accessible

Example —

```
import java.awt.*;
public class MenuExample
{
MenuExample()
{
Frame f = new Frame("Menu demo");
MenuBar mb = new MenuBar();
Menu menu = new Menu("Menu");
Menu submenu = new Menu("Sub menu");
MenuItem i1 = new MenuItem("Menu1");
MenuItem i2 = new MenuItem("Menu2");
MenuItem i3 = new MenuItem("Menu3");
MenuItem i4 = new MenuItem("Menu4");
MenuItem i5 = new MenuItem("Menu5");
menuAdd(i1);
menu.Add(i2);
menu.Add(i3);
```

```
submenu.Add(i4);
submenu.Add(i5);
  menu.Add(submenu);
   mb.Add(menu);
f.setMenuBar(mb);
f.setSize(450,500);
f.setVisible(true);
  }
public static void main(String args[])
 {
 new MenuExample();
  }
  }
```

Output :-



* **JVM :-** JVM (Java Virtual Machine) is an abstract machine. It is called a virtual machine because it doesn't physically exist. It is a specification that provides a runtime environment in which Java bytecode can be executed. It can also run those programs which are written in other languages and compiled to Java bytecode.

JVMs are available for many hardware and software platforms. JVM, JRE and JDK are platform dependent because the configuration of each OS is different from each other. There are three notions of the JVM — specification, implementation and instance.

The JVM performs the following main tasks-

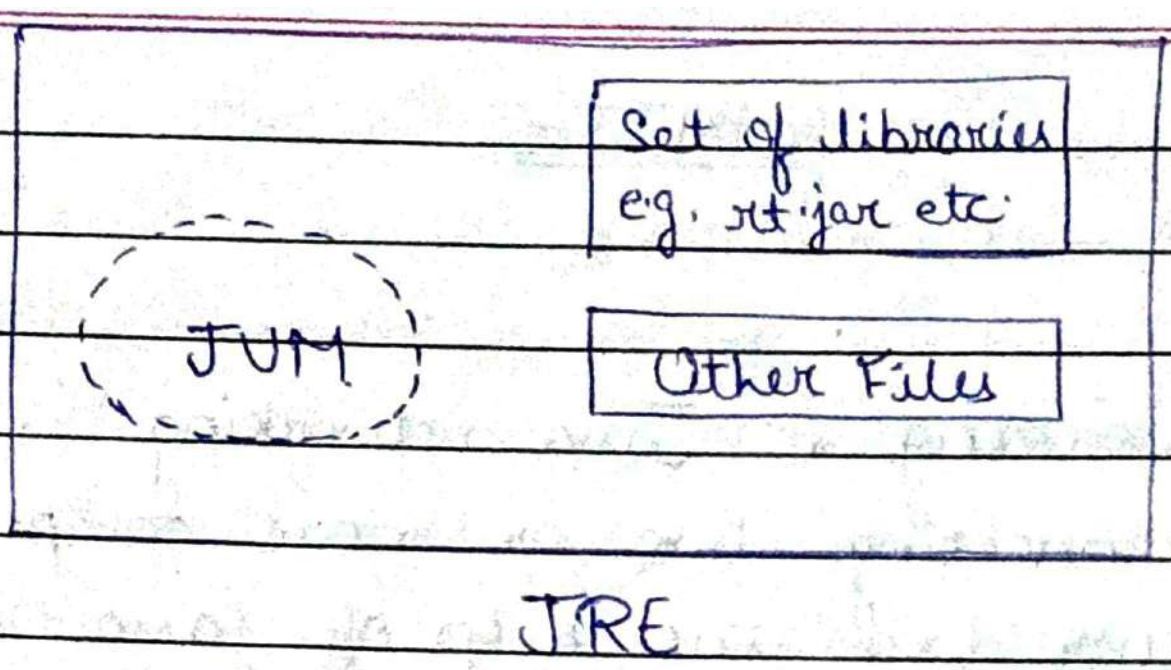
Loads code.

Verifies code.

Executes code.

Provides runtime environment.

* **JRE :-** JRE is an acronym for Java Runtime Environment. It is also written as Java RTE. The JRE is a set of software tools which are used for developing java applications. It is used to provide the runtime environ. It is the implementation of JVM. It physically exists. It contains a set of libraries and other files that JVM uses at runtime.



UNIT-5

* **Networking :-** Java networking is a concept of connecting two or more computing device together. With the help of java networking. we can share resources.

Java socket programming provide facility to share data between computing devices

Terminology — Java networking terminology The widely used java terminology are given below —

1- IP address
2- Protocol
3- Port number
4- MAC address
5- Connection oriented and connectionless protocol
6- Socket

Socket — A socket is an endpoint between two-way communication.

1- Socket programming (Connection oriented)
2- URL class
3- Displaying data on a webpage by URL connection class.
4- INET addresses classes
5- Datagram socket and datagram packet (connectionless)

Java Socket Programming – Java socket programming is used for communication between the application running on different JRE. Java socket programming can be connection oriented or connectionless.

Socket and server socket classes are used for connection oriented and datagram socket and datagram packet are used for connectionless socket programming.

The client in socket programming must know two information –

1. IP Address of server
2. Port number

* Socket class :- A socket class simplify an endpoint for communication. The socket class can be used to create socket.

	Method	Description
~~public input Stream~~	~~get Input~~	
1-	public InputStream getInputStream()	return input stream
2-	public OutputStream getOutputStream()	return output stream with attached socket
3-	public Synchronized void close()	

* Server Socket :- A server socket class can be used to create server socket. This object is used to establish communication with client site.

Method

1. public Socket Accept() → return the socket and stablish connection between server and client.
2. public Synchronized void close() → close the server socket.

Example —

file → MyServer.Java

```
import java.io.*;
import java.net.*;
public class MyServer
{
public static void main(String args[])
{
try
ServerSocket ss = new ServerSocket(5555);
Socket s = ss.accept();
DataInputStream dis = new DataInputStream
                      (s.getInputStream());
String str = (string).dis.readUTF();
System.out.println("Hello" + str);
ss.close();
} catch(Exception e)
{
System.out.println(e);
```

```
}}}
```

file → MyClient.java

```
import java.io.*;
import java.net.*;
public class MyClient
 {
public static void main (String args[])
 {
 try
Socket s = new Socket ("localhost", 5555);
DataOutputStream dout = new DataOutputStream
                        (s.getOutputStream());
dout.writeUTF ("Hello Server");
dout.close();
s.close();
 }
catch (Exception e)
 {
System.out.println (e);
 }
}
}
```

* Java Datagram Socket and Datagram Packet :-
Java datagram socket and datagram packet are used for connectionless socket programmin

★ Java Datagram Socket class:- This class represent a connectionless socket for sending and receiving datagram packets.
A datagram is basically an information of sending and receiving datagram packets but it is not guarantee of its content, arrival and arrival time.

- Commonly used constructor of datagram socket class -

1. DatagramSocket() → throws socket exception
2. DatagramSocket(int port) → throws socket exception
3. DatagramSocket(int port, InetAddress address) → throws socket exception

(1.) accept the port number on local host.
(2) Create own port number
(3) Create own port number and host address.

★ Java Datagram Packet Class :-
Java datagram packet is a message that can be send or received. If you send multiple packets it may arrive in any order.

Commonly used methods of datagram packet class -

1- DatagramPacket(byte[] barr, int length);
↳ Create packets and used to receive packets.

2- DatagramPacket(byte[] barr, int length, InetAddress address, int port)
↳ Create packets and used to receive packets.

Example -

DSender.java

```
import java.net.*;
public class DSender
{
public static void main(String args[]) throws exception
{
DatagramSocket ds = new DatagramSocket();
String str = "Welcome Java";
InetAddress ip = InetAddress.getByName("127.0.0.1");
DatagramPacket dp = new DatagramPacket
                    (str.getBytes, str.length(), ip, 400);
ds.send(dp);
ds.close();
}
}
```

DReceiverJava

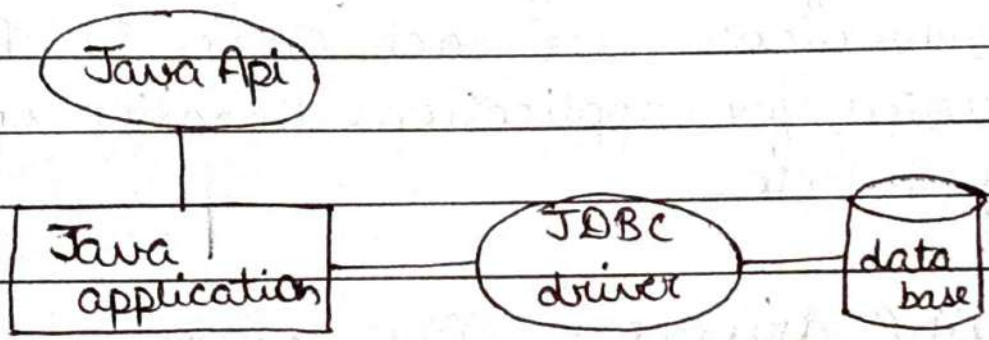
```
import java.net.*;
public class DReceiver
{
public static void main(String args[]) throws exception
{
```

```
DatagramSocket ds = new DatagramSocket(400);
byte[] buf = new byte[1024];
DatagramPacket dp = new DatagramPacket(buf,
                                       1024);
ds.receive(dp);
String str = new String(dp.getData(),0,dp.
                                  getLength());
System.out.println(str);
ds.close();
}
}
```

* JDBC (Java Database Connectivity) :- JDBC stands for Java database connectivity. JDBC is a java API used to connect query with the database. It is a part of Java SE (Standard edition). JDBC API used JDBC drivers to connect with database.

There are 4 types of JDBC

1- JDBC-ODBC Bridge driver
2- Native driver
3- Network Protocol driver
4- Thin driver



* JDBC Api Interface list -

1. Driver interface
2. Connection interface
3. Statement interface
4. PreparedStatement interface
5. CallableStatement interface
6. ResultSet interface
7. DatabaseMetaData interface
8. ResultSetMetaData interface
9. RowSet interface

* JDBC Api classes — A list of popular classes of JDBC API are given below —

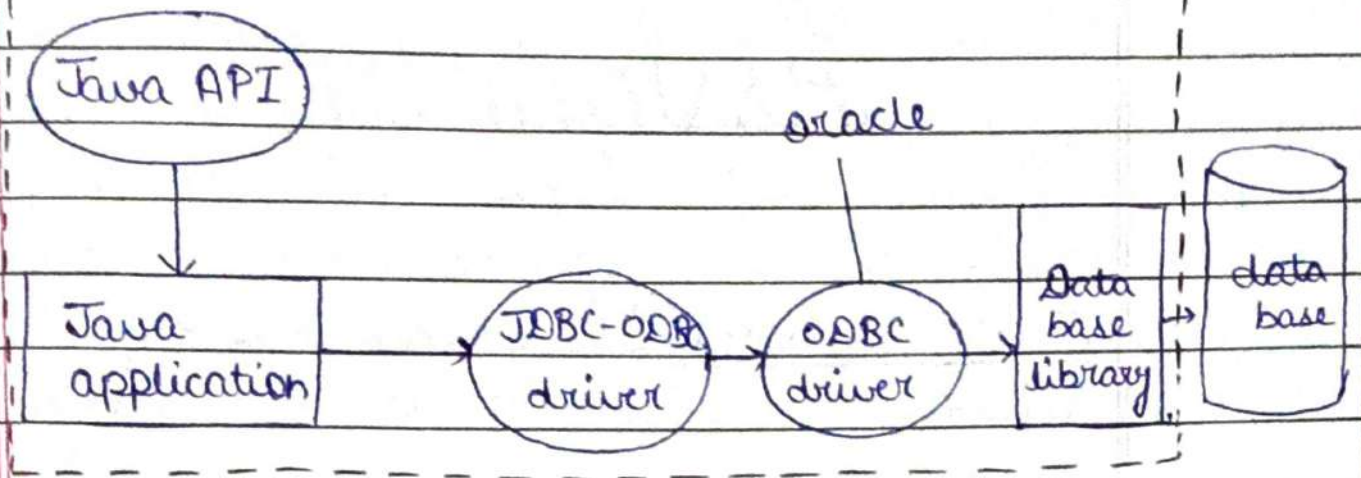
1. DriverManager class
2. Blob class
3. Clob class
4. Types class

* API (Application Programming Interface) :- API is a document that contains a description of all the features of product or software. It represent classes and interface that software programs can follow to communicate with each other. An API can created for application, libraries, operating system etc.

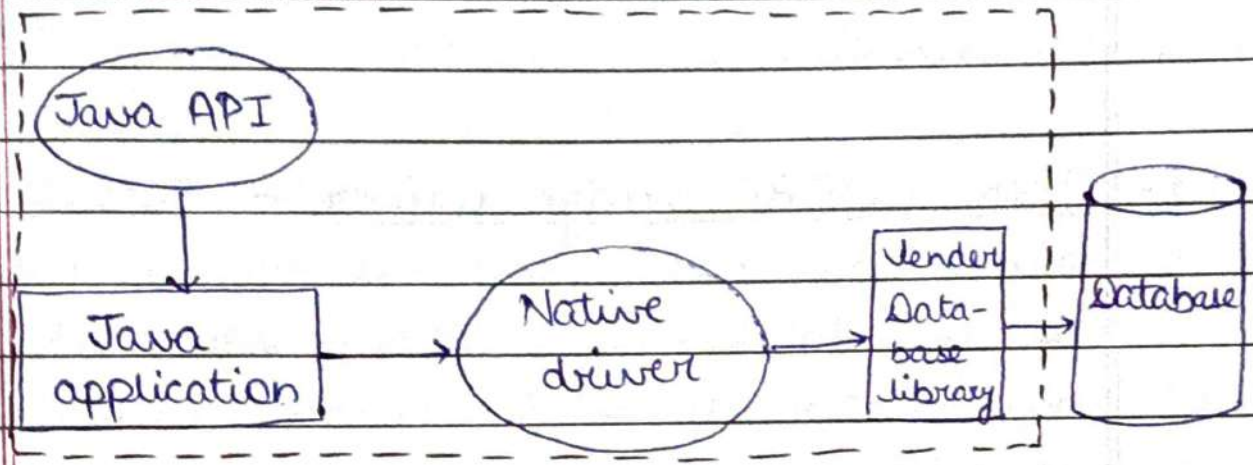
* JDBC drivers — JDBC drivers is a software component that enables java application to interact with the database. There are 4 types of JDBC drivers —

1- JDBC - ODBC bridge driver
2- Native - API driver
3- Network Protocol driver
4- Thin driver

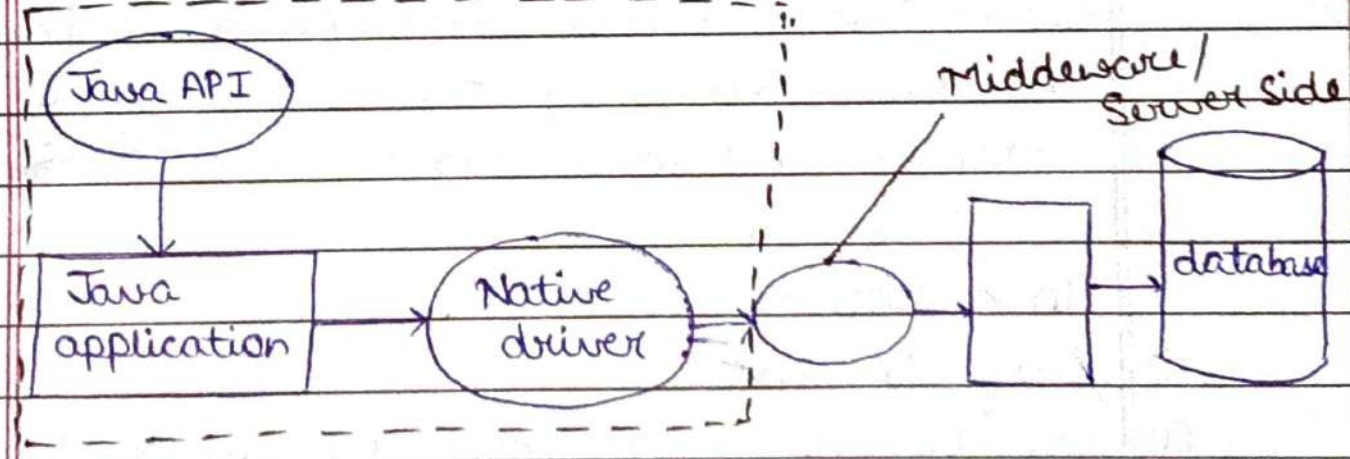
1- JDBC - ODBC bridge driver —



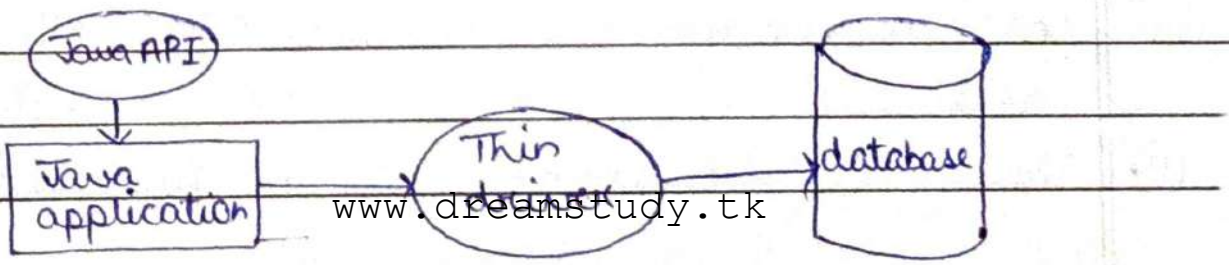
2- Native driver —



3- Network Protocol driver —



4- Thin driver —



Assignment

JDBC drivers :- There are 4 types of JDBC drivers -

1. JDBC - ODBC bridge driver
2. Native - API driver
3. Network Protocol driver
4. Thin driver

1- JDBC - ODBC bridge driver - The JDBC - ODBC bridge driver uses ODBC driver to connect to the database. The JDBC - ODBC bridge driver converts JDBC method calls into the ODBC function calls.
Oracle does not support JDBC - ODBC bridge from Java 8. Oracle recommends using JDBC drivers provided by the vendor of your database instead of the JDBC - ODBC Bridge.

Advantages -
(i) Easy to use.
(ii) Can be easily connected to any database.

Disadvantages -
(i) Performance degraded because JDBC method call is converted into the ODBC function calls.
(ii) The ODBC driver needs to be installed on the

client machine.

2- Native - API driver - The Native API driver uses the client-side libraries of the database. The driver converts JDBC method calls into native calls of the database API. It is not written entirely in java.

Advantages -

(i) Performance upgraded than JDBC-ODBC bridge driver.

Disadvantages-

(i) The native driver needs to be installed on the each client machine.

(ii) The vendor client library needs to be installed on client machine.

3- Network Protocol driver - The Network Protocol driver uses middleware that converts JDBC calls directly or indirectly into the vendor-specific database protocol. It is fully written in java.

Advantage -

(i) NO client side library is required because of application server that can perform many tasks like auditing, load balancing, logging etc.

Disadvantages -

(i) Network support is required on client machin

(ii) Requires database-specific coding to be done in the middle tier.

(iii) Maintenance of Network Protocol driver becomes costly because it requires database-specific coding to be done in the middle tier.

4- Thin driver — The thin driver converts JDBC calls directly into the vendor specific database protocol. That is why it is known as thin driver. It is fully written in Java.

Advantages —

(i) Better performance than all other drivers.

(ii) No software is required at client side or server side.

Disadvantage —

(i) Drivers depend on the Database.

* Steps for connect database by Java :-
There are 5 steps to connect to the database in java.

1- Register the driver class.
2- Create the connection object.
3- Create the statement object.
4- Execute the query.
5- Close the connection.

1- Register the driver class — The forName() method of class method of class class is used to register the driver class.
This method is used to dynamically load the driver class.
Syntax :-
public static void forName (String className) throws class not found exception

Example to register driver class—
Method—
class forName ("oracle.jdbc.driver.OracleDriver");

2- Create the connection object — The get connection method of DriverManager class is used to establish connection with the database.

UNIT - 4

HTML

* **HTML** :- HTML stand for Hyper Text Markup Language. It is used to create webpages. HTML is widely used language on the webpage. With the help of HTML you can create only static webpage. HTML uses only predefine tags. HTML is a case-non sensitive programming language.

* **Description of HTML** :-

1. **Hypertext** - Hypertext simply means "text within text". A text has a link within it is a hypertext.

2. **Markup Language** - A markup language is a programming language that is used to make text interactive and dynamic.

Example -

```
<!DOCTYPE>
<html>
<head>
<title> First webpage </title>
</head>
<h1> This is a heading </h1>
<p> This is a paragraph </p>
</body>
</html>
```

★ History of HTML :- In 1980, Tim Berners-Lee who was a contractor at CERN proposed a system for CERN researcher, in 1989 he wrote a memo proposing an internet based hypertext system. Tim Berners-Lee is known as the father of HTML. The first world wide description of HTML proposed in 1991.

★ 1. Non- empty tag -

opening-tag

<tag-Name> user define data which display on user-server </tag-Name>

closing-tag

2. Empty tag -

<tag Name>

Exe - <img>
<hr>

★ Comments in HTML :- With the help of comment tag we describe the description of tag and in a comment a particular tag information are not displayed in web-browser.

Syntax -

<!---- <p>----- </p> ----- >

★ Text styling in HTML :- There are various types of text which is used to provide style facility to a text which are display in

text are given below –

1- <P> → Paragraph
2- <b> → Bold
3- <i> → Italic
4- <u> → Underline
5- <Sup> → Superscript (a^a)
6- <Sub> → Subscript (a_a)
7- <Strike> → Cross the text
8- <del> → for deleted content
9- <big> → for big content
10- <small> → for small content

Example –

```
<!DOCTYPE>
<html>
<head>
<title> demo of comment, heading and text
        style </title>
</head>
<body>
<p> This is paragraph one <br> This is
second paragraph </p>
<!..... <P> This is comment tag </P> ....>
<h1> This is heading first </h1>
<h2>  "  "  "  second </h2>
<h3>  "  "  "  third </h3>
<h4>  "  "  "  fourth </h4>
<h5>  "  "  "  fifth </h5>
<h6>  "  "  "  sixth </h6>
```

<P> This is paragraph Bold text
Hello italic text Hello <U> under-
line text </u> hello </P>
<P> Hello ^{Hello} </P>
<P> Hello _{Hello} </P>
<Strike> ~~Hello Html~~ </Strike>
<del> This is deleted text </del>
<big> This is bigger text </big>
<small> This is small text </small>
</body>
</html>

* HTML image tag – HTML image tag is used to display image on the web page. HTML img tag is an empty tag. Closing tag are not used in image tag.

Attributes of HTML <img> tag –

1- src – It is necessary attribute to describe the source or path of the image. It instructs the browser that where are the particular image belong.

2- alt – If source is not correct then alt content will be show inside the display area.

3- width – It is used to specify width to display the image.

4. height – It specifies the height of the image.

Syntax – <img scr = "a.png" alt = "Pic" width = "100px" height = "150px" >

★ Font tag :- Font tag is used to describe the font information or styling on font.

Ex- <p style : font-family : Times New Roman, font-color : Red, font-size : 100px, text-align : Top; > Hello </p>

Attributes of font tag –

1. font-family
2. font-color
3. font-size
4. text align

★ Horizontal Rule :- The <hr> tag in HTML stands for horizontal rule and is used to insert a horizontal rule or a thematic break in HTML page to divide or seperate document sections. The <hr> tag is an empty tag and it does not require an end tag. Used to specify the alignment of the horizontal rule.

* HTML br tag — HTML
 tag or element is used to break line in a paragraph. It is generally used in poem or address where the division of line is necessary. It is an empty tag, means it does not need a company of end tag.

Example — If you want to break line
 in a paragraph,
 use the br element.

Output —

If you want to break line
in a paragraph,
use the br element.

* HTML Table :- HTML table tag is used to display data in tabular form. There can be many columns in a row. HTML tables are used to manage the layout of the page e.g. header section, navigation bar, body content etc.

HTML table tags —

Tag	Description
<table>	It defines a table.
<tr>	It defines a row in a table.
<th>	It defines a header cell in a table.
<td>	It defines a cell in a table.
<caption>	It defines the table caption.
<colgroup>	It specifies a group of one or more columns in a table.
<col>	It is used with <colgroup> element to specify column properties.

<tbody> It is used to group the body content.
<thead> It is used to group the header content.
<tfoter> It is used to group the footer content.

* HTML Anchor :- The HTML anchor tag defines a hyperlink that links one page to another page. The "href" attribute is the most important attribute of the HTML a tag.
href attribute of HTML anchor tag — The href attribute is used to define the address of the file to be linked. In other words, it points out the destination page.
Syntax :- Link Text
Ex- Click for Second Page

* Form in HTML :- HTML form is a section of document which contain controls such as text field, password field, checkbox, radio button, submit button, menus etc.
An HTML form facilitates the user to enter data that is to be sent to the server for processing.
Syntax : <form action = "URL address" Method = "get or post">
<input controls like text field, Button, etc >
</form>

• HTML form tags—

Tag name	Description
<form>	It define the form area where user input the data.
<input>	It define the input control.
<textarea>	Multiple line input control.
<label>	It define a label for input element.
<button>	It define a clickable button.

Attributes of <input> tag -

1- Text : <input type = "text" id = "text" name = "First name">

2. Password : <input type = "password" id = "pass" name = "Enter Password Here">

3. E-mail

4. Date

5. Time

6. Radio

7. Check box

Example -

```
<html>
<head><title>form Demo</title></head>
<body>
<form action = "a.html" method = "Get">
<input type = "text" id = "Name" name = "Enter your name">
<input type = "email" id = "email" name = "Enter mail-id here">
<input type = "radio" id = "Gender" name = "Male" value = "Male"> Male
```

<input type = "radio" id = "Gender" name = Female" value = "female"> Female

<input type = "checkbox" id = "BCA" name = "BCA" value = "BCA"> BCA

<input type = "checkbox" id = "B.Tech" name = "B.Tech" value = "B.Tech"> B.Tech

<input type = "Submit" id = "submit" name = "Registeration">

</form>
</body>
</html>

* List in HTML :-

1- Order list - An order list specify all the list item in a ordered way and are proceed with numbers.
SYNTAX : <ol> ———

2- Unorder list - It signifies all the list item using bullets instead of numbering system.
SYNTAX : <ul> ———

3- Definition list - These list element have a unique array of tags and elements.
<dl> ——— </dl>

List item - List item is define individual item of the list.
 ——

Example-

```
<html>
<head>
<title> List demo </title>
</head>
<body>
<h1> List of DBGI course </h1>
<ol>
<li> BCA </li>
<li> MCA </li>
<li> BBA </li>
<li> MBA </li>
<li> B.Tech </li>
<li> Poly </li>
<li> B.Pharma </li>
<li> D.Pharma </li>
</ol>
</body>
</html>
```

* Frameset :- This element holds one or more frame element at one time. It specify how many columns or rows there will be in the frame set and how much space will be occupy between frames.

SYNTAX : <frameset> ——— </frameset>

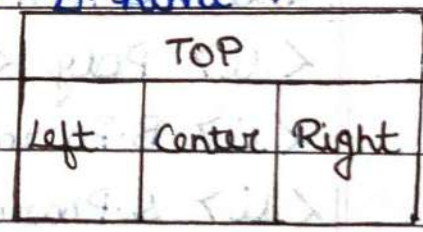
- Attributes-

1. Cols
2. Rows
3. Border
4. Frame Border
5. Frame Spacing

Example -

```
<html>
<head>
</title> frameset Demo </title>
</head>
<body>
<h1> frame demo </h1>
<frameset rows = "120">
<frame name = "top" src. = "A.html">
<frameset cols = "120,25,20,30">
<frame name = "Left" src = "B.html">
<frame name = "Center" src = "C.html">
<frame name = "Right" src = "D.html">
</frameset>
</body>
</html>
```

Output:-



* <u>< META > tag in HTML</u>:- Meta is data (information about data. The meta tag provide related information to the user individual html document. Meta tag ~~is~~ content will not be displayed on the webpage but will be working as machine parsable. These element are typically used to specify page description, keywords, author of the document, last modify and other information.

<u>SYNTAX</u>: < meta ——— >

- Attributes -

2. Content
3. Scheme
5. Author

Example –

```
<html>
<head>
<title> Meta Demo </title>
<Meta name = "Key" Contents = "HTML" >
<Meta author = "ABC" Keywords = "Hello ">
<Meta scheme = "Notify ">
</head>
<body>
<h1> Meta tag description </h1>
<p> This is paragraph of Meta tag </p>
</body>
</html>
```

★ Styling in HTML :- (text styling, CSS)

For decorating tags or text we apply CSS (Cascading Style Sheet) CSS is used to style HTML element. It is a style sheet language which is used to describe the look and formatting of a document written in mark-up language (HTML). It provide an additional feature to HTML. It mainly used with HTML to change the style of webpages and user interface. There are 3 method to apply CSS on webpage –

1. Inline
2. Internal
3. External

CSS SYNTAX :-

Selector { Property : Value , Property : Value ; }

Ex- P { Color : Red , Font-size : 10 px ; }

1- Inline CSS — Inline CSS may be used to apply a unique style or ~~order~~ a single element and individually define within a tag.

Example-

```
<html>
<head>
<title> Inline CSS Demo </title>
</head>
<body>
<h1 style = " Color : Yellow , Text-Size : 10 px ,
 text-align : center "; > This is heading </h1>
<p> This is paragraph </p>
</body>
</html>
```

2- Internal CSS — Internal CSS is used within an HTML program. It is define before the body tag instead of particular tag.

```
<html>
<head>
<title> Internal CSS Demo </title>
<Style>
h1 { color : Red; }
 P { color : Yellow; }
</Style>
```

```
</head>
<body>
<h1> This is heading </h1>
<p> This is paragraph </h2>
</body>
</html>
```

3- External CSS – External CSS not define in ~~CSS~~ HTML webpage. It is in other field which is saved with .CSS extension. External CSS link refer in <head> tag.

Ex-

```
h1{ color: Green, text-align : Center; }
P { color: yellow, font-family : Times New Roman }
```

A.CSS

```
<html>
<head>
<title> External CSS Demo </title>
<link ref = "Style Sheet" type = "text/CSS" href =
                                    "URL/A.CSS" >
```

- Methods to describe color –

1. Color name → Red
2. Hexacode → #012345
3. RGB → rgb (100, 100, 100) [0-255]

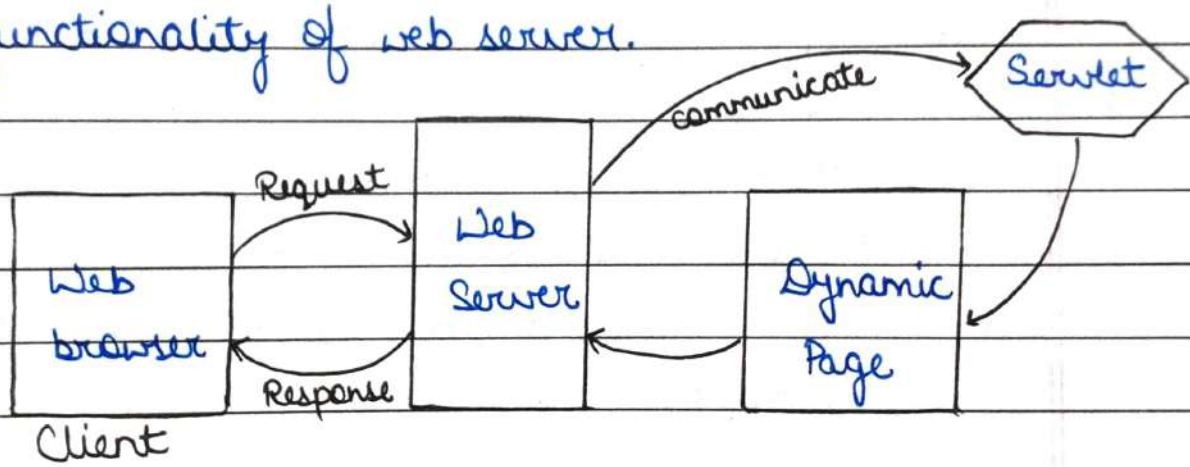
★ Special Characters in HTML :-

There are times when it becomes necessary to display symbols or special characters in HTML that are not available on a standard keyboard, such as ©. You may also need to

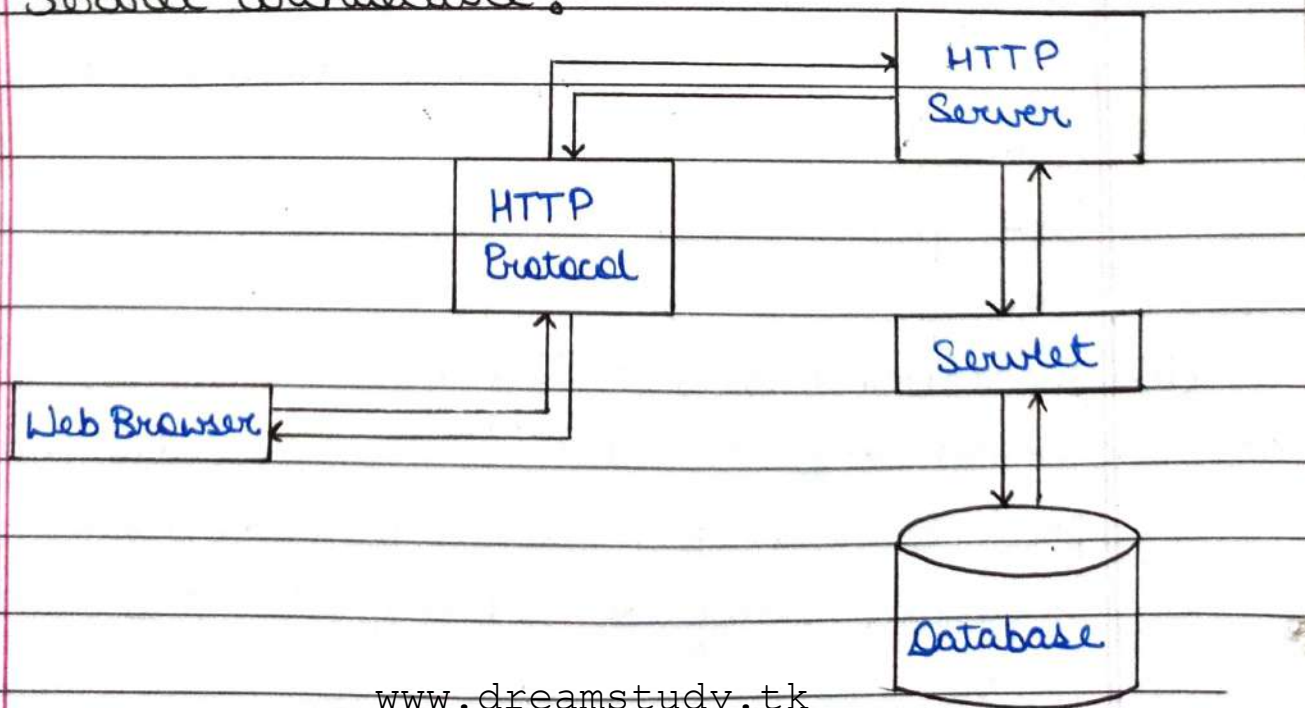
Character	Entity number	Entity name	Description
"	"	"	question mark
'	'	'	apostrophe
&	&	&	ampersand
<	<	<	less-than
>	>	>	greater-than
←	←	←	left arrow
↑	↑	↑	up arrow
→	→	→	right arrow
↓	↓	↓	down arrow
↔	↔	↔	left right arrow
™	™	™	trademark
†	†	†	dagger
‡	‡	‡	double dagger
♠	♠	♠	spade
♣	♣	♣	club
♥	♥	♥	heart
♦	♦	♦	diamond
©	©	©	copyright
®	®	®	registered trademark
¶	¶	¶	paragraph
Φ	Φ	Φ	Phi

SERVLET

* Java Servlet :- Java servlet technology is used to create web application. Web applications are helper applications that resides at web server and build dynamic webpage. Java servlet provide a component based platform independent methods for building web based applications. Servlet are java programs that can be deployed on java enabled web server to inhance and extend the functionality of web server.



* Servlet architecture :-



* Functions of Java Servlet :- These are the functions of java servlet are given below -

1- Read the explicit data which are sent by client (browser). This include HTML from on a webpage.

2- Read the ~~explic~~ implicit HTTP request data send by the client i.e. cookies, media type and comparison scheme the browser understand.

3- Process the data and generate the results.

4- Send the explicit data or document to the client (browser). This document can be send in a variety of format ie. image, excel, docx, gif etc.

5- Send the implicit http response to the client. This include telling the browser or client what type of document is being returned.
Ex- HTML, cookies setting and caching parameter or other task.

* Methods for server side :-

(a) getServerName() - This method (declared in the servlet request interface) returns the server name.
Ex- public String ServletRequest.getServerName()

(b) getServerPort() - This method (declared in the servlet request interface) returns the server port number of a particular request.
Ex- public String ServletRequest.getServerPort()

(c) getServletInfo() – This method (declared in the servlet context interface) returns the name and the version of the server which is seperated by slash (/).
Ex- public String ServletContext.getServerInfo()

(d) getAttribute*() – This method (declared in the servlet context) returns the value of named server attribute as an object.
Ex- public Object ServletContext.getAttribute()

★ Methods for client side –
(a) getRemoteHost()
(b) getRemoteAddr()
(c) getParameterValues(String Name)
(d) getParameterName()

★ Servlet container :- The servlet container is nothing but it is a package of compiler and executable program. The main function of the servlet container is to load program, initialize it and execute servlet program.

★ Web Server :- Web server interact with client through the web browser. The server deliver the web pages to the client through a web browser using HTTP protocol respectively.

★ Server side programming – ASP, .Net,

Java Servlet, JSP (Java Server Page), EJB (Enterprised Java Bean).

* Servlet Life Cycle :- A servlet life cycle can be define as the entire process from its creation till the destroy. The following steps are followed by a result.

(i) The servlet initialized by calling the init() method.
(ii) Servlet call service() method to process client request.
(iii) The servlet is terminated by destroy() method.

Web container maintain these steps -

1- Servlet class is loaded.
2- Servlet instance is created.
3- Init() method is invoked.
4- Service() is invoked.
5- Destroy() is invoked.

1) Init() - The init() method is called only once. It is called only when the servlet is created. It is not created on any user request.

SYNTAX: public void init() throws ServletException
{

}

2) Service() method - Service() is the main method to perform actual task. The servlet container call the service method to handle request which are

coming from the client side and service() is also responsible for write the formatted response to the client. Service() also check HTTP request type (GET, POST, INSERT, DELETE etc) and service method call own sub-method -

doGet()
doPost()
doPut()
doDelete()

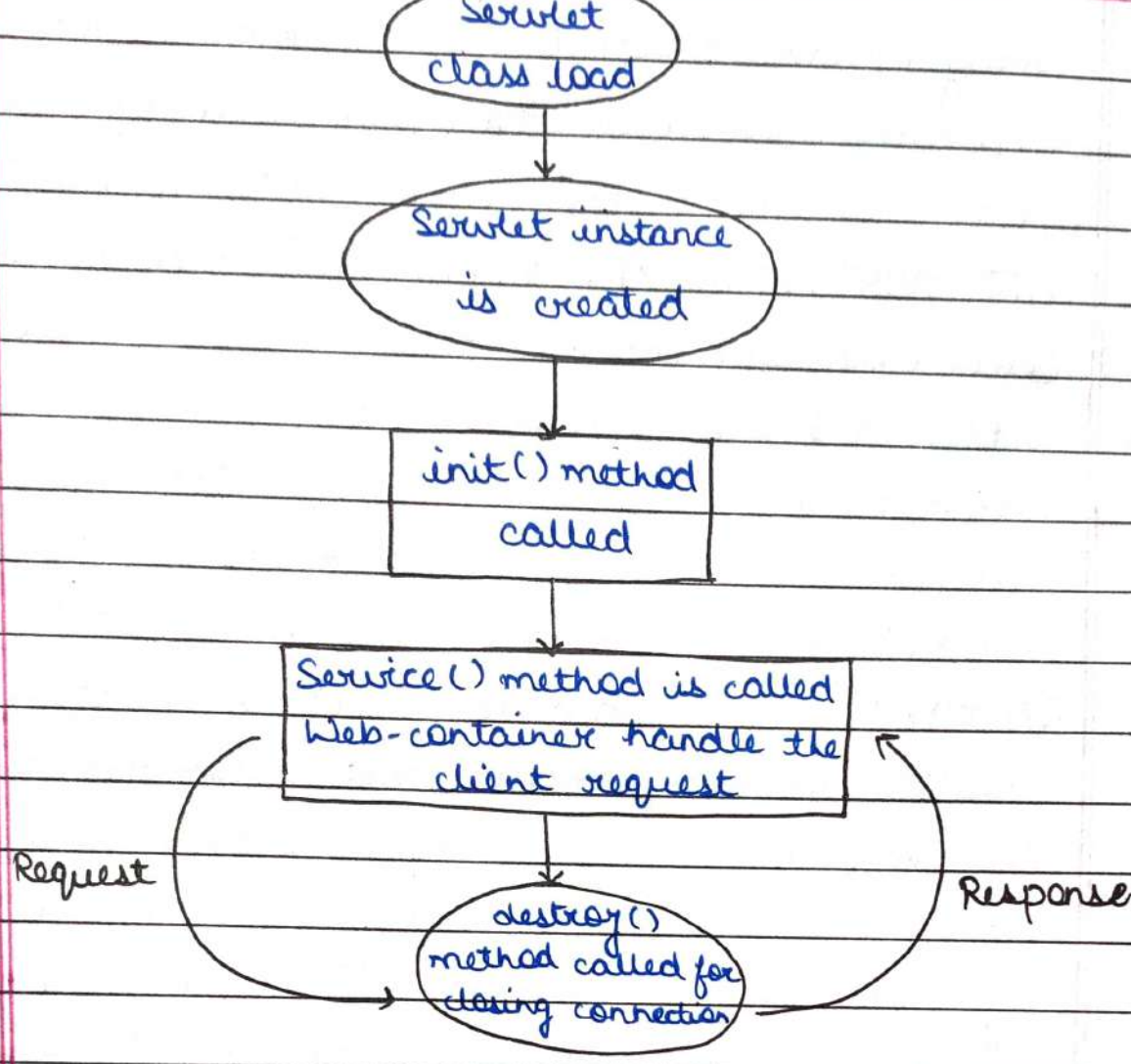
SYNTAX :- public void service(ServletRequest request, ServletResponse response) throws ServletException, IOException

```
{
----
}
```

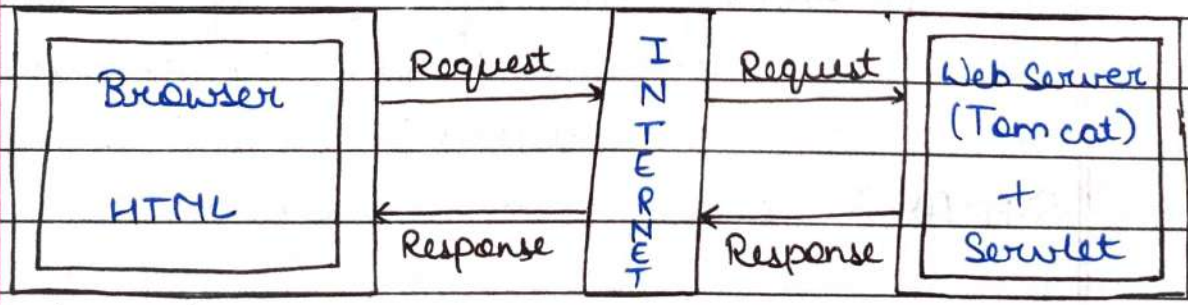
3.) Destroy() method - The destroy() method is call for ending the life cycle of servlet. This method is provide the facility to close the active database connection.

SYNTAX :-

```
public void destroy()
{
----
}
```



★ Java Servlet architecture :-



Java servlet are programs that run on the web server or server and act as a middle layer between a request coming from a web browser or other HTTP client and fetch data into database on the HTTP server. A servlet application consist of

one or more servlets. A server application is called inside a servlet container.

Description of Servlet architecture —

1- In web communication the client and server use web (internet) as an interface of communication i.e. both client and server should connect to web to communicate each other. When they are connected each other the client side is called client server and server side is called web server.

2- The standard protocol used in the which is known as HTTP (Hyper Text Transfer Protocol). When they are connected firstly client send the request and server give the response. To send a request the software is to be loaded on the web client is "browser". Similarly the software required on the web server software. Most commonly used server software are Apache Tomcat, Oracle web logic server, IPM web sphere.

* Basic Java servlet program structure (or) Sending HTML information using servlet —

```
import java.io.*;
import javaservlet.*;
import javax.servlet.http.*;
public class Abc extends HttpServlet
{
public void doGet(HttpServletRequest, request,
```

```
        HttpServletResponse response) throws
        ServletException, IOException
response.setContentType("Text/html");
printwriter out = response.getWriter();
out.print("<html>\n" +
        "<head><title>Servlet demo</title></head
                                >\n" +
        "<body>\n" +
        "<p>this is servlet paragraph</p>\n" +
        "</body>\n" +
        "</html>\n">
    }}
```

* **Cookies:-** The cookies class provide an easy way for servlet to read, create and manipulate HTTP style cookies, which allows servlet to store small amount of data (information) which are maximum use by user.
Cookies are small bit of text information that is send by web server to end user browser and that the browser return unchanged when visiting the same website.
A servlet use the getCookies() of HTTP servlet request to retrieve cookies as request. The addCookies() of HTTP servlet response send a new cookies to the browser user. You can set the age (no. of cookies) by set MaxAge().
A web server can assign a unique session Id as a cookies to each web client for sub-sequence

request from the client they can be recognised using the receiving cookies. Cookies is a key value pair of information. sent by the server to the browser. Whenever the browser send a request to the server it send cookies along with it, then the server can identify the client request using the cookies.

* Advantage of Servlet :-

1- Efficient
2- Powerful
3- Portable
4- Convenient
5- Inexpensive

UNIT – 6
JSP

★ JSP :- Java Server Page is a technology that help to website developer to create dynamically generated web-pages using the concept of HTML, XML, Servlet and other document types.
To deploy and run their java server page is communication with web-server with the help of servlet-container, such as Apache Tomcat server or Oracle web logic in JSP main functionality provide by JSTL (Java Standard Tag Library).

* Directories in JSP :- The JSP directories are message that tells the web-container how to translate a JSP page into corresponding servlet.

SYNTAX of JSP directory -

<%@ directive attribute = "value" %>

1- Page directive - Defines information of page.

<%@ Page ..... %>

2- Include directive - Defines includes file in JSP.

<%@ include ..... %>

3- Taglib directive - For creating custom tag.

<%@ taglib ..... %>

* Literals used in JSP :- Literals are the values such as number or a text string that are written literaly as part of program code.

JSP follow the all data types or literals as defined in java.

Boolean, Integer, float, character, NULL.

JSTL :- It is a collection of useful JSP tags which encapsulate core functionality common to many JSP pages web applications.

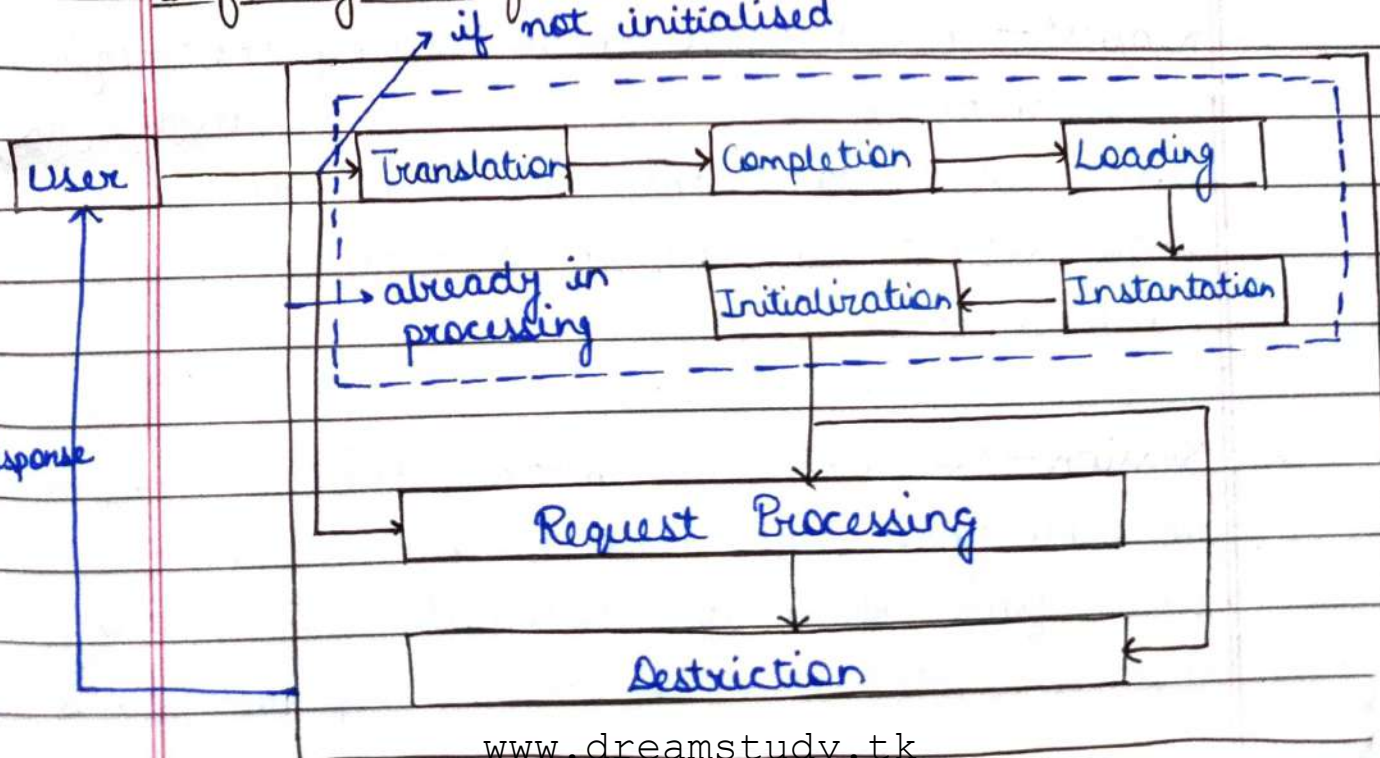
JSTL have support common structure, task such as iteration, conditional.

(a) Core tags
(b) Formatting tags
(c) SQL tags
(d) XML tags
(e) JSTL function

* Implicit object of JSP:-

	Object	Description
1.	Request →	
2.	Response →	
3.	Session →	
4.	Out →	
5	Application →	
6.	Config →	
7.	PageContext →	
8.	Page →	
9.	Exception →	

* Life cycle of JSP :-



* Operations in JSP :- JSP operations are Insert, Update, Delete, Select. Using JSP, we can do multiple operations into the database. We can insert the records and also we can delete the records which are not required. If any record needs to be edited, then we can do using an update.

* Scope of JSP objects - The availability of a JSP object for use from a particular place of the application is defined as the scope of that JSP object. Object scope in JSP is divided into four parts - page, request, session and application.

1- Page - 'page' scope means, the JSP object can be accessed only from within the same page where it was created. The default scope for JSP objects created using <jsp:useBean> tag is page.

2. Request - A JSP object created using the 'request' scope can be accessed from any pages that serves that request. More than one page can serve a single request. Implicit object request has the 'request' scope.

3. Session - 'session' scope means, the JSP object is accessible from pages that belong to the same session from where it was created. The JSP object that is created using the session scope is bound to

the session object.

4. Application– A JSP object created using the "applicatio scope can be accessed from any pages accross the application. The JSP object is bound to the application object.