1117191040
B. SUMANTH
CSE - A

Part - A

1. HTTP Servlet :-

Servlets is a Java code that run in a secure application hence it is called as the name servlets.

Servlets are most commly used for client server communication with HTTP, hence sometimes called as " HTTP Servlet".

2. Hello world using servlet:-

```
impost java.io.*;
impost java.io.servlet.*;
Public class main extends Http servlet{
    Private string message;
    Public void init() throws servlet
    exception{
        message : " Hello world ";
    }
    Public void do get (HTTP servlet request,
    HTTP servlet Response, response)
throws servlet Exception, IO Exception
response.set content Type (" text/html");
Print writes out = response.getwrites();
```

```
out.println ("<h1>" +message +"<h1>");
}
Public void destroy(){
}
}
```

③ case tags in JSP:

<c:set>
<c:remove>
<c:for-each>
<c:if>
<c:chook>
<c:when>

④ Jsp: A Jsp is a web based document that describes how to process a request to create a response.

Jsp → Java Server pages is an alternative way of creating servlets.

5. Session tracking :-

Session tracking API

Persistent cookies

URL rewriting

Hidden form field

⑥ Advantages of using PHP :-

It can generate dynamic page content It can be open, read, write open, close on the server.

PHP can collect form data.

⑦ Display text using PHP :-

"Echo" command in PHP script can be used to display texts in PHP.

If PHP allows us to display the text in various formats using various attribute methods with the help of echo command.

⑧ XML Namespace :-

Sometime we need to create two different element with the same name.

xml document allows us to create different element which are having the common name.

This is called Name space.

Eg:- <vehicles>

<car>

<price> 100000 </price>

</car>

<bike>

<price> 40,000 </price>

</bike>

</vehicles>

(9) XSLT:-

XSLT enables you to transform an xml document into another markup language. They are commonly used to transform information to HTML for display on web It can also convert information from xml into markup for useless display. for transmission to PDAs and cell phones.

(10) ATOM:- The Atom syndication format is an xml language used for web feeds, whole the Atom publishing protocol (APP) is a simple HTTP based protocol

Part-B

11. Servelets :-

Servelets are Java programs that run on web or application servers to build web pages

It acts as middle layer between request from web browsers or other HTTP client and database or applications on the server.

Servlet lifecycle :-

Servlet undergo 3 stages throughout its lifetime

Intialization - init()

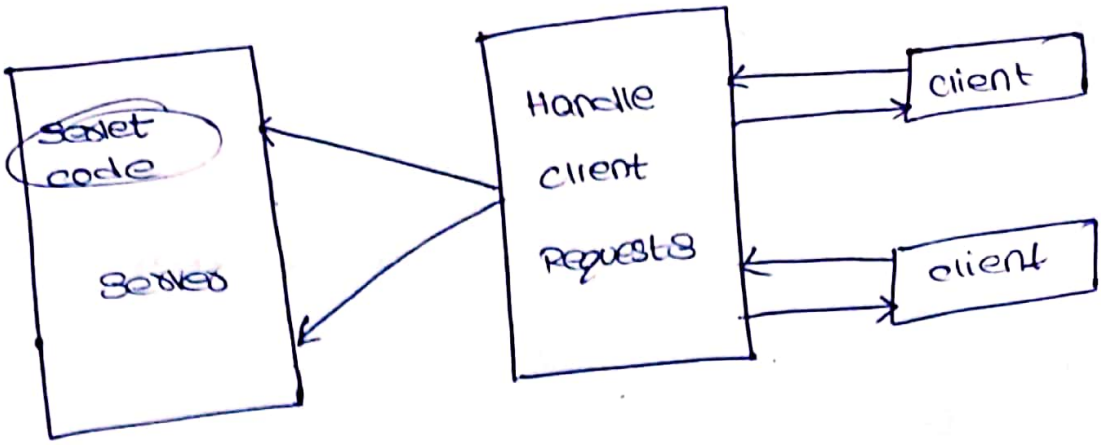
services - service()

Destruction - destroy()

Intialization :-

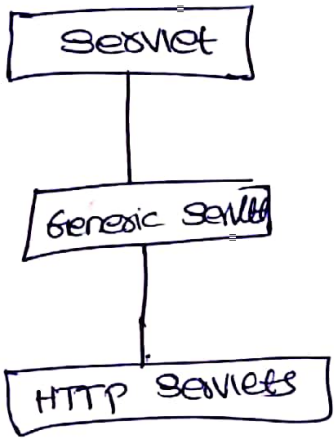
When servlet is first created, init() method is involved, so the one setup code is placed here

Service :- Each user request results in a thread that calls service() method of previously created instance. The service() method calls doGet() or doPost() method.



Destroy:- When a server decides to unload a servlet, it first calls the servlets destroy() method.

Servlet Architecture:-



The Java servlet architecture is made of 2 packages.

* javax servlet
* javax servlet http

Methods:-

void init (servlet config) - called by servlet container when the servlet is intialized.

void services:- called by servlet container to allow the servlet to respond to a request

void destroy():- called by servlet containes when servlet goes out of services

Servlet request Interface:-

Implemented by servlet containes

enable servlet obtain into about client request.

Servlet Response Interface:-

Implemented by servlet container

enables a servlet to formulate response for a client

Steps to implement and Run servlet Application:

- create HTML file.
- write servlet program
- define web.kml file.
- deploy created files in Tomcat webserver.
- Run the application

(12) JSTL :-

The JSP standard tag library represents a set of tags to simplify JSP development.

It is a collection of useful JSP tags that encapsulate the core functionality common to many JSP applications.

It provides a framework for attacking the already existing custom tags whether JSTL.

Advantages :- It has fast development
The code is Reusable
no need of script leg tag

Usestag of JSTL :-

	Prefix
core tags	c
functions tags	fn
formatter tags	fmt
XML tags	x
SQL tags	Sql

Description:-

core: provides flow control, variable support, URL managment

function: used for string length and string manipulations

formatting:- used for numbers, data, message formatting

xml:- transformation and flow control.

sql → sql support

Core tags:-

```
<html>
<head> <title> </title> </head>
<body>
    <c:set var = "welcome" scope : "session"
        value = "fixed" />
    <c:if text = ${"welcome" = "fixed"} >
<p> welcome </c. out value = "${welcome}" />
                                    </p>
    </c if >
    </body> </html>
```

c:out tags:-

<c:out value " {" Hello world"} " />

<choose tag :-

<body>

<c:set var : "color" scope . "session" value : "$blue" />

<c:choose>

<c: when text = " ${black "}"

</c:when>

<c: when text = " ${'blue'}" >

</c : when> <c: otherwise>

<c: otherwise>

</body>

c: for each tag:-

<c: for each var : " Iterator " begin = "100" and : "110" >

count : <c : out value = '${iterator}' /> >

</c: for each>

count = 100

count = 101

count = 110

(13)

b: Student management system:-

login html:-

```
<html>
<head> <title> student management </title>
</head>
<body>
<h3> STUDENT MANAGEMENT DATABASE </h3>
<form method = "get" action "signup.jsp" >
username: <input type = "text" name = "u-name"
value = " " >
email : <input type = "vaschex" name = "name"
value = " " >
name: <input type = "text" name = "name"
vale = " " >
last name: <input type = "text" name = "l-name"
value = " " >
password: <input type = "password" name = "pass"
value = " " >
confirm password: <input type = "confirm password"
name = " c.pass" value = " " >
```

```
<button type: "submit"> signup </button>
</form>
< form method : "get" action : "accept uses. jsp">
uses name : < input type = "text" name = "u.name"
value = " " >
Password < input type = "password" name = "pass"
value = " " >
<button type = "submit > login </button>
</form>
</body>
</html>
```

Signup. jsp :-

```
< %. @ page impost = "java.sql . * " % >
< %. @ page impost = "java.util . * % >
< %. /
    connection con:
    preposed statement ps1, ps2, ps3, ps4, ps5,
    " ps6 ;
public void jsp Init ( )
{ try
  {
```

```
con = DriverManager.getConnection("abc:http://localhost
    :1521/ip lab:", "root");

ps1.con.prepare statement ( update "from users");

ps2:con.prepare statement ( insert 2 mail to users bs
                            username:);

  }
catch(exception ex);
  {
    ev.printstack trace();
  { }
 % }
<%!
   public void jsp destroy()
     {
       try
       { ps1.close(); ps2:close(); ps3.close();
         ps4.close(); ps5.close();
       }
 %>
```

(14) XML parsers :-

(16) An XML parser is a software library or package that provides interface for client applications to work with an XML document.

XML parser is designed to read the XML and create a way for program to use XML.

There are two main types.

- DOM
- SAX

DOM :- Document object model :-

A DOM document is an object that contains all the information of an XML document.

It is composed true structure.

The DOM parser implements a DOM API and it is very simple to use.

Features :- A DOM parser creates an internet structure in memory which is a DOM and client application gets informations of original XML document by invoking methods on this document object.

DOM parser has tree based structure

Advantage :- Supports both read and write operations

API is simple to use.

preferred when random access to widely seperated parts of a document is required.

Disadvantages :- It is memory inefficient

It is slower than others

SAX :- A sax parser implements SAX API then API is an event based API and less initiative

features :- It does not create any internal structure clients do not know what methods to call, they just overide methods of API and place thier own code inside the method.

Advantages :-

simple and memory efficient

fast and for huge documents

disadvantages :- event based so its API is less initiative

- clients never know fault information since data is broken into process.

(15) b. XSLT:-

XSLT is a language for transforming XML documents into XHTML documents to other XML documents

XSLT stands for XSL Transformation.

XSLT is the most important part of XSL.

XSLT transforms an XML document into another XML document.

XSLT is used to transform an XML document into another XML document or another type document that is recognized by a browser like HTML and XHTML normally. XSLT does this transformation each XML element into an (X) HTML element.

With XSLT you can add/remove elements and attributes to or from the output fake.

you can also rearrange and sort elements perform tasks and make decisions about which elements to hide and display and a lot more.

XSLT transforms an XML source tree into an XML result tree.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
<xsl:template match="/">
```

```
<html>
<body>
<h2> my CD collection </h2>
<table border = "1">
    <tr bg color = "black">
    <th> title </th>
    <th> artest </th>
</tr>
<xsl - for- each select = " catalog /cd">
    <tr>
        <td> <xsl value of select = "title /> </td>
        <td> <xsl value of select = "artist" /> </td>
    </tr>
</xsl- for- each>
</table>
</body>
</html>
</xsl : template>
</xsl : stylesheet>
```

16. b

Part-C

Student management system:-

login.html

```
<html>
<head> <title> exam portal </title>
<meta charset = "UTF-8">
<meta name: " viewpoint " content: "width=1.0">
</head>
<body>
<div style = " text-align: center">
<h3> ONLINE EXAM PORTAL - login </h3>
<h4>
<form method = "get" action "accept user.jsp">
Username <input type = "text" name: "uname"
value = " ">
Password <input type = " password " value = " ">
<button type = "submit"> login </button>
</form>
</div>
</body>
</html>
```

accept usex.jsp:-

```
<%@ page import = "java.sql." %>
<%@ page import = "java.util" %>
<%!
    connection con;
    psepased statement PS1, PS2;
    Public void jSPInit()
    {
      try
        {
    class.for.name ("idbc. client drivex").new
    instance();
con : drivex manager. get connection (" idbc:http://
      local host: 1527 // iplab1', "xoot", "xoot");
PS1 : con. psepase statement ("select count(*) from
    usexs where username and password = ?");
PS2 : con. psepasestatement ("select" from usexs");
  }
  catch (expection ex)
    {
      ex. Pount stackTsace();
    }
```

```
}
%>
}
<%! Public void jsp destroy()
{ try
{
ps1.close()
ps2.close();
con.close();
}
catch (Exception ex)
{
ev.printStackTrace();
}
}
%>
```