Q.1 What is Java Servlet? (2 marks)
A servlet is a java class / a program that is a server side component and runs in a web container. The servlet generates the dynamic content. The content is dependent on the client request. It is used to extend the capabilities of servers that host applications accessed by means of a request-response programming model. They can respond to any type of request which are commonly used to extend the applications hosted by web servers. For such type of applications, Java Servlet technology defines HTTP-specific servlet classes. The javax.servlet and javax.servlet.http packages provide interfaces and classes for writing servlets. All servlets implement the Servlet interface, which define the different life-cycle methods. When implementing a generic service, you can use or extend the GenericServlet class provided with the Java Servlet API. The HttpServlet class provides methods, such as doGet and doPost, for handling HTTP-specific services.

Q.2 Describe the main purpose of servlets. (2/4 marks)
A java enabled server’s functionality can be extended by a servlets. Usually a servlet is used to develop web applications in a web server. The servlets are used to create web pages which are called dynamic web pages which mean the content of a web page can change according to the input sent from the web client. Servlets are server independent and platform independent.

Q.3 Describe the basics of Servlets. (2/4 marks)
- Servlets are java classes which run on a web server.
- The results produced by the servlet are viewed on a remote web server.
- Servlet is a server side component in web applications.
- The servlet performs the request / response paradigm using the web container.
- Servlets is the best alternative for CGI.
- HTTP servlets is advanced and mostly used on current web applications.
- These servlets responds to HTTP protocol requests that are being sent from a web server and returns web pages.
- Servlets container creates only one instance for each servlets.
- The requests are handled by a separate thread.
- Each thread will invoke doGet or doPost which in turn invokes the service() method where the actual servlet’s operations are authored.
What are the important items in javax.servlets.

Q.4 Defines all the methods that a servlets implements.
Servlets receives and responds to a request from the web clients. This interface has the methods that are to initialize a servlets, service a request and removal of a servlets from the server.
ServletConfig :
- Servlets configuration is used to send information to a servlets at the time of initializing the servlets.
- This process is handled by the web container.

**ServletRequest :**
- It creates an object that provides the client request to the servlet.
- The servlets container creates the object and sends it to service() method.

**ServletResponse :**
- It creates an object that provides the response to the client by the servlets.
- The servlets container creates the object and sends it to the service() method.

**Q.5 Explain the method of Servlet interface.**

1. **init()**
   - This method initializes the servlet.
   - It invokes only once and automatically invoked by the servlet engine / container.
   - This method is used and guarantees the actions that are to be performed before the service() method is invoked.
   - This is an overloaded method with no parameter and one ServletConfig object.
   - In case of a fatal error during initialization, the init() method throws UnavailableException object.

2. **service() :**
   - To carry a single request sent by a client, this method is used.
   - This method implements / performs the paradigm of request and response.
   - The request object contains the information of the service request including the client sent parameters.
   - The response object is used to send the response to the client.
   - The quality of handling these two methods rely on the underlying network environment.

3. **destroy() :**
   - Invoked just before the stopping the operations by the servlet.
   - It performs the resources cleaning process such memory, files, threads removal and ensures the persistence state is synchronized by the servlet’s current in-memory state.
   - The destroy() method invoked only once before the unloading the servlet.

4. **getServletConfig() :**
   - This method is used to return the servlet config object.
   - This object contains the initialization parameters and startup configuration for this servlet.
   - This method returns the ServletConfig object to init() method and the init() method will store this object in order to return ServletConfig object if needed.
5. `getServletInfo()`:
- This method returns a string that contains the information of the servlet such as version, copyright, author.
- This information is used as administrative tool and be displayed by the servlet engine.
- It returns this information in plain string form, as the information is servlet engine specific.

Q.6 Explain Life Cycle of a Servlets:
The web container maintains the life cycle of a servlet instance. Let's see the life cycle of the servlet:

1. Servlet class is loaded.
2. Servlet instance is created.
3. `init` method is invoked.
4. `service` method is invoked.
5. `destroy` method is invoked.
6. Finally garbage collected

Draw any One diagram of Servlet life cycle

As displayed in the above diagram, there are three states of a servlet: `init()`, `Service()`, `destroy()`

1) **Servlet class is loaded**
The class loader is responsible to load the servlet class. The servlet class is loaded when the first request for the servlet is received by the web container. There is request for JVM, if it does not exist. A servlet is only loaded once

2) **Servlet instance is created**
The web container creates the instance of a servlet after loading the servlet class. The servlet instance is created only once in the servlet life cycle.
3) **init() method is invoked**

The web container calls the init method only once after creating the servlet instance. The init method is used to initialize the servlet. It is the life cycle method of the javax.servlet.Servlet interface. Syntax of the init method is given below:

```java
public void init(ServletConfig config) throws ServletException
```

During initialization, the servlet has access to two objects:
1. ServletConfig  
2. ServletContext

The following are the most common tasks that are implemented in the these method:
- Reading initialization parameters
- Reading configuration data from persistent resource like a config file
- Initializing a database driver
- Writing of information to a network resource

4) **service() method is invoked**

The web container calls the service method each time when request for the servlet is received. If servlet is not initialized, it follows the first three steps as described above then calls the service method. If servlet is initialized, it calls the service method. Notice that servlet is initialized only once. The syntax of the service method of the Servlet interface is given below:

```java
public void service(ServletRequest request, ServletResponse response) throws ServletException, IOException
```

5) **destroy() method is invoked**

The web container calls the destroy method before removing the servlet instance from the service. All resources allocated by init() it should released by destroy. It gives the servlet an opportunity to clean up any resource for example memory, thread etc. This is calls only once in the lifetime cycle of servlet. The syntax of the destroy method of the Servlet interface is given below:

```java
public void destroy()
```

### Q.7 Explain Architecture Diagram of Servlet

The following figure depicts a typical servlet life-cycle scenario.

- First the HTTP requests coming to the server are delegated to the servlet container.
- The servlet container loads the servlet before invoking the service() method.
- Then the servlet container handles multiple requests by spawning multiple threads, each thread executing the service() method of a single instance of the servlet.
Servlets are Java classes which service HTTP requests and implement the `javax.servlet.Servlet` interface. Web application developers typically write servlets that extend `javax.servlet.http.HttpServlet`, an abstract class that implements the Servlet interface and is specially designed to handle HTTP requests.

Following is the sample source code structure of a servlet example to show Hello World —

```java
// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class HelloWorld extends HttpServlet {

    private String message;

    public void init() throws ServletException {
        // Do required initialization
        message = "Hello World";
    }

    public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // Set response content type
        response.setContentType("text/html");

        // Actual logic goes here.
        PrintWriter out = response.getWriter();
        out.println("<h1>" + message + "</h1>");
    }

    public void destroy() {
        //do nothing
    }
}
```
Q.8 Differentiate between doGet and doPost

<table>
<thead>
<tr>
<th>doGet</th>
<th>doPost</th>
</tr>
</thead>
<tbody>
<tr>
<td>In doGet Method the parameters are appended to the URL and sent along with header information</td>
<td>In doPost, parameters are sent in separate line in the body</td>
</tr>
<tr>
<td>Maximum size of data that can be sent using doGet is 240 bytes</td>
<td>There is no maximum size for data</td>
</tr>
<tr>
<td>Parameters are not encrypted</td>
<td>Parameters are encrypted</td>
</tr>
<tr>
<td>DoGet method generally is used to query or to get some information from the server</td>
<td>Dopost is generally used to update or post some information to the server</td>
</tr>
<tr>
<td>DoGet is faster if we set the response content length since the same connection is used. Thus increasing the performance</td>
<td>DoPost is slower compared to doGet since doPost does not write the content length</td>
</tr>
<tr>
<td>DoGet should be idempotent. i.e. doGet should be able to be repeated safely many times</td>
<td>This method does not need to be idempotent. Operations requested through POST can have side effects for which the user can be held accountable, for example, updating stored data or buying items online.</td>
</tr>
<tr>
<td>DoGet should be safe without any side effects for which user is held responsible</td>
<td>This method does not need to be either safe</td>
</tr>
</tbody>
</table>

Q.9 Why is a constructor needed in a servlet even if we use the init method? (2 marks)
- Although the init method of the servlet initializes it, a constructor instantiates it.
- A developer might never explicitly call the servlet's constructor but a container uses it to create an instance of the servlet.

Q.10 What is GenericServlet class? (2 marks)
- GenericServlet is an abstract class which implements the Servlet interface and the ServletConfig interface.
- Other than the methods included in above two interfaces, it also provides simple versions of the lifecycle methods init and destroy, and implements the log method declared in the ServletContext interface.
- Since this class is not specific to any protocol, it is known as generic servlet.

Q.10 How can the session in Servlet be destroyed? (2 marks)
There are two ways to destroy a session:
1. Programatically : By using session.invalidate() method. It makes the container abandon the session on which the method is called.
2. When the server shuts down.

Q.11 What are the types of Session Tracking?
Following are the popular ways of session tracking:
1. URL rewriting: In this method of session tracking, some extra data is appended at the end of
the URL, which identifies the session. This method is used for those browsers which do not support cookies or when the cookies are disabled by the user.

2. Hidden Form Fields: This method is similar to URL rewriting. New hidden fields are embedded by the server in every dynamically generated form page for the client. When the form is submitted to the server the hidden fields identify the client.

3. Cookies: Cookie refers to the small amount of information sent by a servlet to a Web browser. Browser saves this information and sends it back to the server when requested next. Its value helps in uniquely identifying a client.

4. Secure Socket Layer (SSL) Sessions

Q.12 What is session? (2 marks)
- The session may be said as an object.
- It is used by a servlet to track a user’s interaction.
- It interacts with the Web application.
- It works across multiple HTTP requests.
- The sessions are mainly stored in the server.

Q.13 Define the servlet mapping. (2 marks)
- The servlet mapping is defined as an association between the URL pattern and a servlet.
- The mapping is used in mapping requests.
- It maps in the Servlets only.

Q.14 Explain the servlet context. (2 marks)
- The servlet context is coined as an object.
- It is contained information about the Web application and the container.
- With a context, a servlet can be used for logging events, to obtain URL references to resources, and for setting and storing attributes for the other servlets within the context.

Q.15 Explain servlet. (2 marks)
- A servlet is simply a java program.
- It runs for an action.
- It runs inside a web container.
- The servlet is used in various implementation project.

Q.16 what is the procedure for initializing a servlet? (2 marks)
- To initialize a servlet init() is used.
- init() initializes a java program.
- A constructor can also be used to initialize a servlet.

Q.17 What is the requirement of servlet config and servlet context implemented and how are they implemented?
The servlet config is used for initialization of a servlet. Using servlet config initialization parameters passed for the servlet by making use of web xml.

```xml
<servlet>
  <servlet-name> ServletName </servlet-name>
  <servlet-class> JavaClassName </servlet-class>
</servlet>
```
Q.18 What are the objects involved when a servlet receives a call from client?
Servlets are designed to receive request or calls from clients. When a servlet receives a call from a client there are two objects involved in the process. The objects involved the request received from the client and the response of servlet to the client.

The 2 objects involved when a servlet receives a call from a client are:
1. Servlet Request:
The servlet request represents the request made by the client. It encapsulates communication from client side to server side. The servlet request checks the request and finds out which servlet shall respond to the request made by client.
2. Servlet Response:
The servlet response represents the response made by the servlet to the client. It assists the servlet in sending a response to the client.

Q.19 What is HTTP servlet? Explain with the help of an example.
HTTP servlet is an extension of the Generic servlets. It provides various methods such as doGet(), doPost() etc. Which are useful in determining the type of request being made and thus picking the right method

Example:
HttpServletRequest:
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
public class Demo extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException
    {
        resp.setContentType("text/html");
        PrintWriter out = resp.getWriter();
        out.println("<HTML>\n        <HEAD><TITLE>Title</TITLE></HEAD>\n        <BODY><H1>
        Demo Servlet !</H1><H6>Again.</H6></BODY></HTML>");
    } 

Web.xml
<servlet>
    <servlet-name>Demo</servlet-name>
    <servlet-class> Demo </servlet-class>
</servlet>
<servlet-mapping>
Q.20 Why are HTTP servlets used in programming? (2 marks)
HTTP servlets define the servlets for the web to access the services using the HTTP protocol. The servlet interface in this is having a class that consists of the centralized abstraction using the Java Servlet API. The servlet class defines the methods that can be used for the communication between the clients. HTTP is used as a simple and stateless protocol that allows the client to make the request and server responds to it when the transaction is done. When a request is received from the client possibly a web browser than using the HTTP command a method will be called that allow the servers to take certain types of action on the particular request. The URL of the document represents the version of the HTTP protocol that is being used in the program.

Q.21 What are the ways to handle multi-threading in Servlets? (2 marks)
Multi-threading is used with the processes to divide the tasks into multiple threads and share the resources between them. It increases the speed and the performance of the system and the program that is in execution. When there are requests coming at the same time to the servlet, then the server will handle the requests coming from each client by creating a new thread and inserting the request in it. The model that is used consists of single thread per client/server request. This allows the thread to be reused again in the case when the same client is trying to make again the request. This multi-threading environment decreases the time delay using the API and decreases the delay in between the communication of the server and the client used for the requests.

Q.22 What is the process to implement doGet and doPost methods?
- **doGet():**
doGet() methods are the service methods that is included in the servlets. This allow the servlet to handle the GET request that is being processed by the client. This overrides the method to support the GET request that automatically supports the HTTP head request that is given in the no body in the response and included in the header fields. It is being implemented as:

```java
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, java.io.IOException
```

It is used to read the request data and provide the response of it by putting the solution in the header. It uses the response’s writer or the object stream object that provides the response to the client. This is safe to use and can be safely repeated in case of any other request.

- **doPost():**
doPost() is used to handle a POST request that is also given at the time of filling up the form or any other action that is related to the user submission. This method allows the client to send the data of any length to the web server and that is also at single time. To read the request data the response headers are included that takes the response writer class to write that uses the output stream object. It uses the function of PrintWriter object that returns the response to set the content type of accessing the object. It is given as:
**Q.23 What is Servlets and explain the advantages of Servlet life cycle?**
- Servlets are the programs that run under web server environment.
- A copy of servlet class can handle numerous request threads.
- In servlets, JVM stays running and handles each request using a light weight thread.
- Servlets life cycle involve three important methods are:
  1. Init() this is called when servlet first loaded into the web server memory.
  2. Service() Once initialized, servlets stays in memory to process requests. Servlets read the data provided in the request in the service() method.
  3. Destroy() When server unloads servlets, destroy() method is called to clean up resources the servlet is consuming.

**Q.24 Explain the generic servlet and HTTP servlet. [4 marks]**

**Generic Servlet:**
GenericServlet class is direct sub class of Servlet interface. Generic Servlet is protocol independent. It handles all types of protocol like http, smtp, ftp etc. Generic Servlet only supports service() method. It handles only simple request.

```java
public void service(ServletRequest req, ServletResponse res)
```

A generic servlet should override its service() method to handle requests as appropriate for the servlet. The service() method accepts two parameters: a request object and a response object. The request object tells the servlet about the request, while the response object is used to return response. Generic Servlet only supports service() method.

**HttpServlet:**
HttpServlet class is the direct subclass of Generic Servlet. HttpServlet is protocol dependent. It handles only http protocol. HttpServlet supports

```java
public void service(ServletRequest req, ServletResponse res )
```

```java
protected void service(HttpServletRequest req, HttpServletResponse res)
```

HttpServlet supports also doGet(), doPost(), doPut(), doDelete() etc. An HTTP servlet usually does not override the service() method. Instead, it overrides doGet() to handle GET requests and doPost() to handle POST requests.

An HTTP servlet can override either or both of these methods, depending on the type of requests it needs to handle. The service() method of HttpServlet handles the setup and dispatching to all the doXXX() methods.
Q.25 What is HttpServletRequest class?
- The HttpServletRequest class provides an abstract class that is to be subclassed which is suitable for a web site.
- The subclass of HttpServletRequest class must override any one of the following methods:
  1. doGet()
  2. doPost()
  3. doPut()
  4. doDelete()
  5. init()
  6. destroy()
  7. getServletInfo()
- The service() method overriding has almost no reason because it handles the HTTP requests by the handler methods of HTTP request type such as doxxx() methods.

Q.26 Write a Servlet program to display the simple sentence Welcome along with the name entered by the user through an HTML form.

(A) Index.html
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html">
  <title>JSP Page</title>
</head>
<body>
  <form method="post" action="WelcomeServlet">
    <label title="Enter me">N me : </label>
    <input type="text" id="txtN me" name="txtName"/>
    <input type="submit" value="Submit">
  </form>
</body>
</html>

WelcomeServlet.java
import java.io.*;
import java.net.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class WelcomeServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException
    response.setContentType("text/html ");
    PrintWriter out = response.getWriter();
    try {

        out.println("<html>");
        out.println("<head>");
        out.println("<title>ServletWelcomeServlet</title>");

        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet WelcomeServlet at " + request.getParameter("txtName") + "</h1>");

        out.println("</body>");
        out.println("</html>");

    } finally {

        out.close();
    }
}