FSJP

1) Class and Object

```
class Student {
 String name;
 int age;
void displayInfo() {
 System.out.println("Name: " + name);
 System.out.println("Age: " + age);
 }
}
public class Main {
 public static void main(String[] args) {
 Student student1 = new Student();
 student1.name = "Alice";
 student1.age = 20;
 student1.displayInfo();
 }
}
```

2) Method and Constructor

- ### Constructor Overloading

```
class Student {
 Student() {
 System.out.println("Default Constructor");
 }
 Student(String name) {
 System.out.println("Name: " + name);
 }
 Student(int age, String course) {
 System.out.println("Age: " + age + ", Course: " + course);
 }
 public static void main(String[] args) {
 Student s1 = new Student();
 Student s2 = new Student("Gauri");
 Student s3 = new Student(20, "Java");
 }
}
```

- ### Method Overloading

```
class Calculator {
 void add(int a, int b) {
 System.out.println("Sum: " + (a + b));
 }
 void add(double a, double b) {
 System.out.println("Sum: " + (a + b));
 }
void add(String a, String b) {
 System.out.println("Concatenation: " + (a + b));
 }
 public static void main(String[] args) {
 Calculator c = new Calculator();
 c.add(5, 10);
 c.add(2.5, 3.5);
 c.add("Hello", "Java");
 }
}
```

3) Inheritance and Exception handling

- Inheritance

```
class Vehicle {
String brand;
Vehicle(String brand) {
this.brand = brand;
}
void showBrand() {
System.out.println("Vehicle Brand: " + brand);
}
}
class Car extends Vehicle {
String model;
Car(String brand, String model) {
super(brand);
this.model = model;
}
void showModel() {
System.out.println("Car Model: " + model);
}
}
public class InheritanceDemo {
public static void main(String[] args) {
Car myCar = new Car("Toyota", "Corolla");
myCar.showBrand();
myCar.showModel();
}
}
```

- Exception handling

```
public class ExceptionHandlingDemo {
public static void main(String[] args) {
try {
int a = 10;
int b = 0;
int result = a / b;
System.out.println("Result: " + result);
} catch (ArithmeticException e) {
System.out.println("Exception caught: Division by zero is not allowed.");
} finally {
System.out.println("Finally block executed.");
}
System.out.println("Program continues after exception handling.");
}
}
```

4) Website using HTML and CSS

```
<!DOCTYPE html>
<html>
<head>
<title>My Simple Website</title>
<style>
body {
font-family: Arial, sans-serif;
margin: 0;
padding: 0;
background-color: #f4f6f9;
color: #333;
}
header {
background-color: #4CAF50;
color: white;
padding: 20px;
text-align: center;
}
main {
padding: 20px;
max-width: 800px;
margin: auto;
}
h1, h2 {
color: #4CAF50;
}
```

```
a {
color: #4CAF50;
text-decoration: none;
}
a:hover {
text-decoration: underline;
}
footer {
text-align: center;
background: #333;
color: white;
padding: 10px;
margin-top: 20px;
}
ul {
list-style-type: square;
padding-left: 20px;
}
</style>
</head>
<body>
<header>
<h1>Welcome to My Website</h1>
<p>This is a very simple website made with HTML + CSS</p>
</header>
<main>
<h2>About</h2>
<p>I am learning HTML and CSS. This is my first styled web page.</p>
<h2>Links</h2>
<ul>
<li><a href="https://www.google.com">Go to Google</a></li>
<li><a href="https://www.wikipedia.org">Go to Wikipedia</a></li>
</ul>
<h2>Contact</h2>
<p>You can email me at <a
href="mailto:example@example.com">example@example.com</a></p>
</main>
<footer>
<p>&copy; 2025 My Simple Website</p>
</footer>
</body>
</html>
```

5) Mail validation using java script

```
<!DOCTYPE html>
<html>
<head>
<title>Email Validation</title>
<script>
function validateEmail() {
let email = prompt("Please enter your email address:");
// Regex for simple email validation
const emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
// Keep asking until a valid email is entered
while (true) {
if (emailPattern.test(email)) {
alert("Email is valid: " + email);
break;
} else {
alert(" Invalid email! Please enter a valid format (example: user@example.com).");
email = prompt("Re-enter your email address:");
}
}
}
</script>
</head>
<body>
<h2>Email Validation Example</h2>
<button onclick="validateEmail()">Enter Email</button>
</body>
</html>
```

6) Change the background color

```
<!DOCTYPE html>
<html>
<head>
<title>Auto Background Color Change</title>
<script type="text/javascript">
var colors = ["#FFB6C1", "#ADD8E6", "#90EE90", "#FFD700", "#FF7F50"];
var index = 0;
function changeColor() {
document.body.style.backgroundColor = colors[index];
index = (index + 1) % colors.length;
}
setInterval(changeColor, 5000);
</script>
```

```
</head>
<body>
<h2 style="text-align:center;">Background changes every 5 seconds</h2>
</body>
</html>
```

7) Servlet

- Generic servlet

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
*
* @author DS
*/
public class MyGenericServlet extends HttpServlet {
@Override
public void service(ServletRequest request, ServletResponse response)
throws ServletException, IOException {
// Set the content type of the response
response.setContentType("text/html");
// Write HTML content to the response
try ( // Get a PrintWriter to write the response
PrintWriter out = response.getWriter()) {
// Write HTML content to the response
out.println("<html>");
out.println("<head><title>Generic Servlet Example</title></head>");
out.println("<body>");
out.println("<h2>Hello from MyGenericServlet!</h2>");
out.println("<p>This is a simple generic servlet.</p>");
out.println("</body>");
out.println("</html>");
// Close the PrintWriter
}
}
}
```

- Http servlet

```
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
public class MyHttpServlet extends HttpServlet {
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html");
response.getWriter().println("<h1>Hello from HTTP Servlet (GET)!</h1>");
}
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html");
response.getWriter().println("<h1>Hello from HTTP Servlet (POST)!</h1>");
// Process form data, etc.
}
}
```

8) JSP implicit and explicit

```
<%@ page language="java" contentType="text/html" pageEncoding="UTF-8"
isErrorPage="true" %>
<html>
<head><title>JSP Implicit & Explicit Objects</title></head>
<body>
<%
// 1. request - get parameter
String name = request.getParameter("name");
if (name == null) name = "Guest";
// 2. response - set content type (optional, usually set by page directive)
response.setContentType("text/html");
// 3. out - print
out.println("<h3>Hello, " + name + "</h3>");
// 4. session - save attribute
session.setAttribute("username", name);
// 5. application - set app-wide attribute
application.setAttribute("appTitle", "Simple JSP Demo");
// 6. config - get servlet name
out.println("<p>Servlet Name: " + config.getServletName() + "</p>");
// 7. pageContext - get attribute from application scope
```

```
out.println("<p>Application Title: " + pageContext.getAttribute("appTitle",
PageContext.APPLICATION_SCOPE) + "</p>");
// 8. page - current servlet instance class
out.println("<p>Page Object: " + page.getClass().getName() + "</p>");
// 9. exception - only for error pages
if (exception != null) {
out.println("<p>Exception: " + exception.getMessage() + "</p>");
} else {
out.println("<p>No exception on this page.</p>");
}
// Explicit object: create current date
java.util.Date date = new java.util.Date();
out.println("<p>Current Date and Time: " + date.toString() + "</p>");
%>
<form method="get" action="example.jsp">
Enter your name: <input type="text" name="name" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

9) Database connectivity

```
package javaapplication1;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class JavaApplication1 {
public static void main(String[] args) throws SQLException {
Connection
con=DriverManager.getConnection("jdbc:derby://localhost:1527/college","meet","
meet");
System.out.println("Connection created");
java.sql.Statement stmt=con.createStatement();
System.out.println("Statement created");
stmt.executeUpdate("create table library(sname varchar(10),rollno int,gender
varchar(10),class
varchar(10),bookname varchar(10),authorname varchar(10),issuedate
varchar(10),returndate
varchar(10))");
System.out.println("table created");
stmt.executeUpdate("insert into library values('meet',1,'male','sy','java','gauri','2
feb','9 feb')");
System.out.println("Record inserted");
}
}
```

10) React

```
import React from "react";
function App() {
return (
<div style={{ fontFamily: "Arial, sans-serif" }}>
{/* Header */}
<header style={{ background: "#4CAF50", padding: "20px", color: "white" }}>
<h1>My Simple React Page</h1>
<nav>
<a href="#home" style={{ margin: "0 15px", color: "white", textDecoration: "none"
}}>Home</a>
<a href="#about" style={{ margin: "0 15px", color: "white", textDecoration: "none"
}}>About</a>
<a href="#contact" style={{ margin: "0 15px", color: "white", textDecoration: "none"
}}>Contact</a>
</nav>
</header>
{/* Main Content */}
<main style={{ padding: "20px", textAlign: "center" }}>
<section id="home">
<h2>Welcome!</h2>
<p>This is a simple webpage built with React.</p>
</section>
<section id="about" style={{ marginTop: "40px" }}>
<h2>About</h2>
<p>React makes it easy to build interactive UIs with reusable components.</p>
</section>
<section id="contact" style={{ marginTop: "40px" }}>
<h2>Contact</h2>
<p>Email: example@email.com</p>
</section>
</main>
{/* Footer */}
<footer style={{ background: "#333", color: "white", padding: "10px", marginTop:
"20px",
textAlign: "center" }}>
<p>© 2025 My React Page</p>
</footer>
</div>
);
}
export default App;
```