

Experiment 3.1

Student Name: Lipakshi

UID: 20BCS5082

Branch: BE-CSE

Section/Group: 607 B

Semester: 5th

Subject Name : PBLJ Lab

1. **Aim:** Create a palindrome creator application for making a longest possible palindrome out of given input string.

2. **Software/Hardware Requirements:** VS Code or Eclipse

3. **Algorithm/ PsuedoCode:**

STEP 1- Create a index.jsp file in a webapp directory.

STEP 2 - Create a package named as fun and create a java file named as functions.java .

STEP 3 - functions.java file contains the logic for checking the palindromic substring .

STEP 4- At Last start the server and display the output on the web browser.

STEP 5- EXIT



CODE:

Index.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
<style>
    body{
        background: linear-gradient(45deg, red,
        blue); background-size: cover;
        color: white;
        align-items: center;
    }
    h1{
        text-align:center;
    }

    .fall{
        border: 2px solid orange;background: blue;
        padding: 5px;
        max-width:
        500px; height:
        100px; margin:
        auto; font-size:
        19px;
    }
    input{
        width: 250px;
    }
    button{
        position:
        relative; left:
        170px;
        margin: 10px; width:
        60px;height:30px;
        cursor:pointer;border-radius:5px;
    }
    button:hover{
        color:white;
        background: black;
    }
</style>
</head>
<body>
    <h1>find the Longest Palindromic Substring</h1>
    <form class="fall" name="funcitons"
action="<%=request.getContextPath()%>/functions" method="post">
        Enter the Palindromic String: <input class="check" type="text" name="pal"
size="50"><br>
        <button type="submit">Submit</button>
```



```
%></h1>  
</body>  
</html>
```

```
<button type="reset">Reset</button>  
</form>  
<h1> longest Palindromic SubString <br/>  
<%=request.getAttribute("ans")
```

Functions.java

```
package fun;
```

```
import java.io.IOException;
```

```
import javax.servlet.ServletException; import
```

```
javax.servlet.annotation.WebServlet; import
```

```
javax.servlet.http.HttpServlet; import
```

```
javax.servlet.http.HttpServletRequest;
```

```
import javax.servlet.http.HttpServletResponse;
```

```
/**
```

```
 * Servlet implementation class functions
```

```
 */
```

```
@WebServlet(name="functions",urlPatterns={"/functions"
```

```
}) public class functions extends HttpServlet {
```

```
    protected void doPost(HttpServletRequest request, HttpServletResponse  
response) throws ServletException, IOException {
```

```
        String a=request.getParameter("pal");
```

```
//        String fun=request.getParameter("fun");
```

```
        try {
```

```
//            System.out.println(a+fun);
```

```
            int n=a.length();
```

```
            String ans;
```

```
        if(n<=1) {
            ans=a;
            request.setAttribute("ans",ans);

request.getRequestDispatcher("index.jsp").forward(request,response);
        }
        else {
            int len=1,s=0;
            int low,high;
            for(int i=1;i<n;i++) {
                low=i-1;
                high=i+1;
                while(high<n&& a.charAt(high)==a.charAt(i)) {
                    high++;
                }
                while(high<n&& a.charAt(low)==a.charAt(i)) {
                    low++;
                }
                while(low>=0 && high<n &&
a.charAt(low)==a.charAt(high)) {
                    low--;
                    high++;
                }
                int length=high-low-1;
                if(len<length) {
                    len=length;
                    s=low+1;
                }
            }
        }
    }
}
```

```
        }  
    }  
    ans=a.substring(s,s+len);  
    request.setAttribute("ans",ans);  
    request.getRequestDispatcher("index.jsp").forward(request,response);  
}  
}catch(Exception e) {  
    System.out.println(e);  
}  
}  
}
```

OUTPUT:

Discover. Learn. Empower.

localhost:8080/WS_3_1/index.jsp

find the Longest Palindromic Substring

Enter the Palindromic String:

longest Palindromic SubString
null

localhost:8080/WS_3_1/functions

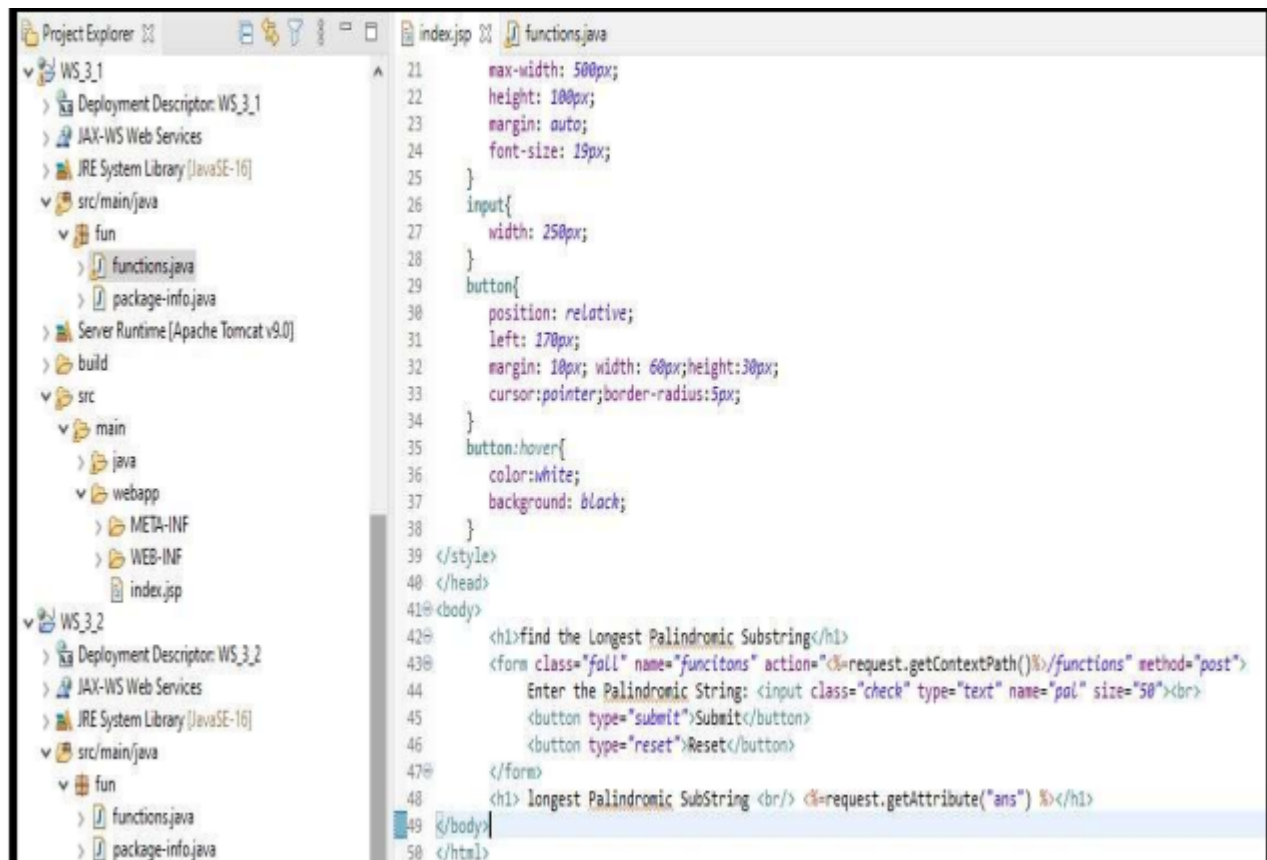
find the Longest Palindromic Substring

Enter the Palindromic String:

longest Palindromic SubString
ssiiss

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

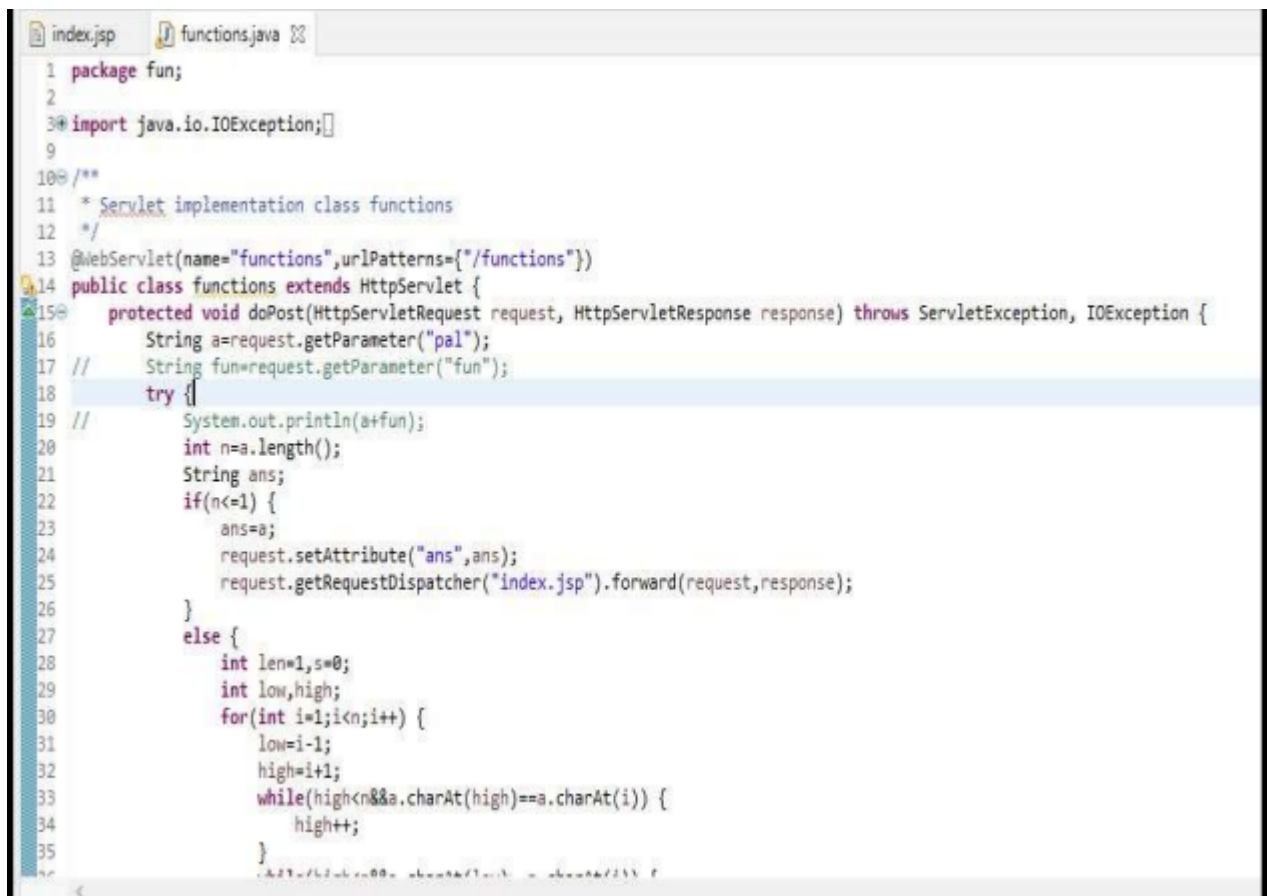
Discover. Learn. Empower.



The screenshot shows an IDE with a Project Explorer on the left and a code editor on the right. The Project Explorer displays a project named 'WS_3_1' with a structure including 'src/main/java' containing 'fun' (with 'functions.java' and 'package-info.java') and 'src' containing 'main' (with 'java' and 'webapp' subdirectories). The code editor shows the content of 'index.jsp' and 'functions.java'.

```
21     max-width: 500px;
22     height: 100px;
23     margin: auto;
24     font-size: 19px;
25 }
26 input{
27     width: 250px;
28 }
29 button{
30     position: relative;
31     left: 170px;
32     margin: 10px; width: 60px; height: 30px;
33     cursor: pointer; border-radius: 5px;
34 }
35 button:hover{
36     color: white;
37     background: black;
38 }
39 </style>
40 </head>
41 <body>
42     <h1>find the Longest Palindromic SubString</h1>
43     <form class="form" name="funcitons" action="<%=request.getContextPath()%/>functions" method="post">
44         Enter the Palindromic String: <input class="check" type="text" name="pal" size="50"><br>
45         <button type="submit">Submit</button>
46         <button type="reset">Reset</button>
47     </form>
48     <h1> longest Palindromic SubString <br/> <%=request.getAttribute("ans") %></h1>
49 </body>
50 </html>
```

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

A screenshot of an IDE window showing the code for functions.java. The code is a Java Servlet implementation. It starts with a package declaration 'package fun;' and an import 'import java.io.IOException;'. A Javadoc comment describes it as a Servlet implementation class. The class is annotated with '@WebServlet(name="functions",urlPatterns={"/functions"})' and is a public class 'functions' extending 'HttpServlet'. The 'doPost' method is implemented, taking 'HttpServletRequest request' and 'HttpServletResponse response' as parameters. It retrieves the 'pal' parameter, calculates its length, and checks if it's less than or equal to 1. If so, it sets an attribute 'ans' and forwards the request to 'index.jsp'. Otherwise, it performs a palindrome check using two pointers, 'low' and 'high', and a 'while' loop to compare characters. The code is as follows:

```
1 package fun;
2
3 import java.io.IOException;
4
5
6 /**
7  * Servlet implementation class functions
8  */
9
10 @WebServlet(name="functions",urlPatterns={"/functions"})
11 public class functions extends HttpServlet {
12     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
13         String a=request.getParameter("pal");
14         // String fun=request.getParameter("fun");
15         try {
16             // System.out.println(a+fun);
17             int n=a.length();
18             String ans;
19             if(n<=1) {
20                 ans=a;
21                 request.setAttribute("ans",ans);
22                 request.getRequestDispatcher("index.jsp").forward(request,response);
23             }
24             else {
25                 int len=1,s=0;
26                 int low,high;
27                 for(int i=1;i<n;i++) {
28                     low=i-1;
29                     high=i+1;
30                     while(high<n&&a.charAt(high)==a.charAt(i)) {
31                         high++;
32                     }
33                 }
34             }
35         }
36     }
37 }
```

Learning outcomes (What I have learnt):

1. Learn About the servlet
2. Learn about jsp and dynamic web project
3. Learn about the tomcat server and its integrations with the java.