1. **Write a html program to create a form that accepts the user inputs and send it to the server**

Top of Form

First name:  
  
Last name:  
  
Top of Form



Bottom of Form

If you click the "Submit" button, the form-data will be sent to a page called "/action\_page.php".

**PROGRAM CODE**

<!DOCTYPE html>

<html>

<body>

<h2>HTML Forms</h2>

<form action="/action\_page.php">

<label for="fname">First name:</label><br>

<input type="text" id="fname" name="fname" value="John"><br>

<label for="lname">Last name:</label><br>

<input type="text" id="lname" name="lname" value="Doe"><br><br>

<input type="submit" value="Submit">

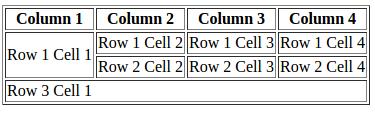
</form>

<p>If you click the "Submit" button, the form-data will be sent to a page called "/action\_page.php".</p>

<body><html>

**2. Write a html program to print the tables with all the specifications as displayed**

<!DOCTYPE html>  
<html>  
     <head>  
          <title>HTML Table Colspan & Rowspan Example</title>  
     </head>  
     <body>  
          <table border="1">  
               <tr>  
                    <th>Column 1</th>  
                    <th>Column 2</th>  
                    <th>Column 3</th>  
                    <th>Column 4</th>  
               </tr>  
               <tr>  
                    <td rowspan="2">Row 1 Cell 1</td>  
                    <td>Row 1 Cell 2</td>  
                    <td>Row 1 Cell 3</td>  
                    <td>Row 1 Cell 4</td>  
               </tr>  
               <tr>  
                    <td>Row 2 Cell 2</td>  
                    <td>Row 2 Cell 3</td>  
                    <td>Row 2 Cell 4</td>  
               </tr>  
               <tr>  
                    <td colspan="4">Row 3 Cell 1</td>  
               </tr>  
          </table>  
     </body>  
</html>

**Output:**  


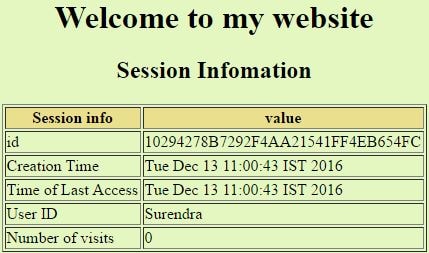
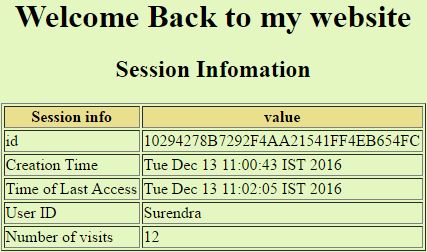
**3. Write an application to demonstrate the session tracking in Servlet.**  
  
**Answer:**  
  
Session is basically a time frame and tracking means maintaining user data for certain period of time frame. Session Tracking is a mechanism used by the web container to store session information for a particular user. It is used to recognize a particular user.  
  
Below example shows how HttpSession object finds out creation time and last accessed time for a session.

#### **SessionDemo.java**

import java.io.\*;  
import javax.servlet.\*;  
import javax.servlet.http.\*;  
import java.util.\*;  
  
// Extend HttpServlet class  
public class SessionDemo extends HttpServlet  
{  
     public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException  
     {  
           // Create a session object if it is already not  created.  
           HttpSession session = request.getSession(true);  
           // Get session creation time.  
           Date createTime = new Date(session.getCreationTime());  
           // Get last access time of this web page.  
           Date lastAccessTime = new Date(session.getLastAccessedTime());  
       
           String title = "Welcome Back to my website";  
           Integer visitCount = new Integer(0);  
           String visitCountKey = new String("visitCount");  
           String userIDKey = new String("userID");  
           String userID = new String("Surendra");  
  
           // Check if this is new comer on your web page.  
           if (session.isNew())  
          {  
               title = "Welcome to my website";  
               session.setAttribute(userIDKey, userID);  
          }  
          else  
          {  
               visitCount = (Integer)session.getAttribute(visitCountKey);  
               visitCount = visitCount + 1;  
               userID = (String)session.getAttribute(userIDKey);  
          }  
          session.setAttribute(visitCountKey,  visitCount);  
  
          // Set response content type  
          response.setContentType("text/html");  
          PrintWriter out = response.getWriter();  
  
          String docType =  
          "<!doctype html public \"-//w3c//dtd html 4.0 " +  
          "transitional//en\">\n";  
  
           out.println(docType +  
               "<html>\n" +  
               "<head><title>" + title + "</title></head>\n" +  
               "<body bgcolor=\"#e5f7c0\">\n" +  
               "<h1 align=\"center\">" + title + "</h1>\n" +  
               "<h2 align=\"center\">Session Infomation</h2>\n" +  
               "<table border=\"1\" align=\"center\">\n" +  
               "<tr bgcolor=\"#eadf8c\">\n" +  
               "<th>Session info</th><th>value</th></tr>\n" +  
               "<tr>\n" +  
               "  <td>id</td>\n" +  
               "  <td>" + session.getId() + "</td></tr>\n" +  
               "<tr>\n" +  
               "  <td>Creation Time</td>\n" +  
               "  <td>" + createTime +  
               "  </td></tr>\n" +  
               "<tr>\n" +  
               "  <td>Time of Last Access</td>\n" +  
               "  <td>" + lastAccessTime +  
               "  </td></tr>\n" +  
               "<tr>\n" +  
               "  <td>User ID</td>\n" +  
               "  <td>" + userID +  
               "  </td></tr>\n" +  
               "<tr>\n" +  
               "  <td>Number of visits</td>\n" +  
               "  <td>" + visitCount + "</td></tr>\n" +  
               "</table>\n" +  
               "</body></html>");  
     }  
}

#### **web.xml**

<web-app>  
     <servlet>  
          <servlet-name>abc</servlet-name>  
          <servlet-class>SessionDemo</servlet-class>  
     </servlet>  
     <servlet-mapping>  
          <servlet-name>abc</servlet-name>  
          <url-pattern>/test</url-pattern>  
     </servlet-mapping>  
</web-app>

**Output:**  
  
**i. When we run the application first time**  
  
  
  
**ii. After refreshing the application**  
  


**4. Create an Html page that contains 4 option buttons Java, UNIX, DDBMS, OOSE and 2 buttons Submit and Reset. When the user clicks on Submit button the server responds by adding cookie containing the selected Subject and sends the html page to the client. Program should not allow duplicate cookie to be written.**  
                      
**Answer:**  
  
Information which is stored on the client machine is called cookies. It has parameters like name, value, path, host, expires and connection type.  
  
In this example, when we click on submit button after checking the value, the cookies add selected value.

#### **index.html**

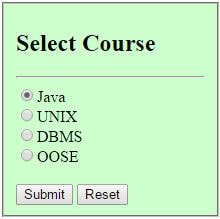
<!doctype html>  
     <head>  
          <title>CookiesExample</title>  
     </head>  
     <body>  
          <form method='post' action='servlet/cookies'>  
               <fieldset style="width:14%; background-color:#ccffcc">  
                    <h2>Select Course</h2> <hr>  
                         <input type='radio' name='course' value='Java'>Java<br>    
                         <input type='radio' name='course' value='UNIX'>UNIX<br>    
                         <input type='radio' name='course' value='MCA'>DBMS<br>    
                         <input type='radio' name='course' value='OOSE '>OOSE<br><br>  
                         <input type='submit'> <input type='reset'><br>    
               </fieldset>  
          </form>    
     </body>  
</html>

#### **AddCookie.java**

import java.io.\*;  
import javax.servlet.\*;  
import javax.servlet.http.\*;    
public class AddCookie extends HttpServlet  
{    
     public void doPost(HttpServletRequest req, HttpServletResponse res) throws ServletException,IOException  
     {  
          res.setContentType("text/html");  
          PrintWriter pw = res.getWriter();  
          Cookie []c = req.getCookies();  
          int id=1;  
          if(c!=null) id = c.length+1;  
               String value = req.getParameter("course");  
          Cookie newCookie = new Cookie("course:"+id,value);  
          res.addCookie(newCookie);    
          pw.println("<h4>Cookie added with value "+value+"</h4>");  
     }  
  
}

#### **web.xml**

<web-app>  
     <servlet>  
          <servlet-name>AddCookie</servlet-name>  
          <servlet-class>AddCookie</servlet-class>  
     </servlet>  
     <servlet-mapping>  
          <servlet-name>AddCookie</servlet-name>  
          <url-pattern>/servlet/cookies</url-pattern>  
     </servlet-mapping>  
     <welcome-file-list>  
          <welcome-file>index.html</welcome-file>  
     </welcome-file-list>  
</web-app>

**Output:**  
  
Select Java and click on submit. It adds the cookies with value Java.  
  
  
  
cookies

**5. Write a Servlet application to count the total number of visits on your website.**  
  
**Answer:**  
  
When first time servlet runs then session is created and value of the counter will be zero and after access of servlet again, the counter value will be increased by one.

#### **CounterServlet.java**

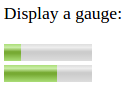
import java.io.IOException;  
import java.io.PrintWriter;  
import javax.servlet.ServletException;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
public class CounterServlet extends HttpServlet  
{  
     //Instance variable used for counting hits on this servlet  
     private int iHitCounter;  
  
     //init method just initializes the hitCounter to zero  
     public void init() throws ServletException  
     {  
          iHitCounter = 0;  
     }  
     public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException  
     {  
          PrintWriter out =  response.getWriter();  
          out.println("<form><fieldset style='width:15%'>");  
          out.println("<h3>Welcome to my website !</h3><hr>");  
          out.println("You are visitor number: "+ (++iHitCounter));  
          out.println("</fieldset></form>");  
     }  
     public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException  
     {  
          doGet(request, response);  
     }  
}

**Output:**  
  


**6. Write a program to show the Usage of progress and meter tag**

Progress tag represents the completion progress of a task whereas Meter tag is used to represent gauges.  
  
Progress tag represents a dynamic data whereas Meter tag represents a static data.  
  
Meter tag is used to represent a scalar measurement within a known range and should not be used to indicate the progress. We have the progress tag to indicate the progress.  
  
**Meter Tag**

<!DOCTYPE html>  
<html>  
   <body>  
      <p>Display a gauge:</p>  
      <meter value="2" min="0" max="10">2 out of 10</meter><br>  
      <meter value="0.6">60%</meter>  
   </body>  
</html>

  
  
**Progress Tag**

<!DOCTYPE html>  
<html>  
   <body>  
      Downloading:  
      <progress value="22" max="100">  
      </progress>  
   </body>  
</html>

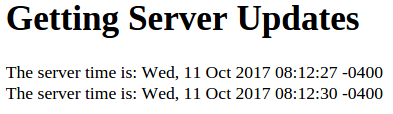
progress tag

## **7. Write a program to describe the Server-Sent Event in HTML5?**

**Answer:**  
  
SSE is a technology where a browser receives automatic updates from a server via HTTP connection.  
  
Server-Sent Events (SSE) allows a web page to get updates from a server.  
  
It describes how servers can initiate data transmission towards clients once an initial client connection has been established.

#### **Example:**

<!DOCTYPE html>  
<html>  
   <body>  
      <h1>Getting Server Updates</h1>  
      <div id="result"></div>  
      <script>  
         if(typeof(EventSource) !== "undefined") {  
            var source = new EventSource("demo\_sse.php");  
            source.onmessage = function(event) {  
               document.getElementById("result").innerHTML += event.data + "<br>";  
            };  
         } else {  
            document.getElementById("result").innerHTML = "Sorry, your browser does not support server-sent events...";  
         }  
      </script>  
   </body>  
</html>

  
  
The EventSource object is used to receive server-sent event notifications.

## **8. Write a program to show how XML work with CSS?**

* Cascading Style Sheet (CSS) controls the format of XML and HTML tags and presents the XML and HTML data.
* If CSS is not used with XML, the XML document will be displayed on the browser with its tag.

## **Linking XML with CSS**

**Step1 : Write a XML document containing information and save with .xml extension.**  
We have created people.xml.

<?xml version="1.0" encoding="UTF-8"?>  
<people>  
     <person>  
          <name>Nirja Shaha</name>  
          <education>MCA</education>  
          <City>Pune</City>  
     </person>  
     <person>  
          <name>Prashant Saxena</name>  
          <education>B.Tech</education>  
          <City>Pune</City>  
     </person>  
</people>

**Step2 : Write CSS code and save with .CSS file with XML code.**

people  
{  
     background-color: white;  
     display: block;  
     font-family: arial;  
}  
person  
{  
     background-color: white;  
     display: block;  
}  
name  
{  
     font-weight: bold;  
     display: block;  
}  
education  
{  
     display: block;  
     color: blue;  
}  
City  
{  
     display: block;  
}

**Step3 : Link CSS file, in XML document as:**  
<?xml-stylesheet type="text/css" href="people.css"?>  
  
Run the people.xml file.  
  
**Output:**  

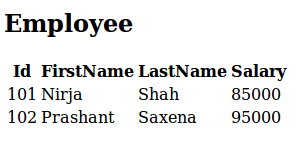

## **9. Write a program to show the transformation of XML document using XSLT.**

**Step 1: Create an XML document which contains information of employee and save with employee.xml.**

<?xml version="1.0"?>  
<class>  
     <Employee Id="101">  
          <firstname>Nirja</firstname>  
          <lastname>Shah</lastname>  
          <salary>85000</salary>  
     </Employee>  
     <Employee Id="102">  
          <firstname>Prashant</firstname>  
          <lastname>Saxena</lastname>  
          <salary>95000</salary>  
     </Employee>  
</class>

In above XML document we have mentioned a employee information.  
  
**Step 2: Create XSLT document to fulfill the above requirement  and save with employee.xsl in same folder along with employee.xml file.**

<?xml version="1.0" encoding="UTF-8"?>  
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">    
<xsl:template match="/">  
<html>  
     <body>  
     <h2>Employee</h2>  
          <table>  
          <tr>  
               <th>Id</th>  
               <th>FirstName</th>  
               <th>LastName</th>  
               <th>Salary</th>  
          </tr>  
          <xsl:for-each select="class/Employee">  
          <tr>  
               <td><xsl:value-of select="@Id"/></td>  
               <td><xsl:value-of select="firstname"/></td>  
               <td><xsl:value-of select="lastname"/></td>  
               <td><xsl:value-of select="salary"/></td>  
          </tr>  
          </xsl:for-each>  
          </table>  
     </body>  
</html>  
</xsl:template>  
</xsl:stylesheet>

**Step 3:** Link the XSLT document to XML document by following xml-stylesheet tag.  
<?xml-stylesheet type="text/xsl" href="employee.xsl"?>  
  
**Step 4:** Run the employee.xml file in the browser.  
  
**Output:**  


## **10. Write a program to demonstrate the difference between the class selector and an ID selector**

**Answer:**  
  
Class selector can be applied to multiple HTML elements, whereas ID selectors are unique and can only be applied to a single element.  
  
The class selector selects elements with a specific class attribute. To select elements with a specific class, write a period (.) character, followed by the name of the class.

The id selector uses the id attribute of an HTML element to select a specific element. To select an element with a specific id, write a hash (#) character, followed by the id of the element.

#### **Example:**

<!DOCTYPE html>  
<html>  
    <head>  
        <style>  
            p.class-selector {  
        background: yellow;  
                text-align: center;  
                color: blue;  
                font-size: 200%  
            }  
            #id-selector {  
                background-color: pink;  
            }  
        </style>  
    </head>  
    <body>  
        <p class="class-selector">This is an example of class selector.</p>  
        <p id="id-selector">This is an example of id selector.</p>  
    </body>  
</html>

## **Output:** class selector and id selector**11. Write a simple form Validation Program to verify email address**

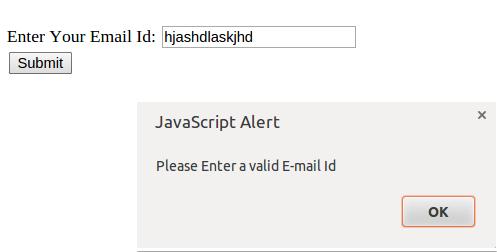
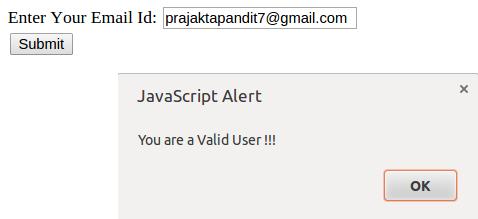
* Form validation is the process of checking the forms that have been filled in correctly before they are processed.
* It provides a method to check the user entered information on client-side before the data is submitted to the server-side.

**It includes two methods for validating forms:**  
1. Server-Side (ASP, PHP)  
2. Client-Side (JavaScript)

* It displays alerts for incorrect data entered by the user.
* Client-side validation is faster than Server-side validation.

#### **Example : Simple Form Validation Program**

**validation.html          //File name**  
  
<html>  
<body>  
     <script>    
     function validateemail()    
     {    
          var a = document.myform.email.value;    
          var atposition = a.indexOf("@");    
          var dotposition = a.lastIndexOf(".");    
          if (atposition<1 || dotposition<atposition+2 || dotposition+2>=a.length)  
          {    
               alert("Please Enter a valid E-mail Id");    
               return false;    
          }    
     }    
     </script>    
</body>  
<body>  
     <form name="myform"  method="post" action="validpage.html" onsubmit="return validateemail();">    
          Enter Your Email Id: <input type="text" name="email"><br/>    
          <input type="submit" value="Submit">    
     </form>    
</body>  
</html>  
  
**validpage.html          //File name**  
  
<html>  
     <body>  
     <script type="text/javascript">  
          alert("You are a Valid User !!!");  
     </script>  
     </body>  
</html>

**Output:**  
  
  


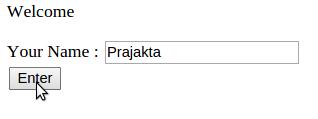
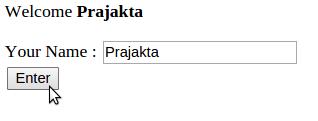
#### **12. Write a Simple Program on Navigator Object**

<html>  
     <body>  
     <script type="text/javascript">  
          document.write("<b>Browser: </b>"+navigator.appName+"<br><br>");  
          document.write("<b>Browser Version: </b>"+navigator.appVersion+"<br><br>");  
          document.write("<b>Browser Code: </b>"+navigator.appCodeName+"<br><br>");  
          document.write("<b>Platform: </b>"+navigator.platform+"<br><br>");  
          document.write("<b>Cookie Enabled: </b>"+navigator.cookieEnabled+"<br><br>");  
          document.write("<b>User Agent: </b>"+navigator.userAgent+"<br><br>");  
          document.write("<b>Java Enabled: </b>"+navigator.javaEnabled()+"<br><br>");  
     </script>  
     </body>  
</html>

**Output:**  
  
**Browser:** Netscape  
  
**Browser Version:** 5.0 (X11; Linux x86\_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/37.0.2062.120 Chrome/37.0.2062.120 Safari/537.36  
  
**Browser Code:** Mozilla  
  
**Platform:** Linux x86\_64  
  
**Cookie Enabled:** true  
  
**User Agent:** Mozilla/5.0 (X11; Linux x86\_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/37.0.2062.120 Chrome/37.0.2062.120 Safari/537.36  
  
**Java Enabled:** true

#### **13. Write a Simple Program on Element Object**

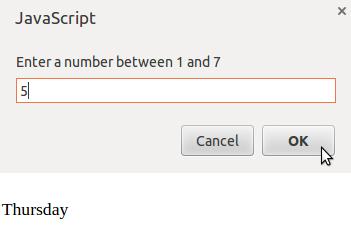
<html>  
<head>  
     <script type="text/javascript">  
     function welcome()  
     {  
          var fname = document.getElementById('fname');  
          var buttxt = document.getElementById('buttxt');  
          buttxt.innerHTML = fname.value;  
     }  
     </script>  
</head>  
<body>  
     <p>Welcome <b id = 'buttxt'></b></p>  
     Your Name : <input type = "text" id="fname"><br>  
     <input type = "button" onClick="welcome()" value="Enter">  
</body>  
</html>

**Output:**  
  
   

#### **14. Write a Simple Program for Switch Statement**

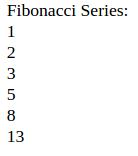
<html>  
<head>  
     <script type="text/javascript">  
     var day = prompt("Enter a number between 1 and 7");  
     switch (day)  
     {  
          case (day="1"):  
               document.write("Sunday");  
               break;  
          case (day="2"):  
               document.write("Monday");  
               break;  
          case (day="3"):  
               document.write("Tuesday");  
               break;  
          case (day="4"):  
               document.write("Wednesday");  
               break;  
          case (day="5"):  
               document.write("Thursday");  
               break;  
          case (day="6"):  
               document.write("Friday");  
               break;  
          case (day="7"):  
               document.write("Saturday");  
               break;  
          default:  
               document.write("Invalid Weekday");  
               break;  
     }  
     </script>  
</head>  
</html>

OUTPUT



#### **15.Write a Fibonacci Series Program using DOM and While Loop**

<html>  
<body>  
     <script type="text/javascript">  
          var no1=0,no2=1,no3=0;  
          document.write("Fibonacci Series:"+"<br>");  
          while (no2<=10)  
          {  
               no3 = no1+no2;  
               no1 = no2;  
               no2 = no3;  
               document.write(no3+"<br>");  
          }  
     </script>  
</body>  
</html>

**Output:**  


**16. Write a simple JSP program to print the current date and time.**  
  
**Answer:**  
  
The most important use of JSP is that we can use all the method available in core java. The Date class is available in java.util package.  
  
Below program shows how to print the current date and time. We can use simple Date object with toString() to print current date and time.

#### **test.jsp**

<html>  
    <head><title>JSPApp</title></head>  
    <body>  
        <form>  
            <fieldset style="width:20%; background-color: #ccffeb;">  
                <legend><b><i>JSP Application<i><b></legend>  
                <h3>Current Date and Time is :</h3>  
                <% java.util.Date d = new java.util.Date();  
                out.println(d.toString()); %>  
            </fieldset>  
        </form>  
    </body>  
</html>

#### **web.xml**

<web-app>  
    <servlet>  
        <servlet-name>xyz</servlet-name>  
        <jsp-file>/test.jsp</jsp-file>  
    </servlet>  
    <servlet-mapping>  
        <servlet-name>xyz</servlet-name>  
        <url-pattern>/test</url-pattern>  
    </servlet-mapping>  
</web-app>

**Output:**  
  


**17. Create a JSP page to accept a number from the user and display it in words.  
Example: 123 - One Two Three.**  
  
**Answer:**  
  
The index.html taking the number from the user and then change it into the words.

#### **index.html**

<!DOCTYPE html>  
<html>  
     <body>  
          <form method=get action="test.jsp">  
               Enter Any Number : <input type=text name=num><br><br>  
               <input type=submit value="Display">  
          </form>  
     </body>  
</html>

#### **test.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>  
<!DOCTYPE html>  
<html>  
    <body>  
       <font color=red>  
          <%! int i,n;  
               String s1;  
          %>  
          <%   s1= request.getParameter("num");  
               n=s1.length();  
               i=0;  
               do  
               {  
                    char ch=s1.charAt(i);  
                    switch(ch)  
                    {  
                         case '0': out.println("Zero  ");  
                              break;  
                         case '1': out.println("One  ");  
                              break;  
                         case '2': out.println("Two  ");  
                              break;  
                         case '3': out.println("Three  ");  
                              break;  
                         case '4': out.println("Four ");  
                              break;  
                         case '5': out.println("Five  ");  
                              break;  
                         case '6': out.println("Six  ");  
                              break;  
                         case '7': out.println("Seven  ");  
                              break;  
                         case '8': out.println("Eight  ");  
                              break;  
                         case '9': out.println("Nine  ");  
                              break;  
                    }  
                    i++;  
               }  
               while(i<n);  
          %>  
       </font>  
   </body>  
</html>

#### **web.xml**

<web-app>  
     <servlet>  
          <servlet-name>xyz</servlet-name>  
          <jsp-file>/test.jsp</jsp-file>  
     </servlet>  
     <servlet-mapping>  
          <servlet-name>xyz</servlet-name>  
          <url-pattern>/test</url-pattern>  
     </servlet-mapping>  
     <welcome-file-list>    
          <welcome-file>index.html</welcome-file>    
     </welcome-file-list>    
</web-app>

**Output:**  
  
  
change numbre

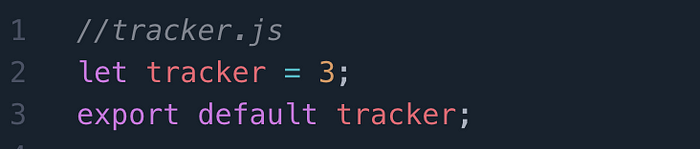
**18. Predict the output of the code snippet below:**



**Answer:**

“155cm” is logged to the console. Let’s take a look at what’s happening. On line 6 when the constructor function is called, the value of 165cm is assigned to the property of height. Even though 165 is assigned, the return statement is ultimately what dictates the returned value when height is accessed. Think of the return statement as an override, as it is not actually changing the value of height, it is simply reporting the value the programmer tells it to.

19. What will be logged to the console after running the attached code snippet?





**Answer:**

An error will be thrown. An imported module is read-only, so we cannot modify *tracker.js* from *script.js.* Only the exporting module, in this case *tracker.js*can modify this value.

## **20. Write a program to demonstrate the HTML links colors**

Colors of hyperlink can be set by using the following attributes.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Link | Used to change the default color of the hyperlink. |
| Alink | Used to change the default color of the active hyperlink. |
| Vlink | Used to change the default color of the visited hyperlink. |

#### **Example:**

<!DOCTYPE html>  
<html>  
     <head>  
          <title>Hyperlink Example</title>  
     </head>  
     <body link="#000080" alink="#54A250" vlink="#ff0000">  
          <p>Click on the following link</p>    
          <a href="http://www.tutor.com/" target="\_blank">Tutor.com</a>  
     </body>  
</html>

**Output:**

Click on the following link

Tutor.com