Midterm Exam
Fall 2019
Due: 11/08/2019, 11:59 P.M.

No delay is accepted whatsoever. After the due date I grade the test for 0
Copy/paste your answer under the word: *Answer* in this document and post it in the folder *Midterm exam on the blackboard*.
I cannot answer the questions to help you, but if you like to clarify anything about the questions email me at mbadii@pace.edu. In this case let me know which part of a question is not clear for you.

Question 1 (Multiple choice question):

Part I) Thread 1 sends four messages successively to Thread 2. Suppose connectionless-oriented socket is used to send each message. What is the possible order in which the message may arrive at Thread 2?

a) 1
b) 8
c) 16
d) 24

Answer:
d) 24

Part II): Thread 1 sends four messages successively to Thread 2. Suppose connection-oriented socket is used to send each message. What is the possible order in which the message may arrive at Thread 2?

a) 1
b) 8
c) 16
d) 24

Answer:
a) 1

Question 1 End

Question 2 (20 points) Write client-server classes to calculate π. We have a number of clients. The server creates a thread for each client. Each client calculates the estimated value of π and sends to the server. The server takes an average of these values and displays a more precise value of π.

Below is the client class to calculate the estimated value of: π

```
import java.util.*;
import java.io.*;
import java.net.Socket;

public class Main{
  public static void main(String[] args){
    long toss, number_of_tosses, number_in_circle = 0;
    double x, y, distance_squared, pi_estimate;
    Random generator = new Random();
    number_of_tosses = generator.nextInt(5000) + 5000;
    for ( toss=0 ; toss< number_of_tosses ; toss++){
      x = 2*Math.random() - 1;
      y =  2*Math.random() - 1;
      distance_squared = x * x + y * y;
      if(distance_squared <= 1) number_in_circle++;
    }
    pi_estimate = 4 * number_in_circle/((double) number_of_tosses);
    try{
      Socket clientSocket = new Socket("localhost", 4321);
      PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
      out.println(pi_estimate);
      clientSocket.close();
    }catch(Exception e){
    }
  }
}
```

Answer:

```
import java.util.*;
import java.io.*;
import java.net.Socket;

public class Main{
  public static void main(String[] args){
    long toss, number_of_tosses, number_in_circle = 0;
    double x, y, distance_squared, pi_estimate;
    Random generator = new Random();
    number_of_tosses = generator.nextInt(5000) + 5000;
    for ( toss=0 ; toss< number_of_tosses ; toss++){
      x = 2*Math.random() - 1;
      y =  2*Math.random() - 1;
      distance_squared = x * x + y * y;
      if(distance_squared <= 1) number_in_circle++;
    }
    pi_estimate = 4 * number_in_circle/((double) number_of_tosses);
    try{
      Socket clientSocket = new Socket("localhost", 4321);
      PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
      out.println(pi_estimate);
      clientSocket.close();
    }catch(Exception e){
    }
  }
}
```

```
import java.io.IOException;
import java.net.ServerSocket;


public class ServerMid {
      public static void main(String[] args) throws IOException {



            System.out.println("Service ready");

            ServerSocket serverSocket = null;

            System.out.println("Waiting for request");

            try {
                  serverSocket = new ServerSocket(4321);

            }catch (IOException e) {
                  System.out.println("Error: " + e);
                  System.exit(0);
            }

            while(true){
                  MyThread a = new MyThread(serverSocket.accept());
                  Thread t = new Thread(a);
                  t.start();
            }

      }
}
```

```
import java.io.*;
import java.net.*;
public class MyThread implements Runnable{
  Socket clientSocket;

  public static double sum;
  double avg = 0.0;
  public static int count = 0;

  public MyThread(Socket clientSocket){
  this.clientSocket = clientSocket;
 }
  public void run(){
 try{
        BufferedReader br = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
        String str = br.readLine();
             System.out.println("Data has been received");

             sum = sum + Double.parseDouble(str);
             count++;
             avg = sum/count;
             str = avg + "";


             if(str != null) {
                    System.out.println("The server has created a thread for client.
");
                    System.out.println("The more precise value of π is " + str);
                    System.out.println("Count of estimates recieved is : " + count);
                    System.out.println("---------------------------------------------
-------------");
             }

             br.close();
    } catch (IOException e){
        System.out.println("Error from the thread: " + e);}
}
}
```

Question 2 End

Question 3 (20 points) Write client-server classes to calculate the sum of 1 + 2 + 3 + …+ n with ***rmi APIs*** for one client and one server. The code for calculation of sum must be written in a remote method in the server programs. The remote method should return the value of: sum to the client.

The client calls the remote method of servers to receive the sum. The client displays the sum on its console. Please copy/paste your java classes and interface under the word: **Answer**.

Below are the program's skeleton and a sample dialog:

The client class and interface:

```
import java.rmi.Remote;
//This interface is complete. Do not change it.
public interface Q3Interface extends Remote {
  public int calculateSum(int n ) throws java.rmi.RemoteException;
}

import java.rmi.*;
import java.util.*;

public class Q3Client{
  //Complete this class.
}
```

The server classes and interface:

```
import java.rmi.Remote;
//This interface is complete. Do not change it.
public interface Q3Interface extends Remote {
  public int calculateSum(int n ) throws java.rmi.RemoteException;
}


import java.rmi.*;
import java.rmi.server.*;

public class CalculateSumServerImpl extends UnicastRemoteObject implements Q3Interface {
  //Complete this class.
}


import java.rmi.*;
import java.rmi.server.*;
import java.rmi.registry.Registry;
import java.rmi.registry.LocateRegistry;
import java.net.*;

public class Q3Server{
 //Complete this class.
}
```

A sample dialog:
Display output on the server side:

```
RMI registry cannot be located at port 16790
RMI registry created at port 16790
Hello Server ready.
```

Display output on the client side:

```
Enter a positive integer: 5
Lookup completed
The sum is: 15
```

Note: The green: 5 is what the user entered.
Note: This is just a sample. Your program should work for any positive integer for n.

Answer:

```
import java.rmi.Remote;
//This interface is complete. Do not change it.
public interface Q3Interface extends Remote {
public int calculateSum(int n ) throws java.rmi.RemoteException;
}
```

```
import java.rmi.*;
import java.util.*;
import java.util.Scanner;


public class Q3Client{
 //Complete this class.
      public static void main(String args[]) {
            Scanner scan = new Scanner(System.in);
            int numToCalculate = 0;

            System.out.print("Enter a positive integer: ");
            numToCalculate = scan.nextInt();

            try {
```

```
				int port = 16790;
				String host = "localhost";

				String registryURL = "rmi://" + host + ":" + port + "/calculate";
				Q3Interface h = (Q3Interface)Naming.lookup(registryURL);

				System.out.println("Lookup completed ");
				int sumNumbers = h.calculateSum(numToCalculate);
				System.out.println("Total = " + sumNumbers);
			}
			catch (Exception e){
				e.printStackTrace();
			}
		}
}
```

```
import java.rmi.*;
import java.rmi.server.*;
import java.rmi.registry.Registry;
import java.rmi.registry.LocateRegistry;
import java.net.*;

public class Q3Server{
	//Complete this class.
	public static void main(String args[]) {

		try {

			int port = 16790;
			String host = "localhost";

		       CalculateSumServerImpl exportedObj = new CalculateSumServerImpl();
			System.out.println("RMI registry cannot be located at port
16790");

			LocateRegistry.createRegistry(port);

			String registryURL = "rmi://" + host + ":" + port + "/calculate";
			Naming.rebind(registryURL, exportedObj);

			System.out.println("RMI registry created at port 16790");
			System.out.println("Hello Server ready.");

		} catch (Exception e) {
			e.printStackTrace();
		}
	}
}
```

```java
import java.rmi.*;
import java.rmi.server.*;

public class CalculateSumServerImpl extends UnicastRemoteObject implements
Q3Interface {
	//Complete this class.
	/**
	 *
	 */
	private static final long serialVersionUID = 1L;

	//Complete this class.

	public CalculateSumServerImpl() throws RemoteException {
		super( );
	}

	public int calculateSum(int n) throws RemoteException {
		int sumCalculated = 0;

		for(int i = 1; i<=n; i++) {
			sumCalculated = sumCalculated + i;
		}

		return sumCalculated;
	}
}
```

Question 3 End

Question 4 (20 points): Use cookies (and not MySQL) to accomplish the following:
1. Make a folder in drive C named: **Temp**.
2. Copy the following line to a notepad editor (or any other text editor) and save the editor with file name: **names.txt** in the folder **C:\Temp** (On each line there are two words separated by a single space):

```
Jane Doe
Adam Smith
Rose Nigel
David Depaoli
Thomas Iacovelli
Dominick Olsen
Danielle Williams
George Perez
Elaine Silva
Angela Dawnell
Mehdi Badii
```

3. Make a servlet named: AddCookies and copy the following over it.

```
import java.io.File;
import java.io.IOException;
import java.util.Scanner;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/AddCookies")
public class AddCookies extends HttpServlet {
  private static final long serialVersionUID = 1L;
  //This servlet is complete. Do not change it.
  public AddCookies() {
    super();
  }

  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    addCookies(response);
    System.out.println("The program terminated with no error.");
  }

  private void addCookies(HttpServletResponse response){
    Scanner inputStream = null;
    try{
      inputStream = new Scanner(new File("C:\\Temp\\names.txt"));
       while(true){
         String name = inputStream.next();
         String value = inputStream.next();
         Cookie c = new Cookie(name, value);
```

```
            c.setMaxAge(365 * 24*60*60);
            response.addCookie(c);
          }
        }catch(Exception e){
        }
        inputStream.close();
      }
    }
```

4. Run the servlet: AddCookies. This servlet reads the file: names.txt one line at a time to make cookies. The servlet makes a cookie for each line. On each line there are two words. The first word would be the name of the cookie and the second word would be the value of the cookie.

5. Make an html document and copy following over it:

```
<!-- The name of this html document must be: Main.html -->
<!-- Do not change this html document -->
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Q 4</title>
</head>
<body>
    <form action="http://localhost:8080/Project/Main">
    <center>
    <pre>
    User Name:<input name="name" type="text" value=""><br>
    Password: <input name="password" type="text" value=""><br><br>
                   Changing Password<br>
    New Password:               <input name="newPassword" type="text" value=""><br>
    Re enter The New Password:<input name="reEnterPassword" type="text" value=""><br>
    <input name="changePassword" type="Submit"    value="Change the Password">
    </pre>
    </center>
  </form>
</body>
</html>
```

6. Make a servlet named: Main and copy the following over it:

```
/*
For the sake of simplicity assume the user enters strings in all
the input boxes.
*/
import java.io.*;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.Cookie;
import java.util.Scanner;

@WebServlet("/Main")
```

```java
public class Main extends javax.servlet.http.HttpServlet implements
javax.servlet.Servlet {
  private String head, tail;
  public Main() {
    super();
    head = "<html><title>Not In The List</title><body><font color = blue><h1>";
    tail = "</h1></font></body></html>";
  }

  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    //This method is complete. Do not change it.
    String userName = request.getParameter("name");
    String password = request.getParameter("password");
    String newPassword = request.getParameter("newPassword");
    String reEnteredPassword = request.getParameter("reEnterPassword");
    displayCookies(request);
    if(isIn(request, userName, password))
      if(isTheSame(newPassword, reEnteredPassword))
        replace(request, response, userName, password, newPassword);
      else
        isNotTheSame(response, newPassword, reEnteredPassword);
    else
      isNotInTheList(response, userName, password);
    displayCookies(request);
  }

  private boolean isIn(HttpServletRequest request, String userName, String password){
    //Complete this method.
    //Check the cookies. If one of them has the name: userName and the value: password
    //return false
    return false; //Once this method is complete remove this line.
  }

  private boolean isTheSame(String newPassword, String reEnteredPassword){
    //Complete this method
    //Make sure the user entered the same new password twice. If both newPassword and
    //reEnteredPassword have the same string return true otherwise return false.
    return false; //Once this method is complete remove this line.
  }

  private void replace(HttpServletRequest request, HttpServletResponse response,
    String userName, String password, String newPassword){
    //Complete this method
    // This is the case that the program is sure there is a cookie with the name:
    // username and the value: password. Replace the value of this cookie by the
    // value of: newPassword.

  }

  private void isNotInTheList(HttpServletResponse response, String userName, String password){
    //Complete this method.
    //Make an html document to display the content of the console.

    System.out.println(userName + " with the password: " + password + " is not in the list");
  }

  private void isNotTheSame(HttpServletResponse response, String newPassword,
    String reEnteredPassword){
    //Complete this method.
    //Make an html document to display the content of the console.

    System.out.println(newPassword + " must be the same as " + reEnteredPassword);
  }

  private void success(HttpServletResponse response){
    //Complete this method.
    //Make an html document to display the content of the console.

    System.out.println("The password is changed successfully.");
  }

  private void displayCookies(HttpServletRequest request){
    //This method is complete. Do not change it.
    Cookie[] cookies = request.getCookies();
    if (cookies != null && cookies.length > 0)
      for(int i=0; i<cookies.length; i++)
        System.out.println(cookies[i].getName() + ", " + cookies[i].getValue());
      System.out.println("*********************************************************");
  }
}
```

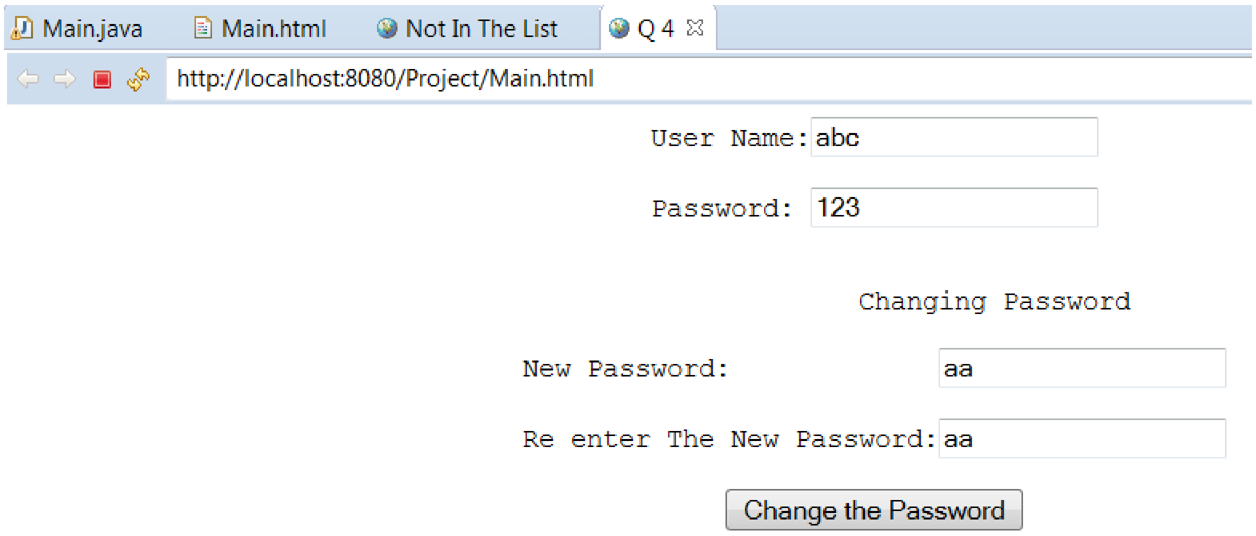
```
/*
For the sake of simplicity assume the user enters strings in all
the input boxes.
*/
import java.io.*;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.Cookie;
import java.util.Scanner;
@WebServlet("/Main")
public class Main extends javax.servlet.http.HttpServlet implements
javax.servlet.Servlet {
  private String head, tail;
  public Main() {
    super();
    head = "<html><title>Not In The List</title><body><font color = blue><h1>";
    tail = "</h1></font></body></html>";
  }
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    //This method is complete. Do not change it.
    String userName = request.getParameter("name");
    String password = request.getParameter("password");
    String newPassword = request.getParameter("newPassword");
    String reEnteredPassword = request.getParameter("reEnterPassword");
    displayCookies(request);
    if(isIn(request, userName, password))
      if(isTheSame(response, newPassword, reEnteredPassword))
        replace(request, response, userName, password, newPassword);
      else
        isNotTheSame(response, newPassword, reEnteredPassword);
    else
      isNotInTheList(response, userName, password);

    displayCookies(request);

  }
  private boolean isIn(HttpServletRequest request, String userName, String password){
    //Complete this method.
    //Check the cookies. If one of them has the name: userName and the value:
password
    //return false

       Cookie[] cookies=request.getCookies();
       if (cookies != null && cookies.length > 0)
           for(int i=0; i<cookies.length; i++) {
              Cookie c = cookies[i];
              if ((c.getName().equals(userName)) &&
(c.getValue().equals(password))){
                    return true;
              }
           }
    //return false; //Once this method is complete remove this line.
   return false;
  }
  private boolean isTheSame(HttpServletResponse response, String newPassword, String
reEnteredPassword) throws IOException{
    //Complete this method
    //Make sure the user entered the same new password twice. If both newPassword and
    //reEnteredPassword have the same string return true otherwise return false.
        if(newPassword.equals(reEnteredPassword)) {
             success(response); //I dont know how to put parameter in it, so just use
the default
             return true;

        }else
             return false;
    //return false; //Once this method is complete remove this line.

  }
  private void replace(HttpServletRequest request, HttpServletResponse response,
    String userName, String password, String newPassword){
    //Complete this method
    // This is the case that the program is sure there is a cookie with the name:
    // username and the value: password. Replace the value of this cookie by the
    // value of: newPassword.
        Cookie[] cookies=request.getCookies();
              if (cookies != null && cookies.length > 0) {
```

```java
                    for(int i=0; i<cookies.length; i++) {
                       Cookie c = cookies[i];
                       if(c.getName().equals(userName)) {
                                   password = newPassword;
                                   c.setValue(password);
                                   c.setMaxAge(365 * 24*60*60);
                                   response.addCookie(c);

                            }
                    }
                }
 }
  private void isNotInTheList(HttpServletResponse response, String userName, String
password) throws IOException{
    //Complete this method.
    //Make an html document to display the content of the console.
         PrintWriter out = response.getWriter();
         out.println("<html><body>\n");
         out.println("<font color = blue>");
         out.println("<h1>" + userName +  " with the password: " + password + " is
not in the list </h1>");
         out.println("</font>");
         out.println("</body></html>");
         out.close();
 }
  private void isNotTheSame(HttpServletResponse response, String newPassword,
    String reEnteredPassword) throws IOException{
    //Complete this method.
    //Make an html document to display the content of the console.
         PrintWriter out = response.getWriter();
         out.println("<html><body>\n");
         out.println("<font color = blue>");
         out.println("<h1>" + newPassword +  " must be the same as " +
reEnteredPassword + "</h1>");
         out.println("</body></html>");
         out.close();
  }
  private void success(HttpServletResponse response) throws IOException{
    //Complete this method.
    //Make an html document to display the content of the console.
         PrintWriter out = response.getWriter();
         out.println("<html><body>\n");
         out.println("<font color = blue>");
         out.println("<h1>The password is changed successfully.</h1>");
         out.println("</body></html>");
         out.close();
  }
  private void displayCookies(HttpServletRequest request){
    //This method is complete. Do not change it.
    Cookie[] cookies = request.getCookies();
    if (cookies != null && cookies.length > 0)
      for(int i=0; i<cookies.length; i++)
        System.out.println(cookies[i].getName() + ", " + cookies[i].getValue());
      System.out.println("*********************************************************");
  }
```

```
}
```

Example 1:

User enters a user name and password that has no cookie:



http://localhost:8080/Project/Main.html

User Name: abc

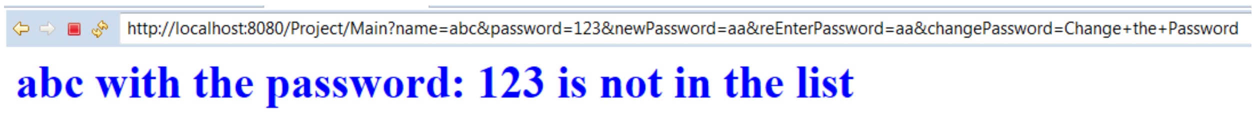
Password: 123

Changing Password

New Password: aa

Re enter The New Password: aa

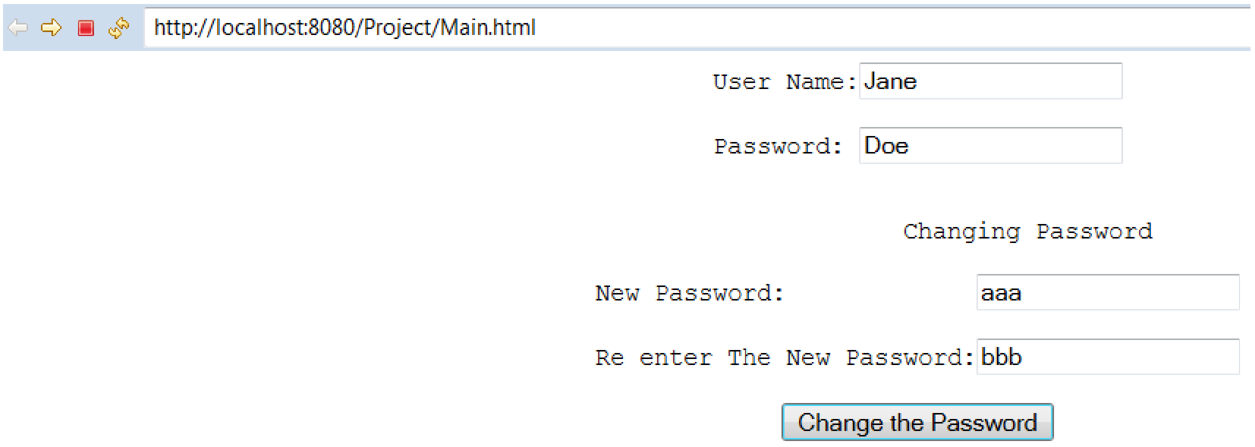
The response would be:



http://localhost:8080/Project/Main?name=abc&password=123&newPassword=aa&reEnterPassword=aa&changePassword=Change+the+Password

abc with the password: 123 is not in the list

Example 2: The new password and reentered password are not the same:



http://localhost:8080/Project/Main.html

User Name: Jane

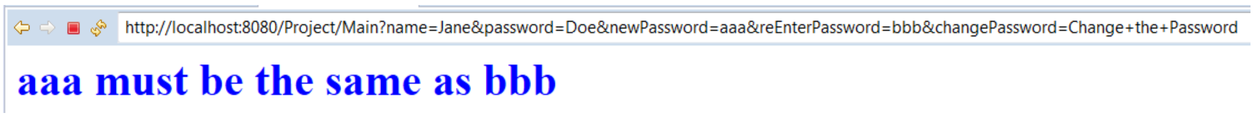
Password: Doe

Changing Password

New Password: aaa

Re enter The New Password: bbb

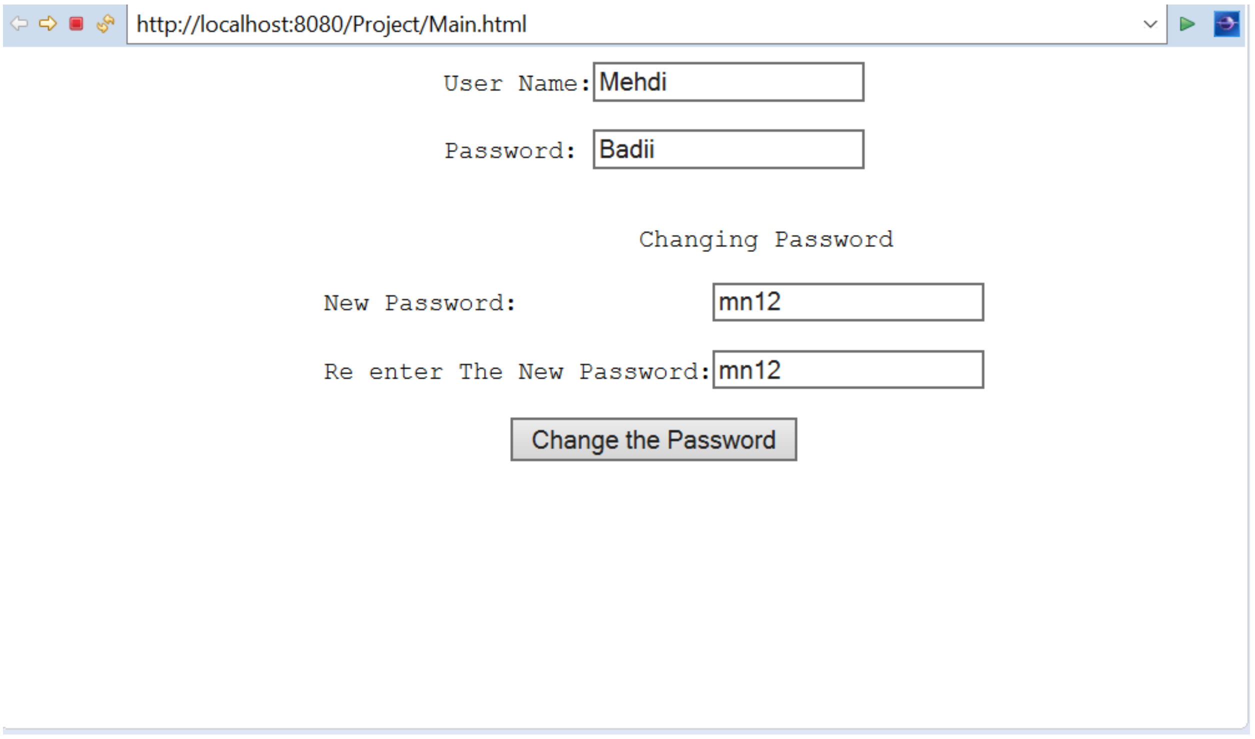
The response would be:



http://localhost:8080/Project/Main?name=Jane&password=Doe&newPassword=aaa&reEnterPassword=bbb&changePassword=Change+the+Password

aaa must be the same as bbb

Example 3: The user name and password has a cookie and want to change the password:



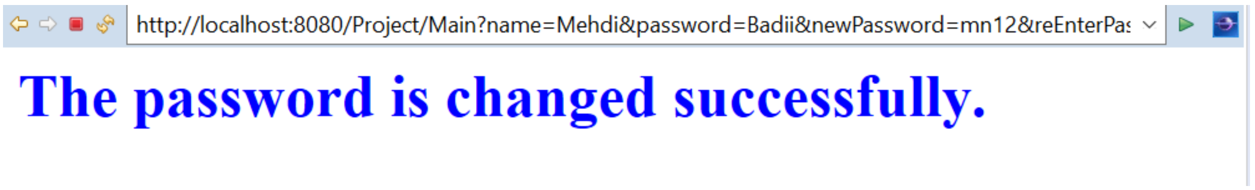
http://localhost:8080/Project/Main.html

User Name:	Mehdi
Password:	Badii

Changing Password

New Password:	mn12
Re enter The New Password:	mn12

The response would be:



http://localhost:8080/Project/Main?name=Mehdi&password=Badii&newPassword=mn12&reEnterPas

The password is changed successfully.

Answer:
```
import java.io.File;
import java.io.IOException;
import java.util.Scanner;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/AddCookies")
```

```java
public class AddCookies extends HttpServlet {
  private static final long serialVersionUID = 1L;
  //This servlet is complete. Do not change it.
  public AddCookies() {
    super();
  }

  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    addCookies(response);
    System.out.println("The program terminated with no error.");
  }

  private void addCookies(HttpServletResponse response){
    Scanner inputStream = null;
    try{
      inputStream = new Scanner(new File("C:\\Temp\\names.txt"));
       while(true){
        String name = inputStream.next();
        String value = inputStream.next();
        Cookie c = new Cookie(name, value);
        c.setMaxAge(365 * 24*60*60);
        response.addCookie(c);
      }
    }catch(Exception e){
    }
    inputStream.close();
  }
}
```

```html
<!-- The name of this html document must be: Main.html -->
<!-- Do not change this html document -->
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Q 4</title>
</head>
<body>
    <form action="http://localhost:8080/Project/Main">
    <center>
    <pre>
    User Name:<input name="name" type="text" value=""><br>
    Password: <input name="password" type="text" value=""><br><br>
                   Changing Password<br>
    New Password:              <input name="newPassword" type="text" value=""><br>
    Re enter The New Password:<input name="reEnterPassword" type="text" value=""><br>
    <input name="changePassword" type="Submit"    value="Change the Password">
    </pre>
    </center>
  </form>
</body>
</html>
```

```
/*
For the sake of simplicity assume the user enters strings in all
the input boxes.
*/
import java.io.*;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.Cookie;
import java.util.Scanner;
@WebServlet("/Main")
public class Main extends javax.servlet.http.HttpServlet implements
javax.servlet.Servlet {
  private String head, tail;
  public Main() {
    super();
    head = "<html><title>Not In The List</title><body><font color = blue><h1>";
    tail = "</h1></font></body></html>";
  }
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    //This method is complete. Do not change it.
    String userName = request.getParameter("name");
    String password = request.getParameter("password");
    String newPassword = request.getParameter("newPassword");
    String reEnteredPassword = request.getParameter("reEnterPassword");
    displayCookies(request);
    if(isIn(request, userName, password))
      if(isTheSame(newPassword, reEnteredPassword))
        replace(request, response, userName, password, newPassword);
      else
        isNotTheSame(response, newPassword, reEnteredPassword);
    else
      isNotInTheList(response, userName, password);

    displayCookies(request);

  }
  private boolean isIn(HttpServletRequest request, String userName, String password){
    //Complete this method.
    //Check the cookies. If one of them has the name: userName and the value:
password
    //return false
       Cookie[] cookies=request.getCookies();
       if (cookies != null && cookies.length > 0)
              for(int i=0; i<cookies.length; i++)
                     if ((cookies[i].getName().equals(userName)) &&
(cookies[i].getValue().equals(password)))
                            return true;

       return false;
  }
  private boolean isTheSame(String newPassword, String reEnteredPassword) throws
IOException{
    //Complete this method
    //Make sure the user entered the same new password twice. If both newPassword and
    //reEnteredPassword have the same string return true otherwise return false.
        if(newPassword.equals(reEnteredPassword))
            return true;
        else
            return false;

  }
  private void replace(HttpServletRequest request, HttpServletResponse response,
    String userName, String password, String newPassword){
    //Complete this method
    // This is the case that the program is sure there is a cookie with the name:
    // username and the value: password. Replace the value of this cookie by the
    // value of: newPassword.
        Cookie[] cookies=request.getCookies();
             if (cookies != null && cookies.length > 0) {
                   for(int i=0; i<cookies.length; i++) {
                 if(cookies[i].getName().equals(userName) &&
cookies[i].getValue().equals(password)) {
                        cookies[i].setValue(newPassword);
                        response.addCookie(cookies[i]);
                        System.out.println("Replaced password for
"+cookies[i].getName());
                 }
                   }
             }
             try{success(response);

             }catch(IOException e){
                   e.printStackTrace();
             }

 }
  private void isNotInTheList(HttpServletResponse response, String userName, String
password) throws IOException{
    //Complete this method.
    //Make an html document to display the content of the console.
        System.out.println(userName + " with the password: " + password + " is not
in the list");
        PrintWriter out = response.getWriter();
        out.println("<html><body>\n");
        out.println("<font color = blue>");
        out.println("<h1>" + userName +  " with the password: " + password + " is
not in the list </h1>");
        out.println("</font>");
        out.println("</body></html>");
        out.close();
 }
  private void isNotTheSame(HttpServletResponse response, String newPassword,
    String reEnteredPassword) throws IOException{
    //Complete this method.
    //Make an html document to display the content of the console.
        System.out.println(newPassword + " must be the same as " +
reEnteredPassword);
```

```
        PrintWriter out = response.getWriter();
        out.println("<html><body>\n");
        out.println("<font color = blue>");
        out.println("<h1>" + newPassword +  " must be the same as " +
reEnteredPassword + "</h1>");
        out.println("</body></html>");
        out.close();
  }
  private void success(HttpServletResponse response) throws IOException{
    //Complete this method.
    //Make an html document to display the content of the console.
        System.out.println("The password is changed successfully.");
        PrintWriter out = response.getWriter();
        out.println("<html><body>\n");
        out.println("<font color = blue>");
        out.println("<h1>The password is changed successfully.</h1>");
        out.println("</body></html>");
        out.close();
  }
  private void displayCookies(HttpServletRequest request){
    //This method is complete. Do not change it.
    Cookie[] cookies = request.getCookies();
    if (cookies != null && cookies.length > 0)
      for(int i=0; i<cookies.length; i++)
        System.out.println(cookies[i].getName() + ", " + cookies[i].getValue());
      System.out.println("*********************************************************");
  }
}
```

Question 4 End

Question 5(20 points): Complete the following client-server program to do the following:
a) On the server side:

- The program prompts the user to enter word: blue, green, or any other word.
- Then the program prompts the user to enter a message.
- The server program sends the color and the message to the client.

b) The client program must be an applet. On the client side when the program receives the message from the server:

- If the color is blue the program sets the background color of the applet to blue and writes the message on the applet.
- If the color is green the program sets the background color of the applet to green and writes the message on the applet.
- If the color is neither blue nor green the program does not change the color of the background and writes the message on the applet.

```
import java.applet.Applet;
import java.awt.*;
import java.io.*;
import java.net.*;

public class MyClient extends Applet {
  private int s;

  public void init(){
    //Complete this method.
    setSize(400, 400);
    try{
      InetAddress host = InetAddress.getByName("localhost");
      Socket clientSocket = new Socket(host, 16790);
      BufferedReader in = new BufferedReader(new
      InputStreamReader(clientSocket.getInputStream()));
      s = Integer.parseInt(in.readLine());
      //You may need repaint();
      repaint();
    }catch(Exception e){
      System.out.println("Error: " + e);
      System.exit(0);
    }
  }

  public void paint(Graphics page){
    //Complete this method.
    if(s == 1)
      page.drawRect(10, 10, 200, 100);
    else if(s == 2)
      page.drawOval(10, 10, 200, 200);
  }
}
```

Client:

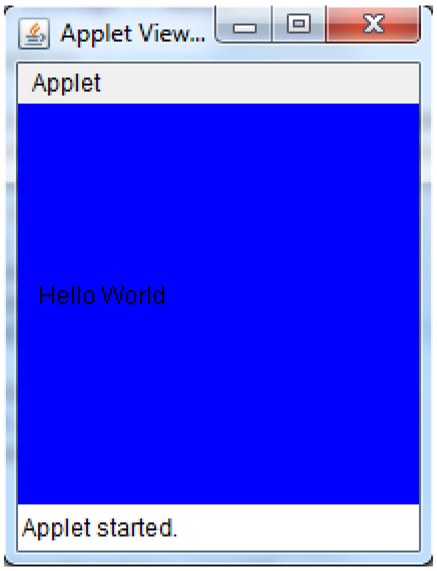
```
import java.io.*;
import java.net.*;
import java.util.Scanner;

public class MyServer {
  public static void main(String[] args){
    //Complete this method.
    ServerSocket serverSocket = null;
    Scanner keyboard = new Scanner(System.in);
    try {
      serverSocket = new ServerSocket(16790);
      Socket clientSocket = null;
      clientSocket = serverSocket.accept();
      System.out.print("Enter 1, or 2: ");
      int n = keyboard.nextInt();
      PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
      out.println("" + n);
      out.close();
      clientSocket.close();
      serverSocket.close();
    } catch (IOException e) {
      System.out.println("Error: " + e);
      System.exit(0);
    }
  }
}
```

Example 1:

Server Console:

```
Server is ready
Enter blue/ green for a color: blue
Enter a message: Hello World
```

Client applet:

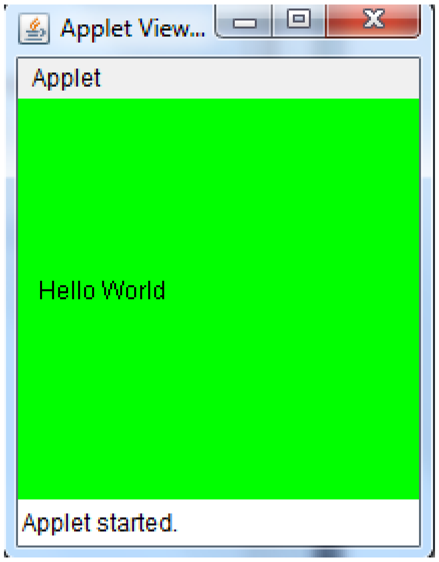


Example 2:

Server Console:

```
Server is ready
Enter blue/ green for a color: green
Enter a message: Hello World
```

Client applet:



Hello World

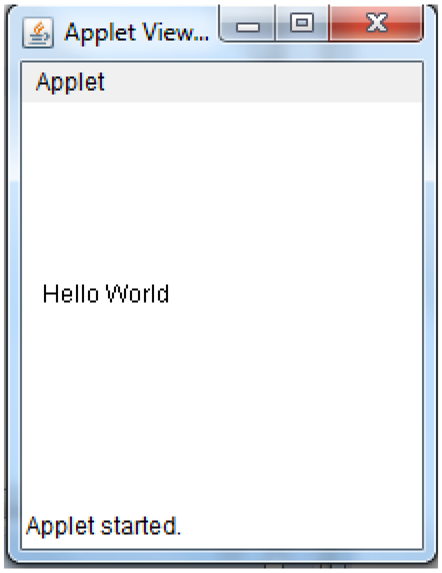
Applet started.

Example 3:

Server Console:

```
Server is ready
Enter blue/ green for a color: xyzt
Enter a message: Hello World
```

Client applet:



Hello World

Applet started.

Complete the following programs:

Answer:

Server:

```
import java.io.*;
import java.net.*;
import java.util.Scanner;

public class MyServer {
	public static void main(String[] args){
		// Complete this method.
		ServerSocket serverSocket = null;
		Scanner keyboard = new Scanner(System.in);

		try{
		serverSocket = new ServerSocket(16790);
		Socket clientSocket = null;
		clientSocket = serverSocket.accept();
		System.out.println("Server is ready");
		System.out.println("Enter blue/ green for a color:");
		String n = keyboard.nextLine();
		System.out.println("Enter a message:");
		String text = keyboard.nextLine();
		n = n + " " + text;

		PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
		out.println("" + n);
		out.close();
		clientSocket.close();
		serverSocket.close();
```

```
        keyboard.close();
        }catch(IOException e){
            System.out.println("Error: " + e);
            System.exit(0);
        }

    }
}
```

Client:

```
import java.applet.~~Applet~~;
import java.awt.Color;
import java.awt.Graphics;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.InetAddress;
import java.net.Socket;


public class MyClient extends Applet {
    private String backgroundColor;
    private String msg;
    private String[] textArr;

    public void init() {
        // Complete this method.
        setSize(400, 400);
        setBackground(Color.YELLOW);

        try {
            InetAddress host = InetAddress.getByName("localhost");
            Socket clientSocket = new Socket(host, 16790);
            BufferedReader in = new BufferedReader(new InputStreamReader(
                    clientSocket.getInputStream()));

            msg = in.readLine();
            textArr = msg.split(" ");

            backgroundColor = textArr[0];

            msg = "";

            for (int i = 1; i < textArr.length; i++) {

                msg = msg + " " + textArr[i];

            }

            repaint();

        } catch (Exception e) {
            System.out.println("Error: " + e);

        }

    }

    public void paint(Graphics g) {

        if ((backgroundColor.toLowerCase()).equals("green")) {
            g.setColor(Color.GREEN);
            g.fillRect(0, 0, 400, 400);
            g.setColor(Color.BLACK);
            g.drawString(msg, 150, 200);

        } else if ((backgroundColor.toLowerCase()).equals("blue")) {
            g.setColor(Color.BLUE);
            g.fillRect(0, 0, 400, 400);
            g.setColor(Color.BLACK);
            g.drawString(msg, 150, 200);

        } else {
            g.setColor(Color.WHITE);
            g.fillRect(0, 0, 400, 400);
            g.setColor(Color.BLACK);
            g.drawString(msg, 150, 200);

        }

    }

}
```

Question 5 End