Assignment 1:

Problem Statement: Implement multi-threaded client/server process communication using RMI

Codes:

Client.java

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



public class Client{
	public static void main(String[] args){
		Scanner sc = new Scanner(System.in);

		try{
			String serverURL = "rmi://localhost/Server";
			ServerIntf serverIntf = (ServerIntf)
Naming.lookup(serverURL);

			System.out.print("Enter First String: ");
			String str1 = sc.nextLine();

			System.out.print("Enter Second String: ");
			String str2 = sc.nextLine();



			System.out.println("--------------- Results ---------
------");
			System.out.println("Strings After Joining Is: " +
serverIntf.stringJoin(str1, str2));


		}catch(Exception e){
			System.out.println("Exception Occurred At Client!" +
e.getMessage());
		}

	}

}
```

Server.java

```
import java.rmi.*;

public class Server{

	public static void main(String[] args){

		try{
			ServerImpl serverImpl = new ServerImpl();
			Naming.rebind("Server", serverImpl);
```

```
                System.out.println("Server Started....");

            }catch(Exception e){
                System.out.println("Exception Occurred At Server!" +
e.getMessage());
            }
      }


}
```

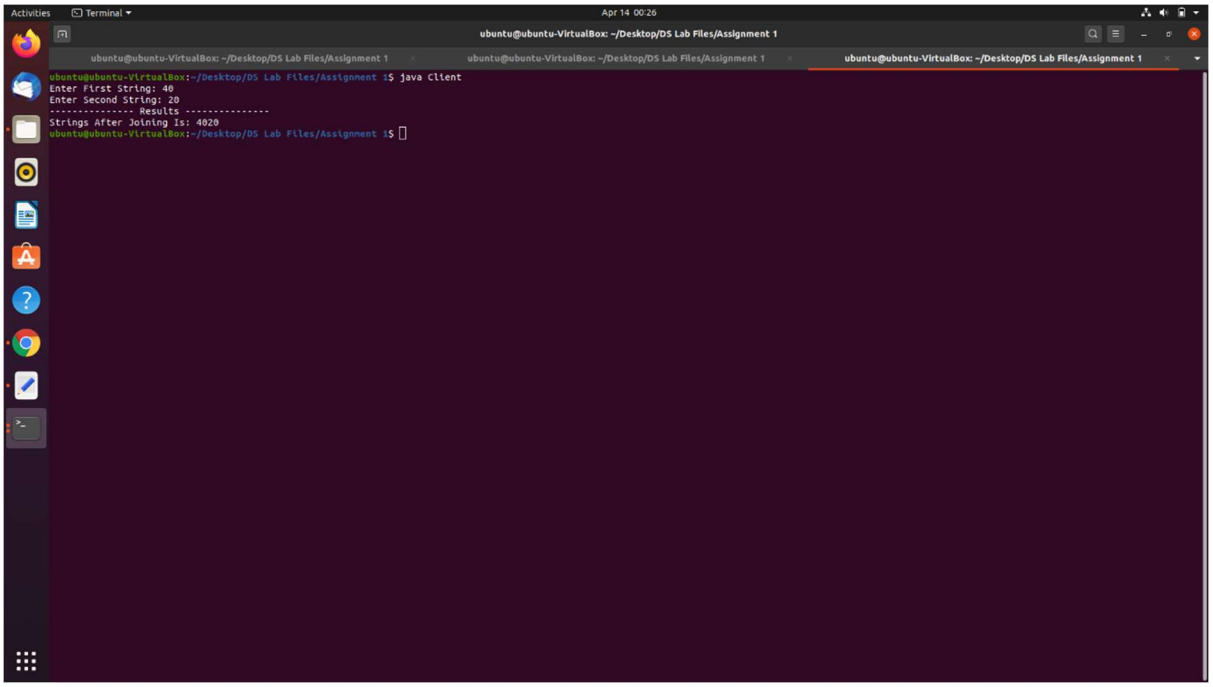
ServerImpl.java

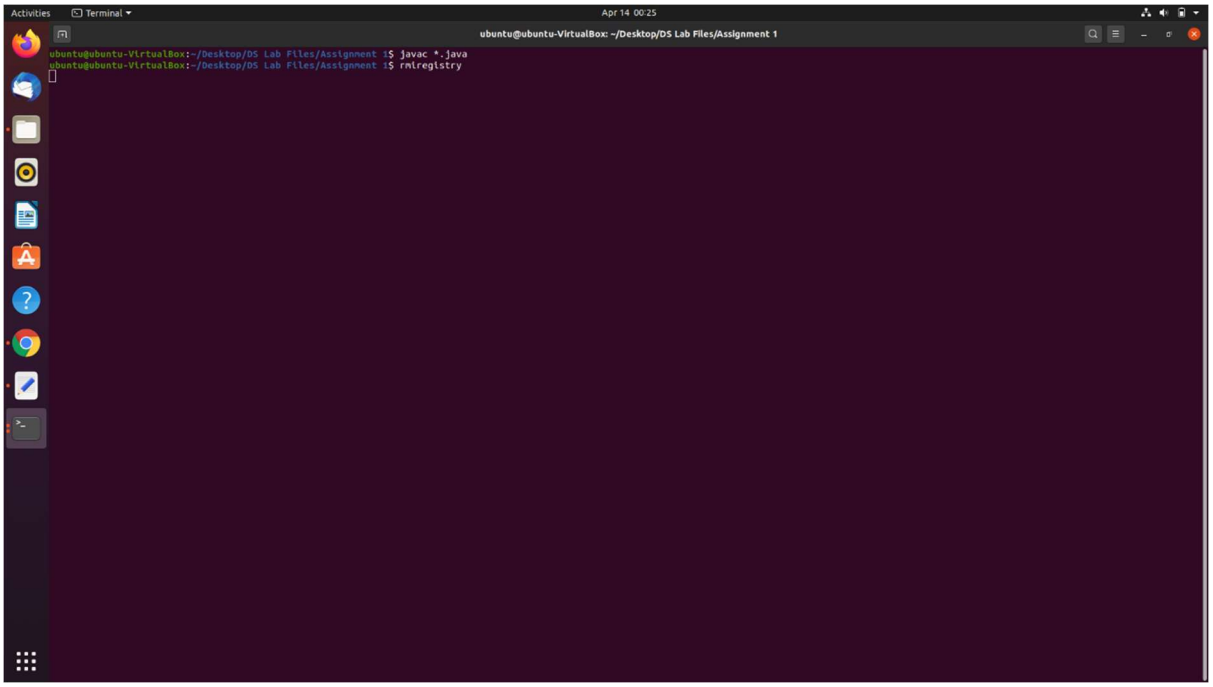
```
import java.rmi.*;
import java.rmi.server.*;

public class ServerImpl extends UnicastRemoteObject
      implements ServerIntf {

            public ServerImpl() throws RemoteException{

            }
            public String stringJoin(String str1, String str2) throws
RemoteException{
                String result = str1 + str2;

                return result;
            }
}
```

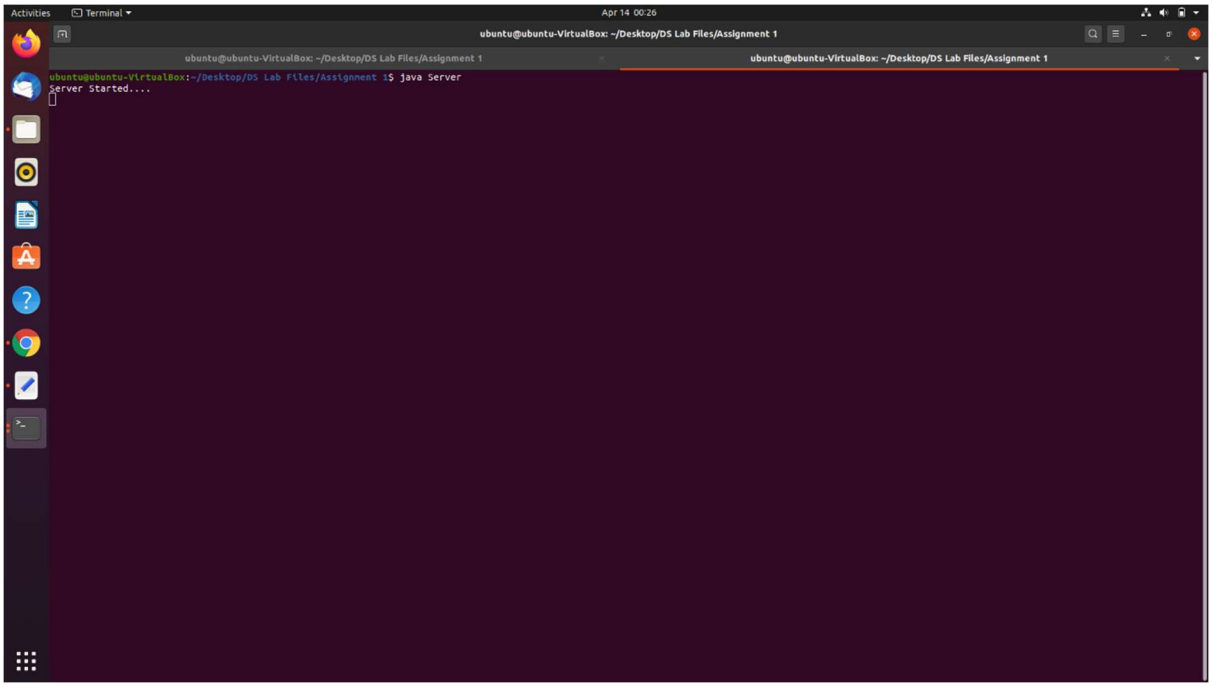
ServerIntf.java

```
import java.rmi.*;


interface ServerIntf extends Remote{
      // Syntax for method declaration: access_specifier return_type
method_name(arguments...){ return value}
      public String stringJoin(String str1, String str2) throws
RemoteException;

}
```

Output:


Activities
Terminal
Apr 14 00:26
ubuntu@ubuntu-VirtualBox: ~/Desktop/DS Lab Files/Assignment 1
ubuntu@ubuntu-VirtualBox:~/Desktop/DS Lab Files/Assignment 1$ java Client
Enter First String: 40
Enter Second String: 20
---------- Results ----------
Strings After Joining Is: 4020
ubuntu@ubuntu-VirtualBox:~/Desktop/DS Lab Files/Assignment 1$


Activities
Terminal
Apr 14 00:25
ubuntu@ubuntu-VirtualBox: ~/Desktop/DS Lab Files/Assignment 1
ubuntu@ubuntu-VirtualBox:~/Desktop/DS Lab Files/Assignment 1$ javac *.java
ubuntu@ubuntu-VirtualBox:~/Desktop/DS Lab Files/Assignment 1$ rmiregistry


Activities
Terminal
Apr 14 00:26
ubuntu@ubuntu-VirtualBox: ~/Desktop/DS Lab Files/Assignment 1
ubuntu@ubuntu-VirtualBox: ~/Desktop/DS Lab Files/Assignment 1
ubuntu@ubuntu-VirtualBox: ~/Desktop/DS Lab Files/Assignment 1
ubuntu@ubuntu-VirtualBox:~/Desktop/DS Lab Files/Assignment 1$ java Server
Server Started....

Assignment 2:

Problem Statement: Develop any distributed application using CORBA to demonstrate objectbrokering (Calculator).

Codes:

ReverseModule.idl:

```
module ReverseModule{

      interface Reverse

      {

            string reverse_string(in string str);
      }

}
```

ReverseServer.java

```
import ReverseModule.Reverse;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;
import org.omg.CORBA.*;
import org.omg.PortableServer.*;

class ReverseServer
{
    public static void main(String[] args)
    {
        try
        {
            // initialize the ORB
            org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);

            // initialize the BOA/POA
            POA rootPOA =
POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
            rootPOA.the_POAManager().activate();

            // creating the calculator object
            ReverseImpl rvr = new ReverseImpl();

            // get the object reference from the servant class
            org.omg.CORBA.Object ref =
rootPOA.servant_to_reference(rvr);

            System.out.println("Step1");
            Reverse h_ref = ReverseModule.ReverseHelper.narrow(ref);
            System.out.println("Step2");

            org.omg.CORBA.Object objRef =
orb.resolve_initial_references("NameService");

            System.out.println("Step3");
            NamingContextExt ncRef =
NamingContextExtHelper.narrow(objRef);
            System.out.println("Step4");
```

```
            String name = "Reverse";
            NameComponent path[] = ncRef.to_name(name);
            ncRef.rebind(path,h_ref);

            System.out.println("Reverse Server reading and
waiting....");
            orb.run();
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

ServerImpl.java

```
import ReverseModule.ReversePOA;
import java.lang.String;
class ReverseImpl extends ReversePOA
{
    ReverseImpl()
    {
        super();
        System.out.println("Reverse Object Created");
    }

    public String reverse_string(String name)
    {
    StringBuffer str=new StringBuffer(name);
    str.reverse();
        return (("Server Send "+str));
    }
}
```

ReverseClient.java

```
import ReverseModule.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;
import org.omg.CORBA.*;
import java.io.*;

class ReverseClient
{

    public static void main(String args[])
    {
        Reverse ReverseImpl=null;
```

```
        try
        {
            // initialize the ORB
            org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);

            org.omg.CORBA.Object objRef =
orb.resolve_initial_references("NameService");
            NamingContextExt ncRef =
NamingContextExtHelper.narrow(objRef);

            String name = "Reverse";
            ReverseImpl =
ReverseHelper.narrow(ncRef.resolve_str(name));

            System.out.println("Enter String=");
            BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
            String str= br.readLine();

            String tempStr= ReverseImpl.reverse_string(str);

            System.out.println(tempStr);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

Output:

```
gaurav@gaurav-VirtualBox:~/Downloads/DS Lab Files/Assignment 2$ idlj -fall Reve
rseModule.idl
gaurav@gaurav-VirtualBox:~/Downloads/DS Lab Files/Assignment 2$ javac *.java Re
verseModule/*.java
ReverseModule/_ReverseStub.java:46: warning: IORCheckImpl is internal proprieta
ry API and may be removed in a future release
     com.sun.corba.se.impl.orbutil.IORCheckImpl.check(str, "ReverseModule._Reve
rseStub");
                                   ^
Note: ReverseModule/ReversePOA.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
1 warning
gaurav@gaurav-VirtualBox:~/Downloads/DS Lab Files/Assignment 2$ orbd -ORBInitia
lPort 1050&
[1] 6796
gaurav@gaurav-VirtualBox:~/Downloads/DS Lab Files/Assignment 2$ java ReverseSer
ver -ORBInitialPort 1050& -ORBInitialHost localhost&
[2] 7076
[3] 7077
gaurav@gaurav-VirtualBox:~/Downloads/DS Lab Files/Assignment 2$ -ORBInitialHost
: command not found
Reverse Object Created
Step1
Step2
Step3
Step4
Reverse Server reading and waiting....
```

```
gaurav@gaurav-VirtualBox:~/Downloads/DS Lab Files/Assignment 2$ java ReverseCli
ent -ORBInitialPort 1050 -ORBInitialHost localhost
Enter String=
Department Of Information Technology
Server Send ygolonhceT noitamrofnI fO tnemtrapeD
gaurav@gaurav-VirtualBox:~/Downloads/DS Lab Files/Assignment 2$
```

Assignment 3:

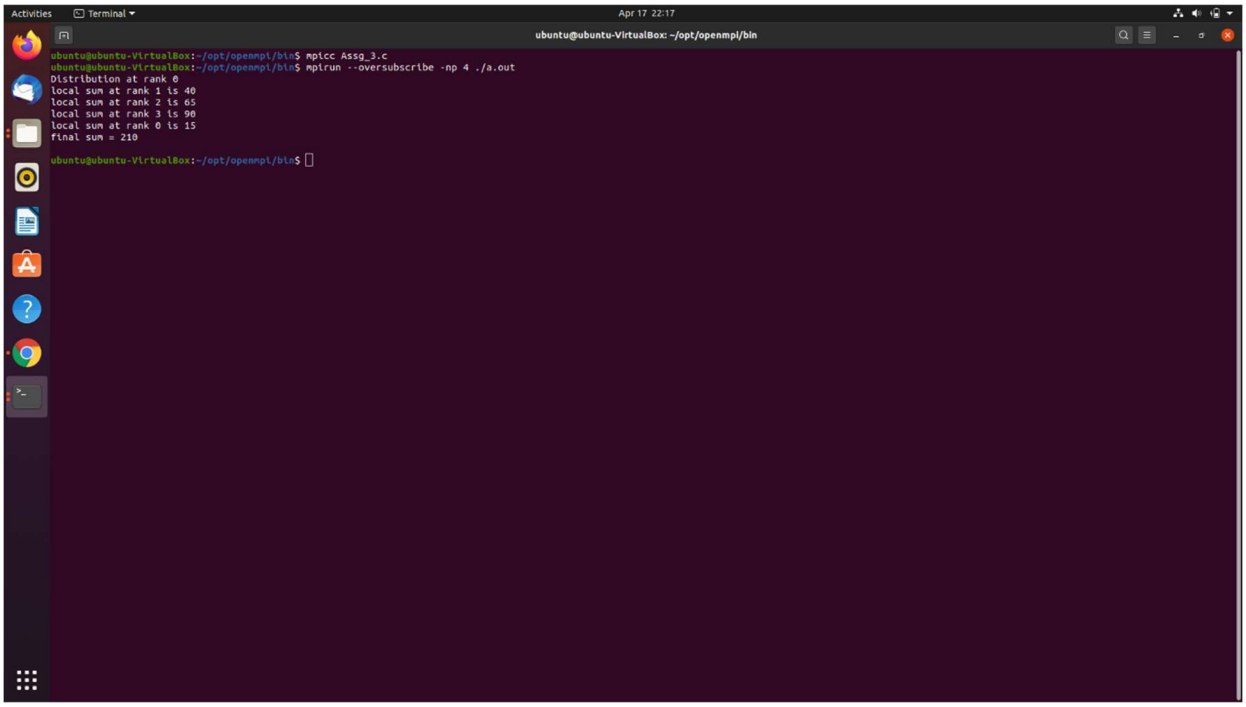
Problem Statement: Develop a distributed system, to find sum of N elements in an array by distributing N/n elements to n number of processors MPI or OpenMP. Demonstrate by displaying theintermediate sums calculated at different processors.

Codes:

Ass3.c:

```
#include <stdio.h>
#include <mpi.h>
int main(int argc, char* argv[])
{
	int rank, size;
	int num[20]; //N=20, n=4
	MPI_Init(&argc, &argv);
	MPI_Comm_rank(MPI_COMM_WORLD, &rank);
	MPI_Comm_size(MPI_COMM_WORLD, &size);
	for(int i = 0;i < 20;i++)
		num[i] = i + 1;
	if(rank == 0){
		int s[4];
		printf("Distribution at rank %d \n", rank);
		for(int i = 1;i < 4;i++)
			MPI_Send(&num[i * 5], 5, MPI_INT, i, 1,
MPI_COMM_WORLD); //N/n i.e. 20/4=5
		int sum = 0, local_sum = 0;
		for(int i = 0;i < 5;i++)
		{
			local_sum = local_sum + num[i];
		}
		for(int i = 1;i < 4;i++)
		{
			MPI_Recv(&s[i], 1, MPI_INT, i, 1, MPI_COMM_WORLD,
MPI_STATUS_IGNORE);
		}
		printf("local sum at rank %d is %d\n", rank,local_sum);
		sum=local_sum;
		for(int i = 1;i < 4;i++)
			sum = sum + s[i];
		printf("final sum = %d\n\n",sum);
	}
	else
	{
		int k[5];
		MPI_Recv(k, 5, MPI_INT, 0, 1, MPI_COMM_WORLD,
MPI_STATUS_IGNORE);
		int local_sum = 0;
		for(int i = 0;i < 5;i++)
		{
			local_sum = local_sum + k[i];
		}
		printf("local sum at rank %d is %d\n", rank, local_sum);
		MPI_Send(&local_sum, 1, MPI_INT, 0, 1, MPI_COMM_WORLD);
	}
	MPI_Finalize();
	return 0;
}
```

Output:


Activities
Terminal
Apr 17 22:17
ubuntu@ubuntu-VirtualBox: ~/opt/openmpi/bin
ubuntu@ubuntu-VirtualBox:~/opt/openmpi/bin$ mpicc Assg_3.c
ubuntu@ubuntu-VirtualBox:~/opt/openmpi/bin$ mpirun --oversubscribe -np 4 ./a.out
Distribution at rank 0
local sum at rank 1 is 40
local sum at rank 2 is 65
local sum at rank 3 is 90
local sum at rank 0 is 15
final sum = 210
ubuntu@ubuntu-VirtualBox:~/opt/openmpi/bin$

Assignment 4:

Problem Statement: Implement Berkeley algorithm for clock

synchronization.Codes:

Server.py:

```
# Python3 program imitating a clock server

from functools import reduce
from dateutil import parser
import threading
import datetime
import socket
import time
# datastructure used to store client address and clock data
client_data = {}
''' nested thread function used to receive
      clock time from a connected client '''
def startReceivingClockTime(connector, address):
      while True:
            # receive clock time
            clock_time_string = connector.recv(1024).decode()
            clock_time = parser.parse(clock_time_string)
            clock_time_diff = datetime.datetime.now() - \

      clock_time

            client_data[address] = {
                              "clock_time"      : clock_time,
                              "time_difference" : clock_time_diff,
                              "connector"  : connector
                              }

            print("Client Data updated with: "+ str(address),
                                                                end =
"\n\n")
            time.sleep(5)



''' master thread function used to open portal for
      accepting clients over given port '''
def startConnecting(master_server):

      # fetch clock time at slaves / clients
      while True:
            # accepting a client / slave clock client
            master_slave_connector, addr = master_server.accept()
            slave_address = str(addr[0]) + ":" + str(addr[1])

            print(slave_address + " got connected successfully")

            current_thread = threading.Thread(
                                    target = startReceivingClockTime,
```

```
                                    args = (master_slave_connector,
                                                          slave_address,
))
            current_thread.start()



# subroutine function used to fetch average clock difference
def getAverageClockDiff():

      current_client_data = client_data.copy()

      time_difference_list = list(client['time_difference']
                                              for client_addr, client
                                                    in
client_data.items())


      sum_of_clock_difference = sum(time_difference_list, \
                                              datetime.timedelta(0, 0))

      average_clock_difference = sum_of_clock_difference \
                                                          /
len(client_data)

      return average_clock_difference


''' master sync thread function used to generate
      cycles of clock synchronization in the network '''
def synchronizeAllClocks():

      while True:

            print("New synchronization cycle started.")
            print("Number of clients to be synchronized: " + \

      str(len(client_data)))

            if len(client_data) > 0:

                  average_clock_difference = getAverageClockDiff()

                  for client_addr, client in client_data.items():
                        try:
                              synchronized_time = \
                                    datetime.datetime.now() + \

      average_clock_difference

                              client['connector'].send(str(
                                          synchronized_time).encode())

                        except Exception as e:
                              print("Something went wrong while " + \
```

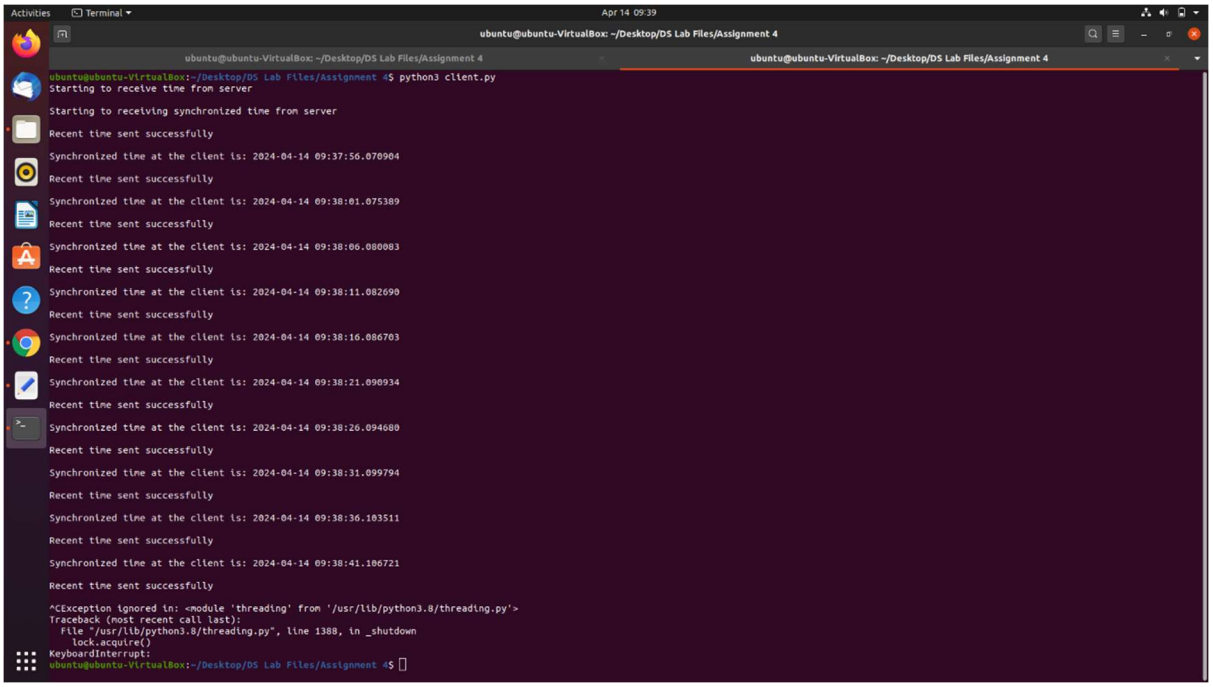
```
                                "sending synchronized time " + \
                                "through " + str(client_addr))

            else :
                  print("No client data." + \
                                " Synchronization not applicable.")

            print("\n\n")

            time.sleep(5)


# function used to initiate the Clock Server / Master Node
def initiateClockServer(port = 8080):

      master_server = socket.socket()
      master_server.setsockopt(socket.SOL_SOCKET,
                                          socket.SO_REUSEADDR, 1)

      print("Socket at master node created successfully\n")

      master_server.bind(('', port))

      # Start listening to requests
      master_server.listen(10)
      print("Clock server started...\n")

      # start  making  connections
      print("Starting to make connections...\n")
      master_thread = threading.Thread(
                                target = startConnecting,
                                args = (master_server, ))
      master_thread.start()

      # start synchronization
      print("Starting synchronization parallelly...\n")
      sync_thread = threading.Thread(
                                target = synchronizeAllClocks,
                                args = ())
      sync_thread.start()



# Driver function
if___name__ == '__main__':

      # Trigger the Clock Server
      initiateClockServer(port = 8080)
```

Client.py:

```
# Python3 program imitating a client process

from timeit import default_timer as timer

from dateutil import parser
import threading
import datetime
import socket
import time


# client thread function used to send time at client side
def startSendingTime(slave_client):

      while True:
            # provide server with clock time at the client
            slave_client.send(str(datetime.datetime.now())
            .encode())

     print("Recent time sent successfully",time.sleep(5)end = "\n\n")


# client thread function used to receive synchronized timedef
startReceivingTime(slave_client):

      while True:
            # receive data from the server
            Synchronized_time = parser.parse(
                                    slave_client.recv(1024).decode())

            print("Synchronized time at the client is: " + \

      str(Synchronized_time),
                                                  end = "\n\n")


# function used to Synchronize client process time
def initiateSlaveClient(port = 8080):

      slave_client = socket.socket()

      # connect to the clock server on local computer
      slave_client.connect(('127.0.0.1', port))

      # start sending time to server
      print("Starting to receive time from server\n")
      send_time_thread = threading.Thread(
                              target = startSendingTime,
                              args = (slave_client, ))
      send_time_thread.start()
```

```
        # start receiving synchronized from server
        print("Starting to receiving " + \
                                    "synchronized time from server\n")
        receive_time_thread = threading.Thread(
                                target = startReceivingTime,
                                args = (slave_client, ))
        receive_time_thread.start()


# Driver function
if___name__ == '__main__':

        # initialize the Slave / Client
        initiateSlaveClient(port = 8080)
```

Output:



```
New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

127.0.0.1:47618 got connected successfully
Client Data updated with: 127.0.0.1:47618

New synchronization cycle started.
Number of clients to be synchronized: 1

Client Data updated with: 127.0.0.1:47618

New synchronization cycle started.
Number of clients to be synchronized: 1

Client Data updated with: 127.0.0.1:47618

New synchronization cycle started.
Number of clients to be synchronized: 1

Client Data updated with: 127.0.0.1:47618

New synchronization cycle started.
Number of clients to be synchronized: 1

Client Data updated with: 127.0.0.1:47618

New synchronization cycle started.
Number of clients to be synchronized: 1
```

```
ubuntu@ubuntu-VirtualBox:~/Desktop/DS Lab Files/Assignment 4$ python3 server.py
Socket at master node created successfully

Clock server started...

Starting to make connections...

Starting synchronization parallelly...

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.

New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.
```

Assignment 5:

Problem Statement: Implement token ring based mutual exclusion

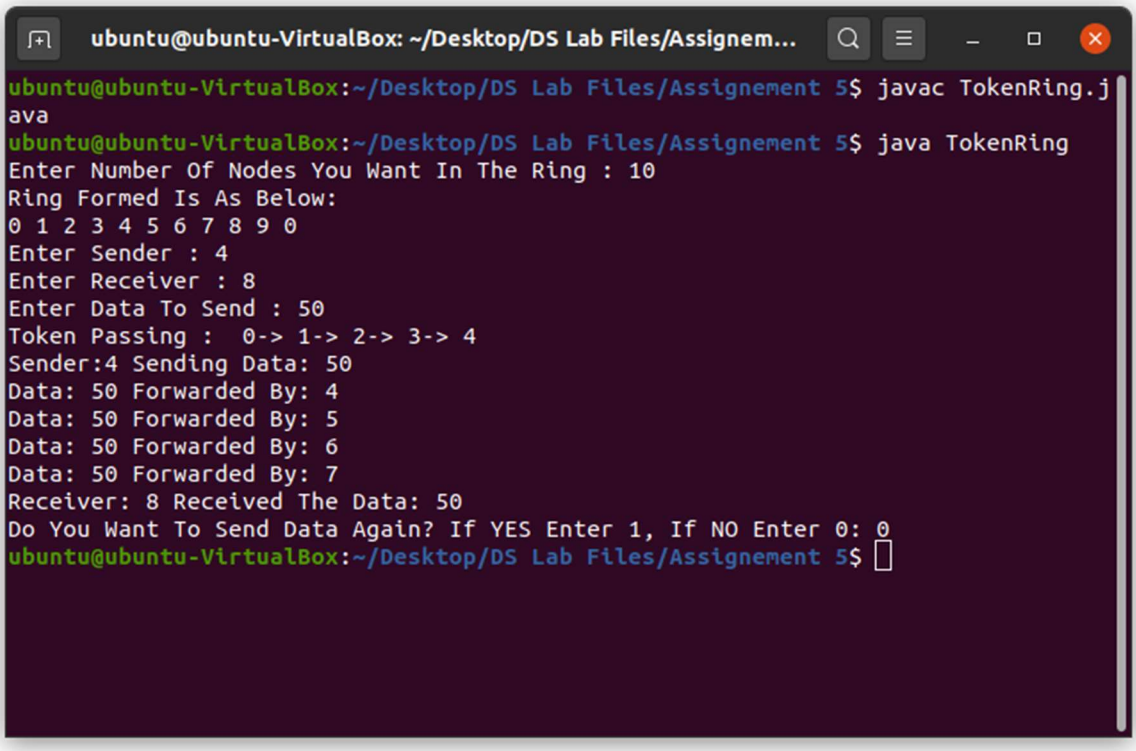
algorithm.Codes:

TokenRing.java:

```
import java.util.*;



public class TokenRing{

	public static void main(String[] args){
		Scanner sc = new Scanner(System.in);

		System.out.print("Enter no. of nodes you want in the
ring: ");
		int n = sc.nextInt();

		System.out.println("Ring Formed is as below: ");
		for(int i=0; i<n; i++){
			System.out.print(i + " ");
		}

		System.out.println("0");

		int choice = 0;

		do{
			System.out.print("Enter Sender: ");
			int sender = sc.nextInt();

			System.out.print("Enter Receiver: ");
			int receiver = sc.nextInt();

			System.out.print("Enter Data to Send: ");
			int data = sc.nextInt();

			int token = 0;
			System.out.println("Token Passing: ");


			for(int i=token; i<sender; i++){
				System.out.print(" " + i + "->");

			}

			System.out.println(" " + sender);
			System.out.println("Sender: " + sender + "
Sending Data: " + data);
			for(int i=sender; i!=receiver; i = (i+1)%n){
				System.out.println("Data: " + data + "
Forwaded by: " + i);
			}
```

```
                        System.out.println("Receiver: " + receiver +
"Received the data: " + data);
                        token = sender;

                        System.out.print("Do you want to send data
                        again?
If yes Enter 1, If no Enter 0: ");
                        choice = sc.nextInt();
                }while(choice == 1);
        }
}
```

Output:



Assignment 6:

Problem Statement: Implement Bully and Ring algorithm for leader election.Codes:

BullyAlgoExample.java:

```
import java.io.*;
import java.util.Scanner;
// create class BullyAlgoExample to understand how bully
algorithms works
class BullyAlgoExample{
// declare variables and arrays for process and their
status
static int numberOfProcess;
static int priorities[] = new int[100];
static int status[] = new int[100];
static int cord;
// main() method start
public static void main(String args[])throws IOException
// handle IOException
{
// get input from the user for the number of processes
System.out.println("Enter total number of processes:");
// create scanner class object to get input from user
Scanner sc = new Scanner(System.in);
numberOfProcess = sc.nextInt();
int i;
// use for loop to set priority and status of each
process
for(i = 0; i<numberOfProcess; i++)
{
System.out.println("Status for process "+(i+1)+":");
status[i] = sc.nextInt();
System.out.println("Priority of process "+(i+1)+":");
priorities[i] = sc.nextInt();
}
```

```
System.out.println("Enter proces which will initiate
election");
int ele = sc.nextInt();
sc.close();
// call electProcess() method
electProcess(ele);
System.out.println("After electing process the final
coordinator is "+cord);
}
// create electProcess() method
static void electProcess(int ele)

{
ele = ele - 1;
cord = ele + 1;
for(int i = 0; i<numberOfProcess; i++)
{
if(priorities[ele]<priorities[i])
{
System.out.println("Election message is sent from
"+(ele+1)+" to "+(i+1));
if(status[i]==1)
electProcess(i+1);
}
}
}
}
```

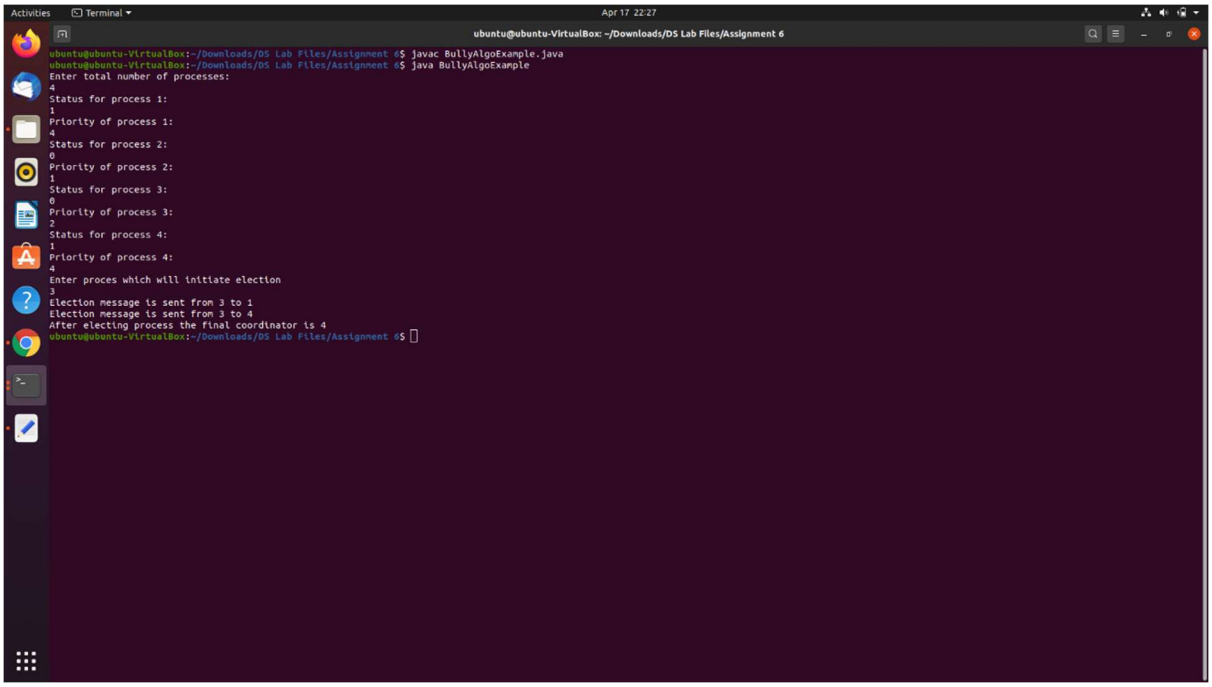
RingAlgorithm.java:

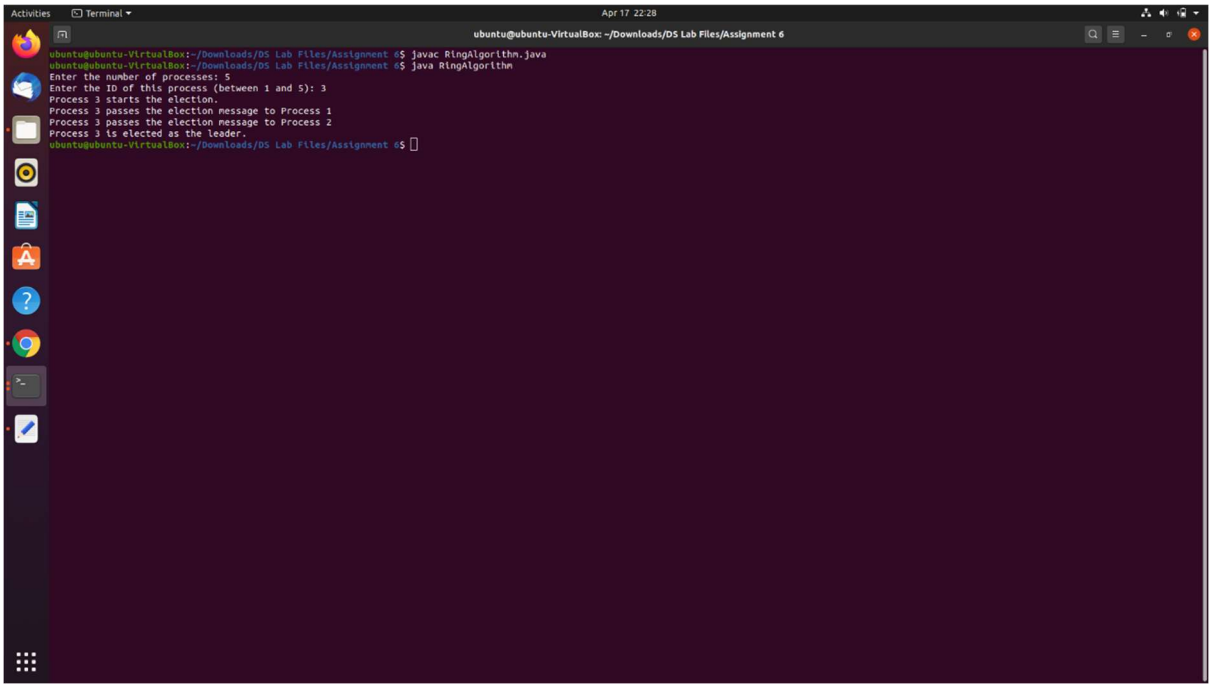
```
import java.util.Scanner;
public class RingAlgorithm {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.print("Enter the number of processes: ");
int numProcesses = scanner.nextInt();
```

```
System.out.print("Enter the ID of this process (between
1 and " + numProcesses + "): ");
int thisProcessId = scanner.nextInt();
// Initialize the ring
RingProcess[] ring = new RingProcess[numProcesses];
for (int i = 0; i < numProcesses; i++) {
ring[i] = new RingProcess(i + 1);
}
// Set the next process in the ring for each process
for (int i = 0; i < numProcesses; i++) {
ring[i].setNextProcess(ring[(i + 1) % numProcesses]);
}
// Start the election
ring[thisProcessId - 1].startElection();
}
}
class RingProcess {
private int processId;
private RingProcess nextProcess;
private boolean isLeader;
public RingProcess(int processId) {
this.processId = processId;
this.isLeader = false;
}
public void setNextProcess(RingProcess nextProcess) {
this.nextProcess = nextProcess;
}
public void startElection() {
System.out.println("Process " + processId + " starts the
election.");
if (isLeader) {
System.out.println("Process " + processId + " is already
the leader.");
return;
}
```

```
RingProcess currentProcess = this;
while (true) {
if (currentProcess.nextProcess.processId == processId) {
currentProcess.isLeader = true;

System.out.println("Process " + processId + " is elected
as the leader.");
break;
} else if (currentProcess.nextProcess.processId >
processId) {
currentProcess = currentProcess.nextProcess;
} else {
System.out.println("Process " + processId + " passes the
election message to Process " +
currentProcess.nextProcess.processId);
currentProcess = currentProcess.nextProcess;
}
}
}
}
```

Output:


Activities
Terminal
Apr 17 22:27
ubuntu@ubuntu-VirtualBox: ~/Downloads/DS Lab Files/Assignment 6
ubuntu@ubuntu-VirtualBox:~/Downloads/DS Lab Files/Assignment 6$ javac BullyAlgoExample.java
ubuntu@ubuntu-VirtualBox:~/Downloads/DS Lab Files/Assignment 6$ java BullyAlgoExample
Enter total number of processes:
4
Status for process 1:
1
Priority of process 1:
4
Status for process 2:
0
Priority of process 2:
1
Status for process 3:
0
Priority of process 3:
2
Status for process 4:
1
Priority of process 4:
4
Enter proces which will initiate election
3
Election message is sent from 3 to 1
Election message is sent from 3 to 4
After electing process the final coordinator is 4
ubuntu@ubuntu-VirtualBox:~/Downloads/DS Lab Files/Assignment 6$


Activities
Terminal
Apr 17 22:28
ubuntu@ubuntu-VirtualBox: ~/Downloads/DS Lab Files/Assignment 6
ubuntu@ubuntu-VirtualBox:~/Downloads/DS Lab Files/Assignment 6$ javac RingAlgorithm.java
ubuntu@ubuntu-VirtualBox:~/Downloads/DS Lab Files/Assignment 6$ java RingAlgorithm
Enter the number of processes: 5
Enter the ID of this process (between 1 and 5): 3
Process 3 starts the election.
Process 3 passes the election message to Process 1
Process 3 passes the election message to Process 2
Process 3 is elected as the leader.
ubuntu@ubuntu-VirtualBox:~/Downloads/DS Lab Files/Assignment 6$

Assignment 7:

Problem Statement: Create a simple web service and write any distributed application to consumethe web service.

Codes:

Index.html

```
<!DOCTYPE html>

<html>
    <head>
        <title>Calculator Web Service </title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    </head>
    <body>
        <form action="CalculatorServlet">
            Enter Number-1:<input type="text" name="number1"
value=""/><br>
            Enter Number-2:<input type="text" name="number2"
value=""/><br>

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

        </form>
    </body>
</html>
```

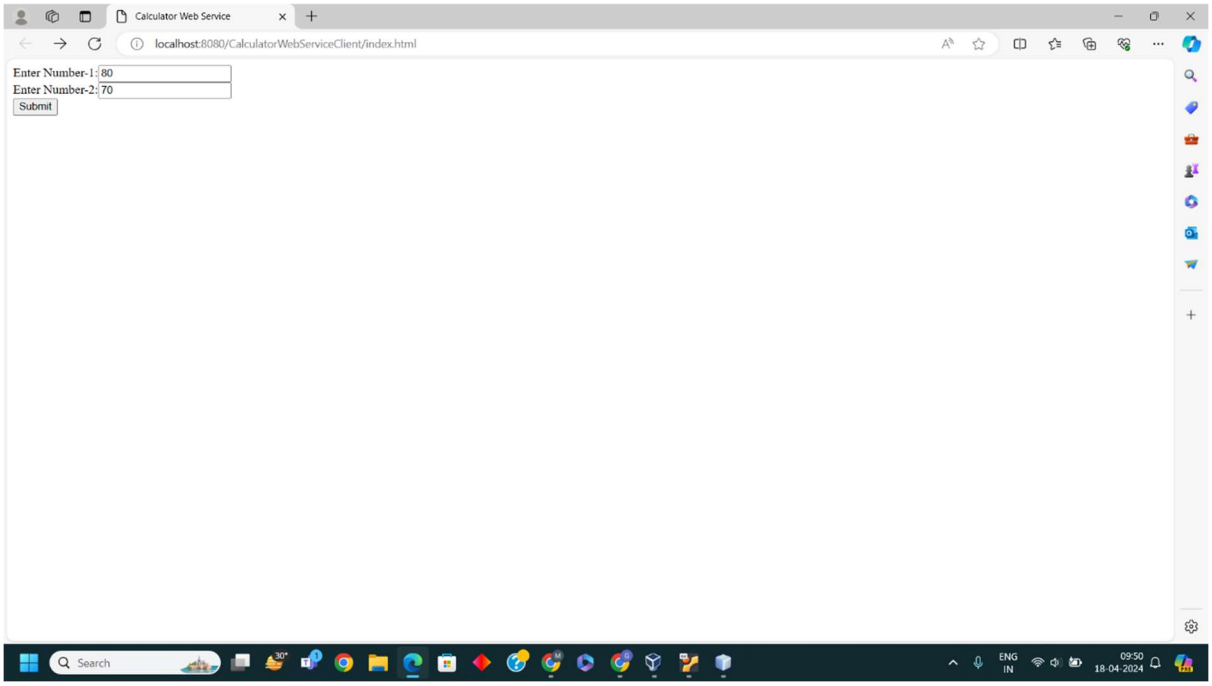
CalculatorServlet.java

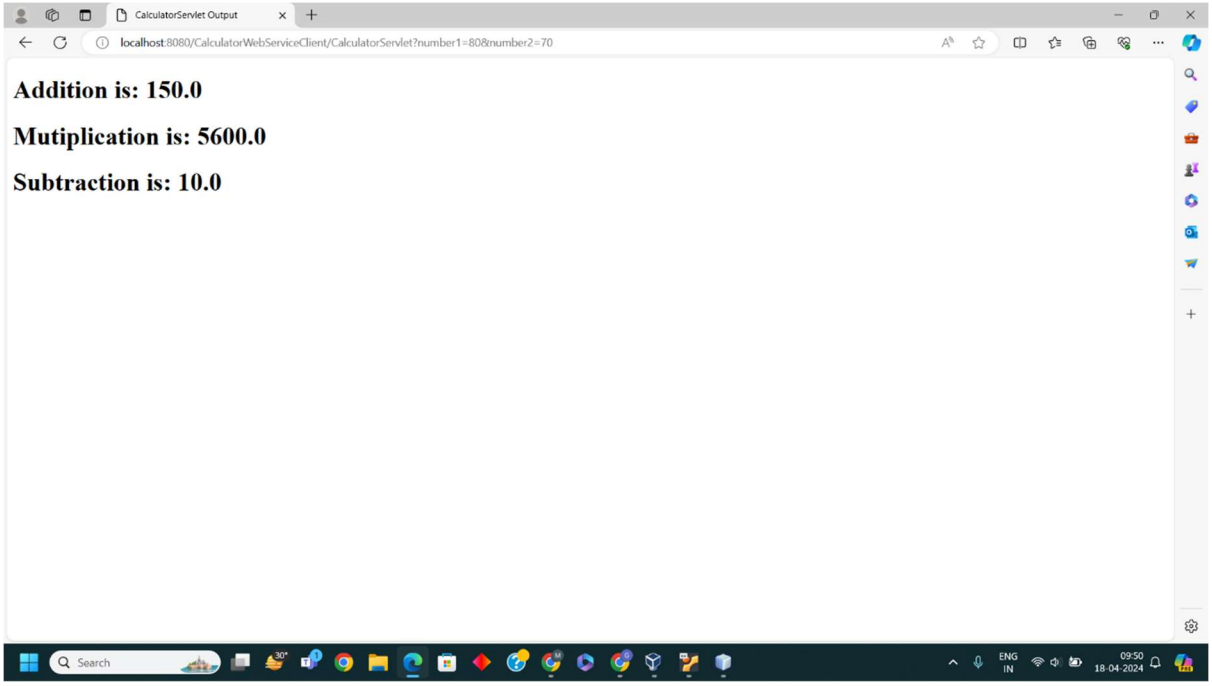
```
import com.myservice.MyCalculatorWebService_Service;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.xml.ws.WebServiceRef;


public class CalculatorServlet extends HttpServlet {

    @WebServiceRef(wsdlLocation = "WEB-INF/wsdl/localhost_8080/Lab-
7/MyCalculatorWebService.wsdl")
    private MyCalculatorWebService_Service service;

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code>
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
```

```
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {

            double num1,num2;

            num1 = Double.parseDouble(request.getParameter("number1"));
            num2 = Double.parseDouble(request.getParameter("number2"));
            /* TODO output your page here. You may use following sample
code. */
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>CalculatorServlet Output</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1> Addition is: " + addition(num1,num2) +
"</h1>");
            out.println("<h1> Mutiplication is: " +
multiplication(num1,num2) + "</h1>");
            out.println("<h1> Subtraction is: " +
subtraction(num1,num2) + "</h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods.
Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
```

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

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

    private double addition(double num1, double num2) {
        // Note that the injected javax.xml.ws.Service reference as
well as port objects are not thread safe.
        // If the calling of port operations may lead to race condition
some synchronization is required.
        com.myservice.MyCalculatorWebService port =
service.getMyCalculatorWebServicePort();
        return port.addition(num1, num2);
    }

    private double multiplication(double num1, double num2) {
        // Note that the injected javax.xml.ws.Service reference as
well as port objects are not thread safe.
        // If the calling of port operations may lead to race condition
some synchronization is required.
        com.myservice.MyCalculatorWebService port =
service.getMyCalculatorWebServicePort();
        return port.multiplication(num1, num2);
    }

    private double subtraction(double num1, double num2) {
        // Note that the injected javax.xml.ws.Service reference as
well as port objects are not thread safe.
        // If the calling of port operations may lead to race condition
some synchronization is required.
        com.myservice.MyCalculatorWebService port =
service.getMyCalculatorWebServicePort();
        return port.subtraction(num1, num2);
    }



}
```

Output:


Calculator Web Service
localhost:8080/CalculatorWebServiceClient/index.html
Enter Number-1: 80
Enter Number-2: 70
Submit


CalculatorServlet Output
localhost:8080/CalculatorWebServiceClient/CalculatorServlet?number1=80&number2=70
Addition is: 150.0
Mutiplication is: 5600.0
Subtraction is: 10.0