ASSIGNMENT 1

Name- Tanmay Borse

Roll No- 10

Class- BE(IT)

AddClient.java

```
import java.rmi.*;
import java.net.*;
import java.io.*;
import java.util.*;
public class AddClient
{public static void main(String args[]){
    String host="localhost";
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter 1st Parameter");
    int a =sc.nextInt();
    System.out.println("Enter 2nd Parameter");
    int b = sc.nextInt();
    try{AddRem remobj =(AddRem)Naming.lookup("rmi://"+host+"/AddRem");
      System.out.println(remobj.addNum(a,b)); }
    catch(RemoteException re)
    {re.printStackTrace();}
    catch(NotBoundException nbe){
      nbe.printStackTrace();}
    catch(MalformedURLException mfe)
    {mfe.printStackTrace(); }}}
```

AddRem.java

```
import java.rmi.*;
public interface AddRem extends Remote
{public int addNum(int a, int b) throws RemoteException;}
```

AddRemImpl.java

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

```
public class AddRemImpl extends UnicastRemoteObject implements AddRem {
   public AddRemImpl() throws RemoteException {
      // Constructor implementation
   }@Override
   public int addNum(int a, int b) throws RemoteException {
      // Method implementation
      return a + b;}}
```

AddServer.java

```
import java.rmi.*;
import java.net.*;
public class AddServer
{public static void main(String[] args) {
      try{AddRemImpl localobj = new AddRemImpl();
         Naming.rebind("rmi:///AddRem",localobj);}
      catch(RemoteException re)
      {re.printStackTrace();}
      catch(MalformedURLException mfe)
      {mfe.printStackTrace(); }   }}
```

Output:

```
C:\Windows\System32\cmd.e

Microsoft Windows [Version 10.0.22621.1555]
(c) Microsoft Corporation. All rights reserved.

D:\Distributed System Codes\All Programs\All Programs\1>javac Client.java

D:\Distributed System Codes\All Programs\All Programs\1>java Client
Enter message: Hello World
Enter message:
```

```
C:\Windows\System32\cmd.e

Microsoft Windows [Version 10.0.22621.1555]
(c) Microsoft Corporation. All rights reserved.

D:\Distributed System Codes\All Programs\All Programs\1>javac Server.java

D:\Distributed System Codes\All Programs\All Programs\1>java Server
Server ready
Received message: Hello World
```

ASSIGNMENT 2

Name- Tanmay Borse

Roll No- 10

Class- BE(IT)

ReverseModule.idl

```
module ReverseModule //module ReverseModule is the name of the module
{interface Reverse{
	string reverse_string(in string str); }; };
```

ReverseImpl.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";
            //Helper class provides narrow method that cast corba object reference (ref) into the java
            // System.out.println("Step2");
            // Look ups "Reverse" in the naming context
            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();}}}
```

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 portable object adaptor (BOA/POA) connects client request using object reference
        //uses orb method as resolve_initial_references
        POA rootPOA = POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
        rootPOA.the_POAManager().activate();
        // creating an object of ReverseImpl class
        ReverseImpl rvr = new ReverseImpl();
        //server consist of 2 classes ,servent and server. The servent is the subclass of ReversePOA which is
generated by the idlj compiler
        // The servent ReverseImpl is the implementation of the ReverseModule idl interface
```

```
        // get the object reference from the servant class
        //use root POA class and its method servant_to_reference
        org.omg.CORBA.Object ref = rootPOA.servant_to_reference(rvr);
        // System.out.println("Step1");
        Reverse h_ref = ReverseModule.ReverseHelper.narrow(ref);// Helper class provides narrow method that
cast corba object reference (ref) into the java interface
        // System.out.println("Step2");
        // orb layer uses resolve_initial_references method to take initial reference as NameService
        org.omg.CORBA.Object objRef = orb.resolve_initial_references("NameService");
        //Register new object in the naming context under the Reverse
        // 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);
        //Server run and waits for invocations of the new object from the client
        System.out.println("Reverse Server reading and waiting....");
        orb.run();}
      catch(Exception e){
        e.printStackTrace(); }}}
```

ASSIGNMENT 3

Name- Tanmay Borse

Roll No- 10

Class- BE(IT)

arr_sum_mpi.c

```
#include<stdio.h>
#include<mpi.h>
#define arr_size 15
int main(int argc, char *argv[]){
   int rank, size;
   MPI_Init(&argc, &argv);
   MPI_Comm_rank(MPI_COMM_WORLD, &rank);
   MPI_Comm_size(MPI_COMM_WORLD, &size);
   //Code that will execute inside process 0 or rank 0
   if(rank == 0){
      int arr[]= {12,4,6,3,21,15,3,5,7,8,9,1,5,3,5};
      int global_sum = 0, local_sum = 0, recv_local_sum;
      //If the array size is perfectly divisible by number of process.
      if(arr_size%size == 0){
         int array_element_per_process = arr_size/size;
         int sub_arr[array_element_per_process];
         for(int i=1; i<size; i++){
            //Copying the sub array
            for(int j=0; j<array_element_per_process;j++){
               sub_arr[j] = arr[i*array_element_per_process+j];}
            //Sending array chunk of equal size to all the process.
            MPI_Send(sub_arr, array_element_per_process, MPI_INT, i, 1, MPI_COMM_WORLD);
            MPI_Send(&array_element_per_process, 1, MPI_INT, i, 1, MPI_COMM_WORLD);}
         //Calculating the local sum of rank 0 itself
         for(int j=0; j<array_element_per_process; j++){
```

```
            local_sum += arr[j];}
        printf("Rank %d: local sum: %d\n", rank, local_sum);
        global_sum += local_sum;
    //When the array size is not perfectly divisible by number of process.
    }else{
        int array_element_per_process = arr_size/size + 1;
        int sub_arr[array_element_per_process];
        for(int i=1; i<size; i++){
            if(i == size - 1){
                //last sub array will have the size less than other process array size
                int total_array_size_of_last_process = arr_size - array_element_per_process * i;
                for(int j=0; j< total_array_size_of_last_process; j++){
                    sub_arr[j] = arr[i*array_element_per_process+j];}
                MPI_Send(&sub_arr, total_array_size_of_last_process, MPI_INT, i, 1,
MPI_COMM_WORLD);
                MPI_Send(&total_array_size_of_last_process, 1, MPI_INT, i, 1, MPI_COMM_WORLD);
            }else{
                //Copying the sub array
                for(int j=0; j<array_element_per_process;j++){
                    sub_arr[j] = arr[i*array_element_per_process+j];}
                MPI_Send(&sub_arr, array_element_per_process, MPI_INT, i, 1, MPI_COMM_WORLD);
                MPI_Send(&array_element_per_process, 1, MPI_INT, i, 1, MPI_COMM_WORLD}
Calculating the local sum of rank 0 itself
        for(int j=0; j<array_element_per_process; j++){
            local_sum += arr[j];}
        printf("Rank %d: local sum: %d\n", rank, local_sum);
        global_sum += local_sum;
    }//calculating the global sum of the array
    //Receving the local sum from the other process and updating the global sum
    for(int i=1; i<size; i++){
        MPI_Recv(&recv_local_sum, 1, MPI_INT, i, 1, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
```

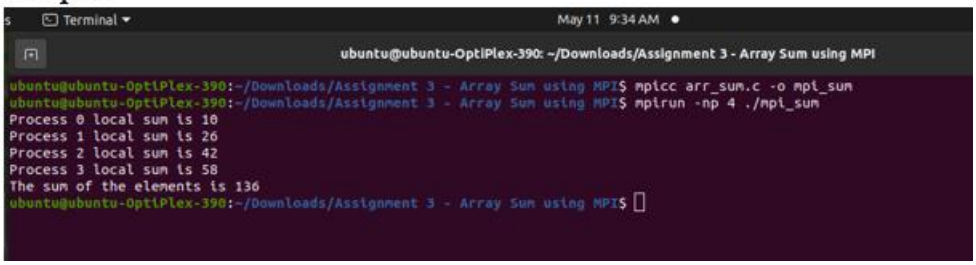
```
        global_sum += recv_local_sum;
      }//Printing the output
      printf("The sum of the array is %d\n", global_sum);
   //Code that will get executed inside other than process 0 or rank 0.
   }else{//The other process will receive the chunck of array
      int array_element_per_process = arr_size/size + 1;
      int recv_sub_arr[array_element_per_process];
      int recv_array_element_per_process, local_sum = 0;
      MPI_Recv(recv_sub_arr, recv_array_element_per_process, MPI_INT, 0, 1, MPI_COMM_WORLD,
MPI_STATUS_IGNORE);
      MPI_Recv(&recv_array_element_per_process, 1, MPI_INT, 0, 1, MPI_COMM_WORLD,
MPI_STATUS_IGNORE);
      //Calculating local sum for the sub array
      for(int j=0; j<recv_array_element_per_process; j++){
         local_sum += recv_sub_arr[j];}
      //Printing the local sum
      printf("Rank %d: local sum: %d\n", rank, local_sum);
      //Sending back the local sum to the rank 0 or process 0.
      MPI_Send(&local_sum, 1, MPI_INT, 0, 1, MPI_COMM_WORLD);}
   MPI_Finalize();
   return 0;}
```

arr_sum.c

```
#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>
#define ARRAY_SIZE 16
int main(int argc, char** argv) {
  int rank, size;
  int sum = 0;
  int array[ARRAY_SIZE];
  // Initialize MPI
```

```
 MPI_Init(&argc, &argv);

 MPI_Comm_rank(MPI_COMM_WORLD, &rank);

 MPI_Comm_size(MPI_COMM_WORLD, &size);

// Populate the array on the root process

 if (rank == 0) {

   for (int i = 0; i < ARRAY_SIZE; i++) {

     array[i] = i + 1;}}

// Scatter the array to all processes

 int subarray_size = ARRAY_SIZE / size;

 int subarray[subarray_size];

 MPI_Scatter(array, subarray_size, MPI_INT, subarray, subarray_size, MPI_INT, 0,
MPI_COMM_WORLD);

// Sum the local elements

 int local_sum = 0;

 for (int i = 0; i < subarray_size; i++) {

   local_sum += subarray[i];}

// Display the local sum of each process

 printf("Process %d local sum is %d\n", rank, local_sum);

 // Reduce the local sums to get the final sum on the root process

 MPI_Reduce(&local_sum, &sum, 1, MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD);

 // Print the result on the root process

 if (rank == 0) {

   printf("The sum of the elements is %d\n", sum);}

 // Finalize MPI

 MPI_Finalize();

 return 0;}
```

Output-



ASSIGNMENT 4

Name- Tanmay Borse

Roll No- 10

Class- BE(IT)

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",
              end="\n\n")
        time.sleep(5)
```

```
# client thread function used to receive synchronized time
def 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()
if __name__ == '__main__':
initiateSlaveClient(port=8080)
```

ASSIGNMENT 5

Name- Tanmay Borse

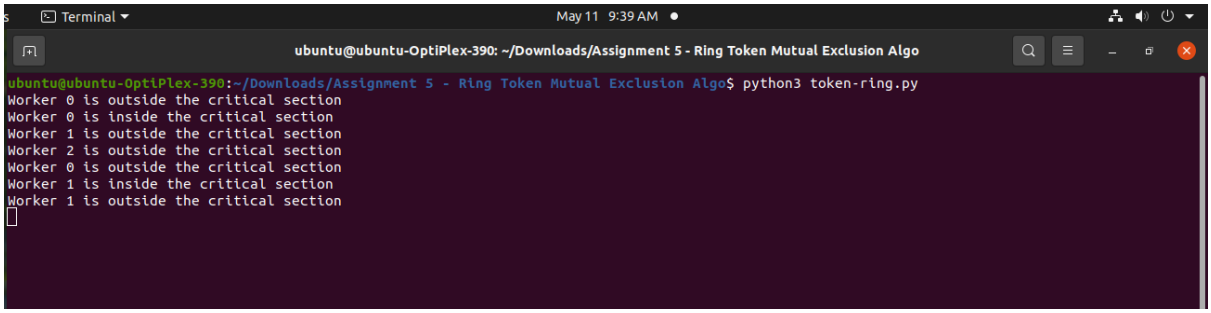
Roll No- 10

Class- BE(IT)

Ring-token.c

```
#include<stdio.h>
#include<conio.h>
#include<dos.h>
#include<time.h>
void main(){
   int  cs=0,pro=0;
   double run=5;
   char key='a';
   time_t t1,t2;
   printf("Press a key(except q) to enter a process into critical section.");
   printf(" \nPress q at any time to exit.");
   t1 = time(NULL) - 5;
   while(key!='q')
   {while(!kbhit())
      if(cs!=0)
      {t2 = time(NULL);
         if(t2-t1 > run)
         {printf("Process%d ",pro-1);
            printf(" exits critical section.\n");
            cs=0; }}
      key = getch();
      if(key!='q')
      {if(cs!=0)
         printf("Error: Another process is currently executing critical section Please wait till its n");
         else printf("Process %d ",pro);
            printf(" entered critical section\n");
            cs=1;
            pro++;
            t1 = time(NULL); }}}}
```

Output-


Terminal
May 11 9:39 AM
ubuntu@ubuntu-OptiPlex-390: ~/Downloads/Assignment 5 - Ring Token Mutual Exclusion Algo
ubuntu@ubuntu-OptiPlex-390:~/Downloads/Assignment 5 - Ring Token Mutual Exclusion Algo$ python3 token-ring.py
Worker 0 is outside the critical section
Worker 0 is inside the critical section
Worker 1 is outside the critical section
Worker 2 is outside the critical section
Worker 0 is outside the critical section
Worker 1 is inside the critical section
Worker 1 is outside the critical section

ASSIGNMENT 6

Name- Tanmay Borse

Roll No- 10

Class- BE(IT)

Bully_ring.cpp

```
// first we include the necessary header files
#include <iostream>
#include <cstdlib>
// we define MAX as the maximum number of processes our program can simulate
// we declare array pStatus[MAX] to store the process status; 0 for dead and 1 for alive
// we declare n as the number of processes
// we declare coordinator to store the winner of election
int pStatus[MAX], n, coordinator;
using namespace std;
void bully();
void ring();
// void ring_();   // this is also another approach ring implementation, and works well.
void display();
int main()
{int i, j, fchoice;
   cout << "Enter number of processes: ";
   cin >> n;
   for (i = 1; i <= n; i++)
   {cout << "Enter Process " << i << " is alive or not(0/1): ";
      cin >> pStatus[i];
      if (pStatus[i])
         coordinator = i;}
   display();
   do{cout << "-----------------------------------------------";
      cout << "\n1.BULLY ALGORITHM\n2.RING\n3.DISPLAY\n4.EXIT\n";
      cout << "-----------------------------------------------\n\n";
```

```
        cout << "Enter your choice: ";
        cin >> fchoice;
        switch (fchoice){
        case 1:
            bully();
            break;
        case 2:
            ring();
            // ring_()
            break;
        case 3:
            display();
            break;
        case 4:
            exit(1);
            break;}
    } while (fchoice != 3);
    return 0;}
void display()
{ int i;
    // we display the processes, their status and the coordinator
    cout << "-----------------------------------------------\n";
    cout << "Processes:  ";
    for (i = 1; i <= n; i++) // PID from 1 to n
        cout << i << "\t";
    cout << endl
         << "Alive:      ";
    for (i = 1; i <= n; i++)
        cout << pStatus[i] << "\t";
    cout << "\n-----------------------------------------------\n";
    cout << "COORDINATOR IS " << coordinator << endl;}
void bully()
{int schoice, crash, activate, i, gid, flag, subcoordinator;
```

```
do
{cout << "-----------------------------------------------";
   cout << "\n1.CRASH\n2.ACTIVATE\n3.DISPLAY\n4.EXIT\n";
   cout << "-----------------------------------------------\n";
   cout << "Enter your choice: ";
   cin >> schoice;
   switch (schoice){
   case 1:
      // we manually crash the process to see if our implementation
      // can elect another coordinator
      cout << "Enter process to crash: ";
      cin >> crash;
      // if the process is alive then set its status to dead
      if (pStatus[crash])
         pStatus[crash] = 0;
      else
         cout << "Process " << crash << " is already dead!" << endl;
      do{cout << "Enter election generator id: ";
         cin >> gid;
         if (gid == coordinator || pStatus[gid] == 0)
            cout << "Please, enter a valid generator id.." << endl;
      } while (gid == coordinator || pStatus[gid] == 0);
      flag = 0;
      // if the coordinator has crashed then we need to find another coordinator
      if (crash == coordinator)
      {// the election generator process will send the message to all higher process
         for (i = gid + 1; i <= n; i++)
         {cout << "Message  is sent from " << gid << " to " << i << endl;
            // if the higher process is alive then it will respond
            if (pStatus[i])
            {subcoordinator = i;
               cout << "Response is sent from " << i << " to " << gid << endl;
               flag = 1;}}
```

```
        // the highest responding process is selected as the coordinator
        if (flag == 1)
            coordinator = subcoordinator;
        // else if no higher process are alive then the election generator process
        // is selected as coordinator
        else
            coordinator = gid;}
    display();
    break;
case 2:
    // enter process to revive
    cout << "Enter Process ID to be activated: ";
    cin >> activate;
    // if the entered process was dead then it is revived
    if (!pStatus[activate])
    {pStatus[activate] = 1;}
    else
    {cout << "Process " << activate << " is already alive!" << endl;
        break;}
    if (activate == n)
    {coordinator = n;
        break; }
    flag = 0;
    // else, the activated process sends message to all higher process
    for (i = activate + 1; i <= n; i++)
    {cout << "Message is sent from " << activate << " to " << i << endl;
        // if higher process is active then it responds
        if (pStatus[i])
        {subcoordinator = i;
            cout << "Response is sent from " << i << " to " << activate << endl;
            flag = 1;}}
    // the highest responding process is made the coordinator
    if (flag == 1)
```

```
            coordinator = subcoordinator;
          // if no higher process respond then the activated process is coordinator
          else
            coordinator = activate;
          display();
          break;
        case 3:
          display();
          break;
        case 4:
          break;}
    } while (schoice != 4);}
// ring algorithm implementation
void ring()
{int tchoice, crash, activate, gid, subcoordinator, i;
    do
    {cout << "-----------------------------------------------";
        cout << "\n1.CRASH\n2.ACTIVATE\n3.DISPLAY\n4.EXIT\n";
        cout << "-----------------------------------------------\n\n";
        cout << "Enter your choice: ";
        cin >> tchoice;
        switch (tchoice)
        {case 1:
          cout << "\nEnter Process ID to crash : ";
          cin >> crash;
          if (pStatus[crash])
            pStatus[crash] = 0;
          else
            cout << "Process " << crash << " is already dead!" << endl;
          do
          {cout << "Enter election generator id: ";
            cin >> gid;
            if (gid == coordinator)
```

```
                cout << "Please, enter a valid generator id.." << endl;
            } while (gid == coordinator);
 if (crash == coordinator)
            {subcoordinator = 1;
              for (i = 0; i < (n + 1); i++)
              {int pid = (i + gid) % (n + 1);
                if (pid != 0) // since process id starts from 1 (to n)
                {if (pStatus[pid] && subcoordinator < pid)
                  {subcoordinator = pid; }
                  cout << "Election message sent from " << pid << ": #Msg" << subcoordinator << endl; }}}
coordinator = subcoordinator;}
            display();
            break;
        case 2:
            cout << "Enter Process ID to be activated: ";
            cin >> activate;
            if (!pStatus[activate])
              pStatus[activate] = 1;
            else{cout << "Process " << activate << " is already alive!" << endl;
              break;}
 subcoordinator = activate;
            for (i = 0; i < n + 1; i++)
            {int pid = (i + activate) % (n + 1);
              if (pid != 0)
              {if (pStatus[pid] && subcoordinator < pid)
                {subcoordinator = pid; }
                cout << "Election message passed from " << pid << ": #Msg" << subcoordinator << endl;}}
            coordinator = subcoordinator;
        case 3:
            display();
            break;
        default:
            break; } while (tchoice != 4);
```

Output-

```
ubuntu@ubuntu-OptiPlex-390:~/Downloads/Assignment 6 - Bully And Ring Election Algorithm$ python3 bully_ring.py
Enter number of processes: 4
Enter Process  1  is alive or not(0/1):
1
Enter Process  2  is alive or not(0/1):
1
Enter Process  3  is alive or not(0/1):
0
Enter Process  4  is alive or not(0/1):
0
--------------------------------------------
PROCESS:  1     2       3       4
ALIVE:    1     1       0       0
--------------------------------------------
COORDINATOR IS 2
--------------------------------------------
1.BULLY ALGORITHM
2.RING ALGORITHM
3.DISPLAY
4.EXIT
--------------------------------------------

Enter your choice: 1
--------------------------------------------
1.CRASH
2.ACTIVATE
3.DISPLAY
4.EXIT
--------------------------------------------

Enter your choice: 1
Enter process to crash: 2
Enter election generator id: 1
Message is  sent from 1  to 2
Message is  sent from 1  to 3
Message is  sent from 1  to 4
--------------------------------------------
PROCESS:  1     2       3       4
```

```
3.DISPLAY
4.EXIT
--------------------------------------------

Enter your choice: 1
Enter process to crash: 2
Enter election generator id: 1
Message is  sent from 1  to 2
Message is  sent from 1  to 3
Message is  sent from 1  to 4
--------------------------------------------
PROCESS:  1     2       3       4
ALIVE:    1     0       0       0
--------------------------------------------
COORDINATOR IS 1
--------------------------------------------
1.CRASH
2.ACTIVATE
3.DISPLAY
4.EXIT
--------------------------------------------

Enter your choice: 2
Enter Process ID to be activated: 3
Message is  sent from 3 to 4
--------------------------------------------
PROCESS:  1     2       3       4
ALIVE:    1     0       1       0
--------------------------------------------
COORDINATOR IS 3
--------------------------------------------
1.CRASH
2.ACTIVATE
3.DISPLAY
4.EXIT
--------------------------------------------

Enter your choice:
```

ASSIGNMENT 7

Name- Tanmay Borse

Roll No- 10

Class- BE(IT)

CalcServlet.java file:

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet CalcServlet</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet CalcServlet at " + request.getContextPath() + "</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.">
   * @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);
    PrintWriter out = response.getWriter();
    int x,y;
```

```
    String str = "";
    x = Integer.parseInt(request.getParameter("txtfno"));
    y = Integer.parseInt(request.getParameter("txtsno"));
    str = request.getParameter("operation");
    if(str.equals("add"))
    {out.println("<h1>Result of Addition is:" + (x+y) +"</h1>");}
    else if(str.equals("sub"))
    {out.println("<h1>Result of Subtraction is:" + (x-y) +"</h1>");}
    else if(str.equals("mult"))
    {out.println("<h1>Result of Multiplication is:" + (x*y) +"</h1>");}
    else if(str.equals("add"))
    {out.println("<h1>Result of Division is:" + (x/y) +"</h1>");}
    else{out.println("<h1>Result of Modulus is:" + (x%y) +"</h1>");}}
/**
 * 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>}
```

Index.html file-

```
<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <div>TODO write content</div>
    <form method="get" action="CalcServlet">
      <h1> Calculator </h1>
      First Number:
      <input type="text" name="txtfno"/><br/>
      Second Number:
      <input type="text" name="txtsno"/><br/>
      Select the operation:<br/>
      <input type="radio" name="operation" value="add">Addition
      <input type="radio" name="operation" value="sub">Subtraction
      <input type="radio" name="operation" value="mult">Multiplication
      <input type="radio" name="operation" value="divi">Division
      <input type="radio" name="operation" value="modu">Modulus <br/>
      <input type="submit" value="Calculate"/>
      <input type="reset" value="Reset"/>
</form>
  </body>
</html>
```

ASSIGNMENT 8 mini project

Name- Tanmay Borse

Roll No- 10

Class- BE(IT)

Server.py :

```
import socket
from _thread import *
import pickle
from game import Game
server = "192.168.20.158"
port = 5555
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
try:
    s.bind((server, port))
except socket.error as e:
    str(e)
s.listen(2)
print("Waiting for a connection, Server Started")
connected = set()
games = {}
idCount = 0
def threaded_client(conn, p, gameId):
    global idCount
    conn.send(str.encode(str(p)))
    reply = ""
    while True:
        try:
```

```
            data = conn.recv(4096).decode()
            if gameId in games:
                game = games[gameId]
                if not data:
                    break
                else:
                    if data == "reset":
                        game.resetWent()
                    elif data != "get":
                        game.play(p, data)

                    conn.sendall(pickle.dumps(game))
            else:
                break
        except:
            break
    print("Lost connection")
    try:
        del games[gameId]
        print("Closing Game", gameId)
    except:
        pass
    idCount -= 1
    conn.close()
while True:
    conn, addr = s.accept()
    print("Connected to:", addr)
```

```
    idCount += 1
    p = 0
    gameId = (idCount - 1)//2
    if idCount % 2 == 1:
        games[gameId] = Game(gameId)
        print("Creating a new game...")
    else:
        games[gameId].ready = True
        p = 1
    start_new_thread(threaded_client, (conn, p, gameId))
```

Client.py :

```
import pygame
from network import Network
import pickle
pygame.font.init()

width = 700
height = 700
win = pygame.display.set_mode((width, height))
pygame.display.set_caption("Client")



class Button:
    def __init__(self, text, x, y, color):
        self.text = text
        self.x = x
```

```
        self.y = y
        self.color = color
        self.width = 150
        self.height = 100

    def draw(self, win):
        pygame.draw.rect(win, self.color, (self.x, self.y, self.width, self.height))
        font = pygame.font.SysFont("comicsans", 40)
        text = font.render(self.text, 1, (255,255,255))
        win.blit(text, (self.x + round(self.width/2) - round(text.get_width()/2),
self.y + round(self.height/2) - round(text.get_height()/2)))

    def click(self, pos):
        x1 = pos[0]
        y1 = pos[1]
        if self.x <= x1 <= self.x + self.width and self.y <= y1 <= self.y +
self.height:
            return True
        else:
            return False



def redrawWindow(win, game, p):
    win.fill((128,128,128))

    if not(game.connected()):
        font = pygame.font.SysFont("comicsans", 80)
        text = font.render("Waiting for Player...", 1, (255,0,0), True)
```

```
    win.blit(text, (width/2 - text.get_width()/2, height/2 - text.get_height()/2))
else:
    font = pygame.font.SysFont("comicsans", 60)
    text = font.render("Your Move", 1, (0, 255,255))
    win.blit(text, (80, 200))

    text = font.render("Opponents", 1, (0, 255, 255))
    win.blit(text, (380, 200))

    move1 = game.get_player_move(0)
    move2 = game.get_player_move(1)
    if game.bothWent():
        text1 = font.render(move1, 1, (0,0,0))
        text2 = font.render(move2, 1, (0, 0, 0))
    else:
        if game.p1Went and p == 0:
            text1 = font.render(move1, 1, (0,0,0))
        elif game.p1Went:
            text1 = font.render("Locked In", 1, (0, 0, 0))
        else:
            text1 = font.render("Waiting...", 1, (0, 0, 0))

        if game.p2Went and p == 1:
            text2 = font.render(move2, 1, (0,0,0))
        elif game.p2Went:
            text2 = font.render("Locked In", 1, (0, 0, 0))
        else:
```

```
            text2 = font.render("Waiting...", 1, (0, 0, 0))


        if p == 1:
            win.blit(text2, (100, 350))
            win.blit(text1, (400, 350))
        else:
            win.blit(text1, (100, 350))
            win.blit(text2, (400, 350))
        for btn in btns:
            btn.draw(win)
    pygame.display.update()
btns = [Button("Rock", 50, 500, (0,0,0)), Button("Scissors", 250, 500,
(255,0,0)), Button("Paper", 450, 500, (0,255,0))]
def main():
    run = True
    clock = pygame.time.Clock()
    n = Network()
    player = int(n.getP())
    print("You are player", player)
    while run:
        clock.tick(60)
        try:
            game = n.send("get")
        except:
            run = False
            print("Couldn't get game")
            break
        if game.bothWent():
```

```
            redrawWindow(win, game, player)
            pygame.time.delay(500)
            try:
                game = n.send("reset")
            except:
                run = False
                print("Couldn't get game")
                break
            font = pygame.font.SysFont("comicsans", 90)
            if (game.winner() == 1 and player == 1) or (game.winner() == 0 and
player == 0):
                text = font.render("You Won!", 1, (255,0,0))
            elif game.winner() == -1:
                text = font.render("Tie Game!", 1, (255,0,0))
            else:
                text = font.render("You Lost...", 1, (255, 0, 0))
            win.blit(text, (width/2 - text.get_width()/2, height/2 -
text.get_height()/2))
            pygame.display.update()
            pygame.time.delay(2000)
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                run = False
                pygame.quit()
            if event.type == pygame.MOUSEBUTTONDOWN:
                pos = pygame.mouse.get_pos()
                for btn in btns:
                    if btn.click(pos) and game.connected():
```

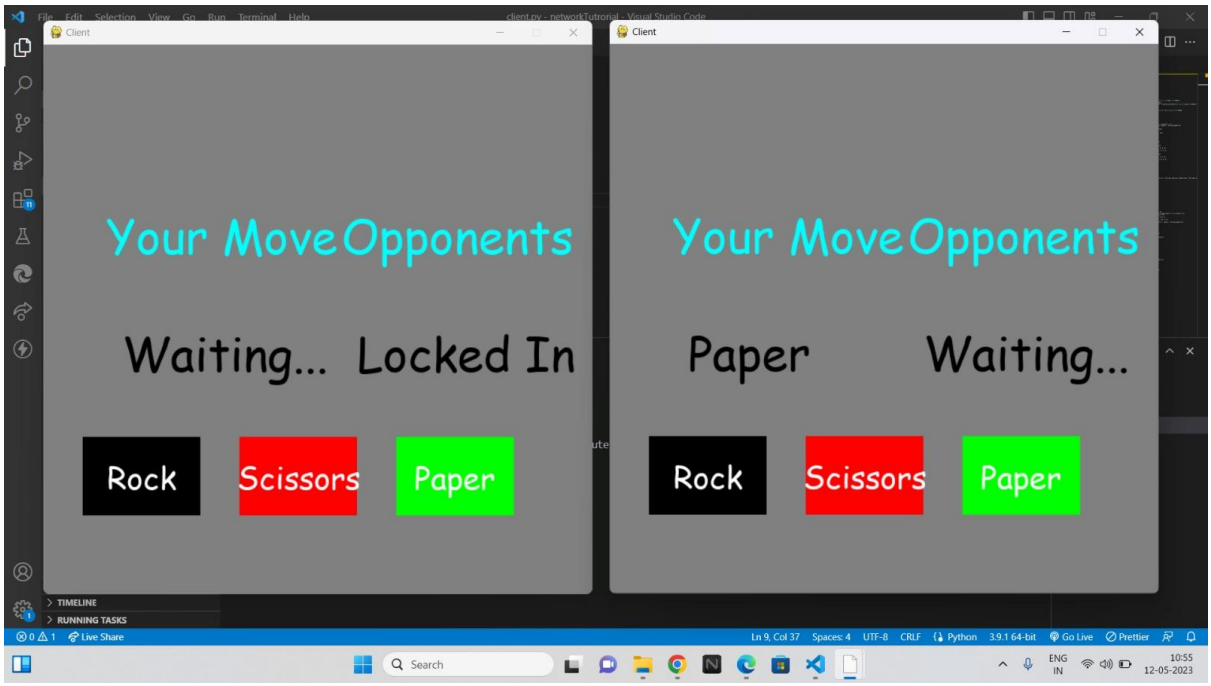
```
                if player == 0:
                    if not game.p1Went:
                        n.send(btn.text)
                else:
                    if not game.p2Went:
                        n.send(btn.text)
        redrawWindow(win, game, player)
def menu_screen():
    run = True
    clock = pygame.time.Clock()
    while run:
        clock.tick(60)
        win.fill((128, 128, 128))
        font = pygame.font.SysFont("comicsans", 60)
        text = font.render("Click to Play!", 1, (255,0,0))
        win.blit(text, (100,200))
        pygame.display.update()
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                run = False
            if event.type == pygame.MOUSEBUTTONDOWN:
                run = False
    main()
while True:
    menu_screen()
```

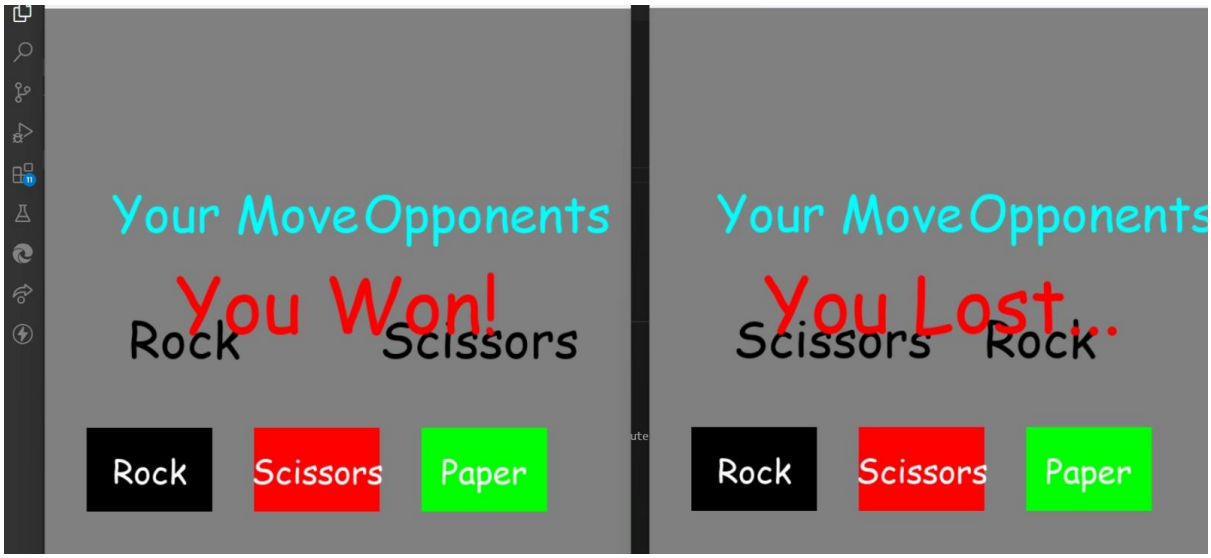
Game.py :

```
class Game:
    def __init__(self, id):
        self.p1Went = False
        self.p2Went = False
        self.ready = False
        self.id = id
        self.moves = [None, None]
        self.wins = [0,0]
        self.ties = 0

    def get_player_move(self, p):
        """
        :param p: [0,1]
        :return: Move
        """
        return self.moves[p]

    def play(self, player, move):
        self.moves[player] = move
        if player == 0:
            self.p1Went = True
        else:
            self.p2Went = True

    def connected(self):
        return self.ready

    def bothWent(self):
        return self.p1Went and self.p2Went
```

```
def winner(self):

    p1 = self.moves[0].upper()[0]
    p2 = self.moves[1].upper()[0]

    winner = -1
    if p1 == "R" and p2 == "S":
        winner = 0
    elif p1 == "S" and p2 == "R":
        winner = 1
    elif p1 == "P" and p2 == "R":
        winner = 0
    elif p1 == "R" and p2 == "P":
        winner = 1
    elif p1 == "S" and p2 == "P":
        winner = 0
    elif p1 == "P" and p2 == "S":
        winner = 1

    return winner

def resetWent(self):
    self.p1Went = False
    self.p2Went = False
```

OUTPUT :


Client
Your Move Opponents
Waiting... Locked In
Rock
Scissors
Paper
Client
Your Move Opponents
Paper Waiting...
Rock
Scissors
Paper


Your Move Opponents
You Won!
Rock Scissors
Rock
Scissors
Paper
Your Move Opponents
You Lost...
Scissors Rock
Rock
Scissors
Paper