**Assignment 6**

**Program:**

import java.util.Scanner;

class Process {

int prio, active, no;

};

public class BullyRing {

static Process[] p = new Process[10];

static Process[] q = new Process[10];

static int ch, p\_no, i = 0, j, max, co\_ordinator, new\_p, new\_p\_prio;

static int[] alive = new int[10];

static int[] high\_prio = new int[10];

static int high\_prio\_cnt, pn, high;

public static void main(String[] args) {

for (int i = 0; i < p.length; i++) {

p[i] = new Process();

}

for (int i = 0; i < q.length; i++) {

q[i] = new Process();

}

Scanner scan = new Scanner(System.in);

System.out.println("\n1.Bully\n2.Ring\n3.exit");

System.out.println("\nEnter the choice : ");

ch = scan.nextInt();

switch (ch) {

case 1:

System.out.println("\nEnter the No of processes : ");

p\_no = scan.nextInt();

for (i = 0; i < p\_no; i++) {

System.out.println("\nEnter the priority of Process P" + i + " : ");

p[i].prio = scan.nextInt();

p[i].no = i;

p[i].active = 1;

}

sort();

display();

System.out.println(

"\n\nProcess Co-ordinator is Process P" + p[0].no + " with priority " + p[0].prio + " ...");

System.out.println("\n\nChoose Process want to communicate with Co-ordinator : ");

new\_p = scan.nextInt();

for (i = 0; i < p\_no; i++) {

if (p[i].no == new\_p)

new\_p\_prio = p[i].prio;

}

System.out.println("\nIt's Priority is " + new\_p\_prio + " ");

System.out.println("\n\nProcess P" + new\_p + " sending request to Co-ordinator ...");

System.out.println(

"\n\nProcess P" + new\_p + " doesn't getting response from Co-ordinator before TimeOut...");

System.out.println("\n\ni.e. Co-Ordinator has been crashed ...");

p[0].active = 0;

System.out.println("\n\nProcess P" + new\_p + " Ininiates Election Algo ...");

System.out.println("\n\nProcess P" + new\_p + " sending election massages to : ");

high\_prio\_cnt = 0;

j = 0;

for (i = 0; i < p\_no; i++) {

if (p[i].prio >= new\_p\_prio && p[i].active == 1 && p[i].no != new\_p) {

System.out.println(" P" + p[i].no + " ");

high\_prio\_cnt++;

high\_prio[j++] = p[i].no;

}

}

if (high\_prio\_cnt == 0) {

System.out.println("\n\n No one is Alive ...");

System.out.println("\n\n\*\*\* Process P" + new\_p + " is new Co-ordinator ... \*\*\*");

} else {

System.out.println("\n\n Processes replied to Process P" + new\_p + " : ");

for (i = 0; i < high\_prio\_cnt; i++)

System.out.println(" P" + high\_prio[i] + " ");

System.out.println("\n\n Now Processes ");

for (i = 0; i < high\_prio\_cnt; i++)

System.out.println(" P" + high\_prio[i] + " ");

System.out.println(" doing Election among them");

System.out.println("\n\n New Co-ordinator is Process P" + high\_prio[0] + " ");

System.out.println("\n\n New Co-ordinator sending Co-ordinator msg to all other Processes ... ");

}

break;

case 2:

high = -999;

System.out.println("\n\nEnter The No Of Process:");

p\_no = scan.nextInt();

System.out.println("\n\nEnter The Priorities For: ");

for (i = 0; i < p\_no; i++) {

System.out.println("\n\nPriority Of P" + i + " :");

p[i].prio = scan.nextInt();

if (p[i].prio > high) {

pn = i;

high = p[i].prio;

}

System.out.println("\n\nP" + i + " Is Sending Msg To p" + (i + 1) % p\_no);

System.out.println("\n P" + i + " -> P" + (i + 1) % p\_no);

}

System.out.println("\n\nDeciding Co=ordinator....");

System.out.println("\n\nThe p" + pn + " Is Co-ordinator with priority " + high + "....");

break;

case 3:

System.exit(0);

}

}

static void display() {

System.out.println("\nP\_ID\t Priority\tActive");

for (i = 0; i < p\_no; i++)

System.out.printf("\n\n%d\t\t%d\t\t%d", p[i].no, p[i].prio, p[i].active);

}

static void sort() {

int j;

Process process = new Process();

Process temp = new Process();

for (i = 0; i < p\_no; i++) {

max = p[i].prio;

for (j = i + 1; j < p\_no; j++) {

if (p[j].prio > p[i].prio) {

temp = p[j];

p[j] = p[i];

p[i] = temp;

}

}

}

}

}

**Output:**

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 6 % javac BullyRing.java

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 6 % java BullyRing

1.Bully

2.Ring

3.exit

Enter the choice : 1

Enter the No of processes : 7

Enter the priority of Process P0 : 1

Enter the priority of Process P1 : 2

Enter the priority of Process P2 : 3

Enter the priority of Process P3 : 4

Enter the priority of Process P4 : 5

Enter the priority of Process P5 : 6

Enter the priority of Process P6 : 7

P\_ID Priority Active

6 7 1

5 6 1

4 5 1

3 4 1

2 3 1

1 2 1

0 1 1

Process Co-ordinator is Process P6 with priority 7 ...

Choose Process want to communicate with Co-ordinator : 1

It's Priority is 2

Process P1 sending request to Co-ordinator ...

Process P1 doesn't getting response from Co-ordinator before TimeOut...

i.e. Co-Ordinator has been crashed ...

Process P1 Ininiates Election Algo ...

Process P1 sending election massages to : P5 P4 P3 P2

Processes replied to Process P1 : P5 P4 P3 P2

Now Processes P5 P4 P3 P2 doing Election among them

New Co-ordinator is Process P5

New Co-ordinator sending Co-ordinator msg to all other Processes ... %

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 6 % ./a.out

1.Bully

2.Ring

3.exit

Enter the choice : 2

Enter The No Of Process:5

Enter The Priorities For:

Priority Of P0 :1

P0 Is Sending Msg To p1

P0 -> P1

Priority Of P1 :2

P1 Is Sending Msg To p2

P1 -> P2

Priority Of P2 :3

P2 Is Sending Msg To p3

P2 -> P3

Priority Of P3 :4

P3 Is Sending Msg To p4

P3 -> P4

Priority Of P4 :5

P4 Is Sending Msg To p0

P4 -> P0

Deciding Co=ordinator....

The p4 Is Co-ordinator with priority 5....%

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 6 %

**Assignment 5**

**Program:**

import java.util.InputMismatchException;

import java.util.Scanner;

class TokenRing {

public static void main(String args[]) throws Throwable {

Scanner scan = new Scanner(System.in);

System.out.println("Enter the num of nodes:");

int n = scan.nextInt();

int m = n - 1;

// Decides the number of nodes forming the ring

int token = 0;

int ch = 0, flag = 0;

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

System.out.print(" " + i);

}

System.out.println(" " + 0);

do {

System.out.println("Enter sender:");

int s = scan.nextInt();

System.out.println("Enter receiver:");

int r = scan.nextInt();

System.out.println("Enter Data:");

int a;

a = scan.nextInt();

System.out.print("Token passing:");

for (int i = token, j = token; (i % n) != s; i++, j = (j + 1) % n) {

System.out.print(" " + j + "->");

}

System.out.println(" " + s);

System.out.println("Sender " + s + " sending data: " + a);

for (int i = s + 1; i != r; i = (i + 1) % n) {

System.out.println("data " + a + " forwarded by " + i);

}

System.out.println("Receiver " + r + " received data: " + a + "\n");

token = s;

do {

try {

if (flag == 1)

System.out.print("Invalid Input!!...");

System.out.print("Do you want to send again?? enter 1 for Yes and 0 for No : ");

ch = scan.nextInt();

if (ch != 1 && ch != 0)

flag = 1;

else

flag = 0;

} catch (InputMismatchException e) {

System.out.println("Invalid Input");

}

} while (ch != 1 && ch != 0);

} while (ch == 1);

}

}

**Output:**

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 5 % javac TokenRing.java

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 5 % java TokenRing

Enter the num of nodes:

5

0 1 2 3 4 0

Enter sender:

3

Enter receiver:

4

Enter Data:

3

Token passing: 0-> 1-> 2-> 3

Sender 3 sending data: 3

Receiver 4 received data: 3

Do you want to send again?? enter 1 for Yes and 0 for No : 1

Enter sender:

3

Enter receiver:

2

Enter Data:

5

Token passing: 3

Sender 3 sending data: 5

data 5 forwarded by 4

data 5 forwarded by 0

data 5 forwarded by 1

Receiver 2 received data: 5

Do you want to send again?? enter 1 for Yes and 0 for No : 0

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 5 %

**Assignment 4**

**Program:**

file: master.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)

**----------------------------------------------------------------------------------------------------------------------**

file: 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()

# Driver function

if \_\_name\_\_ == '\_\_main\_\_':

# initialize the Slave / Client

initiateSlaveClient(port = 8080)

**Output:**

master.py

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 4 % python master.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.

127.0.0.1:61915 got connected successfully

Client Data updated with: 127.0.0.1:61915

New synchronization cycle started.

Number of clients to be synchronized: 1

Client Data updated with: 127.0.0.1:61915

New synchronization cycle started.

Number of clients to be synchronized: 1

Client Data updated with: 127.0.0.1:61915

**----------------------------------------------------------------------------------------------------------------------**

client.py

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 4 % python client.py

Starting to receive time from server

Starting to receiving synchronized time from server

Recent time sent successfully

Synchronized time at the client is: 2023-03-26 19:13:32.452529

Recent time sent successfully

Synchronized time at the client is: 2023-03-26 19:13:37.458662

Recent time sent successfully

**Assignment 1**

**Program:**

Search.java

import java.rmi.Remote;

import java.rmi.RemoteException;

interface Search extends Remote

{

// Declaring the method prototype

public String query(String search) throws RemoteException;

}

**----------------------------------------------------------------------------------------------------------------------**

SearchQuery.java

// Java program to implement the Search interface

import java.rmi.\*;

import java.rmi.server.\*;

public class SearchQuery extends UnicastRemoteObject

implements Search

{

// Default constructor to throw RemoteException

// from its parent constructor

SearchQuery() throws RemoteException

{

super();

}

// Implementation of the query interface

public String query(String search)

throws RemoteException

{

String result;

if (search.equals("Reflection in Java"))

result = "Found";

else

result = "Not Found";

return result;

}

}

**----------------------------------------------------------------------------------------------------------------------**

SearchServer.java

// Java program for server application

import java.rmi.\*;

import java.rmi.registry.\*;

public class SearchServer

{

public static void main(String args[])

{

try

{

// Create an object of the interface

// implementation class

Search obj = new SearchQuery();

// rmiregistry within the server JVM with

// port number 1900

LocateRegistry.createRegistry(3000);

// Binds the remote object by the name

Naming.rebind("rmi://localhost:3000"+

"/suyash",obj);

}

catch(Exception ae)

{

System.out.println(ae);

}

}

}

**----------------------------------------------------------------------------------------------------------------------**

ClientRequest.java

import java.rmi.Naming;

// Java program for client application

public class ClientRequest

{

public static void main(String args[])

{

String answer,value="Reflection in Java";

try

{

// lookup method to find reference of remote object

Search access =

(Search)Naming.lookup("rmi://localhost:3000"+

"/suyash");

answer = access.query(value);

System.out.println("Article on " + value +

" " + answer);

}

catch(Exception ae)

{

System.out.println(ae);

}

}

}

**Output:**

**Console 1:**

(base) suyashnehete@Suyashs-MacBook-Pro src % Javac SearchQuery.java

(base) suyashnehete@Suyashs-MacBook-Pro src % rmic SearchQuery

Warning: generation and use of skeletons and static stubs for JRMP

is deprecated. Skeletons are unnecessary, and static stubs have

been superseded by dynamically generated stubs. Users are

encouraged to migrate away from using rmic to generate skeletons and static

stubs. See the documentation for java.rmi.server.UnicastRemoteObject.

(base) suyashnehete@Suyashs-MacBook-Pro src % rmiregistry

**Console 2:**

(base) suyashnehete@Suyashs-MacBook-Pro src % javac SearchServer.java

(base) suyashnehete@Suyashs-MacBook-Pro src % java SearchServer

**Console 3:**

(base) suyashnehete@Suyashs-MacBook-Pro src % java ClientRequest

Article on Reflection in Java Found

(base) suyashnehete@Suyashs-MacBook-Pro src %

**Assignment 2**

**Program:**

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();

}

}

}

**----------------------------------------------------------------------------------------------------------------------**

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));

}

}

**----------------------------------------------------------------------------------------------------------------------**

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();

}

}

}

**Output:**

**Server Side**

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 2 % idlj -fall ReverseModule.idl

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 2 % javac \*.java ReverseModule/\*.java

Note: ReverseModule/ReversePOA.java uses unchecked or unsafe operations.

Note: Recompile with -Xlint:unchecked for details.

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 2 % orbd -ORBInitialPort 1050&[1] 5163

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 2 % java ReverseServer - ORBInitialPort 1050& -ORBInitialHost localhost& [1] 4933 [2] 4934

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 2 % -ORBInitialHost: command not found

Reverse Object Created

Step1

Step2

Step3

Step4

Reverse Server reading and waiting....

**----------------------------------------------------------------------------------------------------------------------**

**Client Side**

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 2 % java ReverseClient -ORBInitialPort 1050 -ORBInitialHost localhost

Enter String=

Hello world.

Server Send .dlrow olleH

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 2 %

**Assignment 3**

**Program:**

import mpi.MPI;

public class ScatterGather {

public static void main(String args[]){

//Initialize MPI execution environment

MPI.Init(args);

//Get the id of the process

int rank = MPI.COMM\_WORLD.Rank();

//total number of processes is stored in size

int size = MPI.COMM\_WORLD.Size();

int root=0;

//array which will be filled with data by root process

int sendbuf[]=null;

sendbuf= new int[size];

//creates data to be scattered

if(rank==root){

sendbuf[0] = 10;

sendbuf[1] = 20;

sendbuf[2] = 30;

sendbuf[3] = 40;

//print current process number

System.out.print("Processor "+rank+" has data: ");

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

System.out.print(sendbuf[i]+" ");

}

System.out.println();

}

//collect data in recvbuf

int recvbuf[] = new int[1];

//following are the args of Scatter method

//send, offset, chunk\_count, chunk\_data\_type, recv, offset,

//chunk\_count, chunk\_data\_type, root\_process\_id

MPI.COMM\_WORLD.Scatter(sendbuf, 0, 1, MPI.INT, recvbuf, 0,

1, MPI.INT, root);

System.out.println("Processor "+rank+" has data:"+recvbuf[0]);

System.out.println("Processor "+rank+" is doubling the data");

recvbuf[0]=recvbuf[0]\*2;

//following are the args of Gather method

//Object sendbuf, int sendoffset, int sendcount, Datatype

//sendtype, Object recvbuf, int recvoffset, int recvcount,

//Datatype recvtype,

//int root)

MPI.COMM\_WORLD.Gather(recvbuf, 0, 1, MPI.INT, sendbuf, 0,

1, MPI.INT, root);

//display the gathered result

if(rank==root){

System.out.println("Process 0 has data: ");

for(int i=0;i<4;i++){

System.out.print(sendbuf[i]+ " ");

}

}

//Terminate MPI execution

//environment MPI.Finalize();

}

}

**Output:**

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 3 % javac -cp mpj/lib/mpj.jar ScatterGather.java

(base) suyashnehete@Suyashs-MacBook-Pro Assignment 3 % mpj/bin/mpjrun.sh -np 4 ScatterGather

MPJ Express (0.44) is started in the multicore configuration

Processor 0 has data: 10 20 30 40

Processor 0 has data: 10

Processor 2 has data: 30

Processor 1 has data: 20

Processor 3 has data: 40

Processor 2 is doubling the data

Processor 1 is doubling the data

Processor 3 is doubling the data

Processor 0 is doubling the data

Process 0 has data: 20 40 60 80