Practical 1

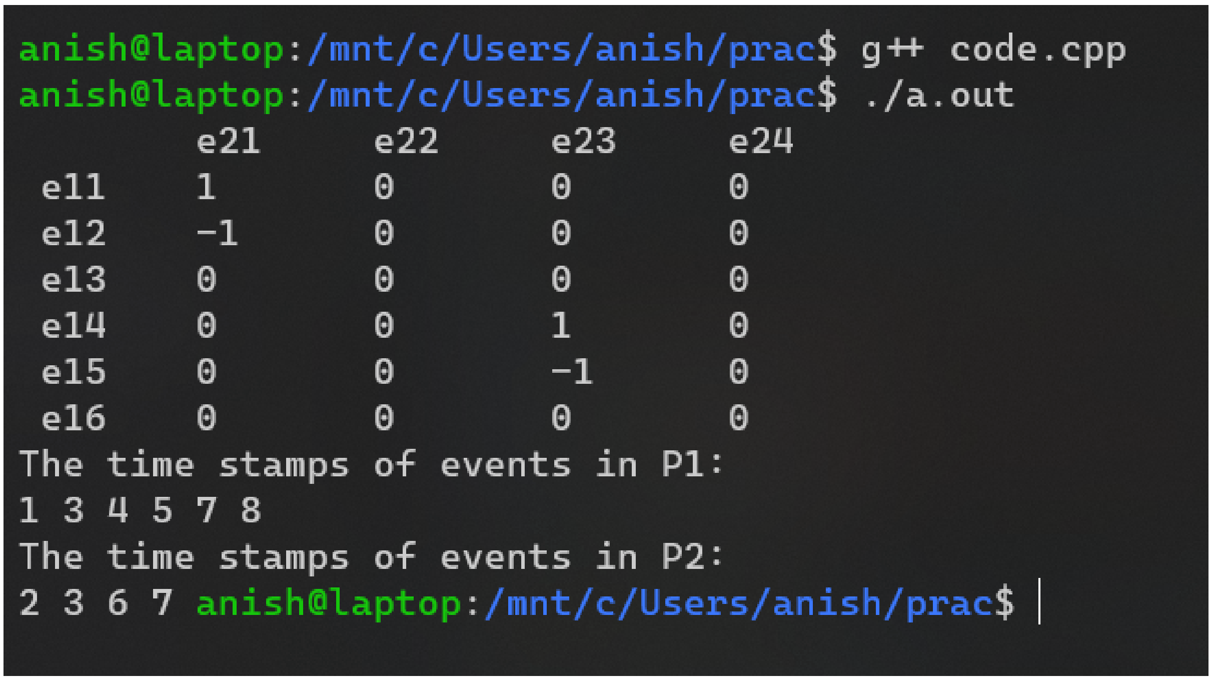
Aim - Program to Implement Lamport's Logical Clock

Code

```
#include <bits/stdc++.h>
using namespace std;
void stamps(int, int, int *, int *);
void lamportLogicalClock(int, int, int (*)[4]);
int findmax(int, int);
int main()
{
    int e1 = 6, e2 = 4, m[6][4];
    m[0][0] = 1;
    m[0][1] = 0;
    m[0][2] =
        0;
    m[0][3] = 0;
    m[1][0] = -1;
    m[1][1] = 0;
    m[1][2] = 0;
    m[1][3] = 0;
    m[2][0] = 0;
    m[2][1] = 0;
    m[2][2] = 0;
    m[2][3] = 0;
    m[3][0] =
        0;
    m[3][1] = 0;
    m[3][2] = 1;
    m[3][3] = 0;
    m[4][0] = 0;
    m[4][1] =
        0;
    m[4][2] = -1;
    m[4][3] = 0;
    m[5][0] = m[5][1] = m[5][2] = m[5][3] = 0;
    lamportLogicalClock(e1, e2,
                        m);
    return 0;
}
void stamps(int e1, int e2, int p1[6], int p2[4])
```

```cpp
{
    int i;
    cout << "\nThe time stamps of events in P1:\n";
    for (i = 0; i < e1; i++)
        cout << p1[i] << " ";
    cout << "\nThe time stamps of events in P2:\n";
    for (i = 0; i < e2; i++)
        cout << p2[i] << " ";
}
void lamportLogicalClock(int e1, int e2, int m[6][4])
{
    int i, j, k, p1[e1], p2[e2];
    for (i = 0; i < e1; i++)
        p1[i] = i + 1;
    for (i = 0; i < e2; i++)
        p2[i] = i + 1;
    for (i = 0; i < e2; i++)
        cout << "\te2" << i + 1;
    for (i = 0; i < e1; i++)
    {
        cout << "\n e1" << i + 1 << "\t";
        for (j = 0; j < e2; j++)
            cout << m[i][j] << "\t";
    }
    for (i = 0; i < e1; i++)
    {
        for (j = 0; j < e2; j++)
        {
            if (m[i][j] == 1)
            {
                p2[j] = findmax(p2[j], p1[i] + 1);
                for (k = j + 1; k < e2; k++)
                    p2[k] = p2[k - 1] + 1;
            }
            if (m[i][j] == -1)
            {
                p1[i] = findmax(p1[i], p2[j] + 1);
                for (k = i + 1; k < e1; k++)
                    p1[k] = p1[k - 1] + 1;
            }
        }
    }
    stamps(e1, e2, p1, p2);
}
int findmax(int a, int b)
{
    if (a > b)
        return a;
    return b;
}
```

Output



```
anish@laptop:/mnt/c/Users/anish/prac$ g++ code.cpp
anish@laptop:/mnt/c/Users/anish/prac$ ./a.out
        e21     e22     e23     e24
 e11    1       0       0       0
 e12    -1      0       0       0
 e13    0       0       0       0
 e14    0       0       1       0
 e15    0       0       -1      0
 e16    0       0       0       0
The time stamps of events in P1:
1 3 4 5 7 8
The time stamps of events in P2:
2 3 6 7 anish@laptop:/mnt/c/Users/anish/prac$
```

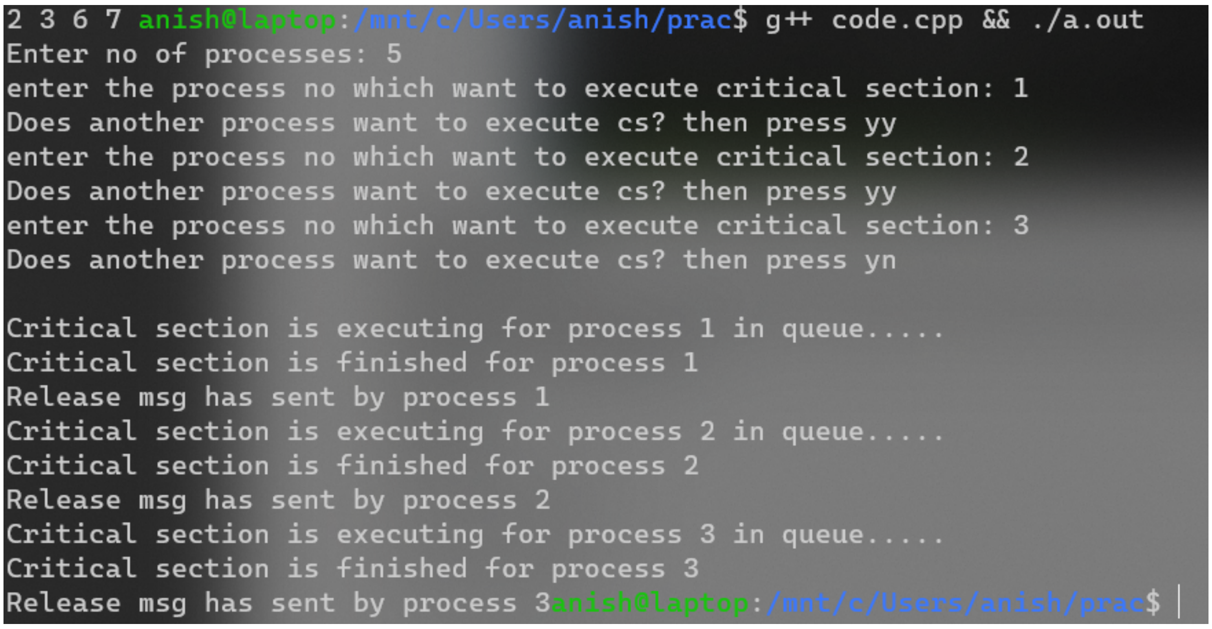
Practical 2

Aim - Program to implement non token based algorithm for Mutual Exclusion

Code

```cpp
#include <iostream>
using namespace std;
int main()
{
    int i = 0, d, p, a, c = 0, aa[10], j, n;
    char ch = 'y';
    cout << "Enter no of processes: ";
    cin >> n;
    do
    {
        cout << "enter the process no which want to execute critical section: ";
        cin >> a;
        aa[i] = a;
        i++;
        c = c + 1;
        d = i;
        cout << "Does another process want to execute cs? then press y";
        cin >> ch;
    } while (ch == 'y');
    for (j = 1; j <= c; j++)
    {
        cout << "\nCritical section is executing for process " << j << " in queue..... ";
        cout << "\nCritical section is finished for process " << j;
        cout << "\nRelease msg has sent by process " << j;
    }
    return 0;
}
```

Output



```
2 3 6 7 anish@laptop:/mnt/c/Users/anish/prac$ g++ code.cpp && ./a.out
Enter no of processes: 5
enter the process no which want to execute critical section: 1
Does another process want to execute cs? then press yy
enter the process no which want to execute critical section: 2
Does another process want to execute cs? then press yy
enter the process no which want to execute critical section: 3
Does another process want to execute cs? then press yn

Critical section is executing for process 1 in queue.....
Critical section is finished for process 1
Release msg has sent by process 1
Critical section is executing for process 2 in queue.....
Critical section is finished for process 2
Release msg has sent by process 2
Critical section is executing for process 3 in queue.....
Critical section is finished for process 3
Release msg has sent by process 3anish@laptop:/mnt/c/Users/anish/prac$
```

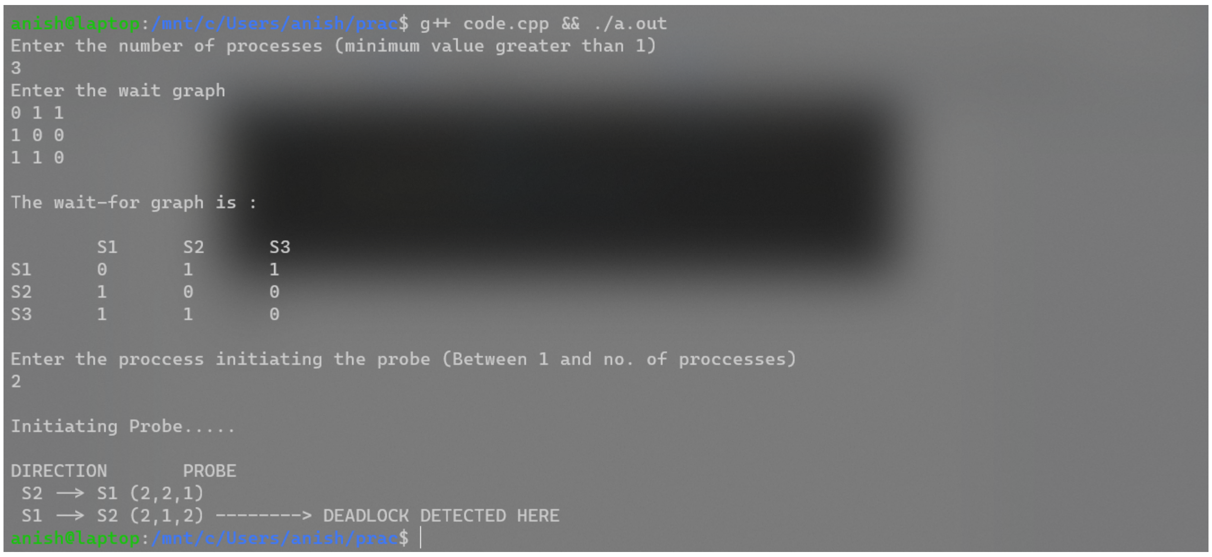
Practical 3

Aim - Program to implement edge chasing distributed deadlock detection algorithm.

Code

```
#include <iostream>
#include <vector>
#include <string>
#include <queue>
#include <stdio.h>
using namespace std;
void detectDeadlock(vector<std::vector<int>> &graph, int init, int
dest);
void displayGraph(vector<vector<int>> mat);
int processes;
bool deadlocked = false;
int main()
{
    int pid_probe;
    cout << "Enter the number of processes (minimum value greater than
1)" << endl;
    cin >> processes;
    if (processes > 1)
    {
        cout << "Enter the wait graph" << endl;
        vector<vector<int>> wait_graph(processes);
        for (int from = 0; from < processes; from++)
        {
            for (int to = 0; to < processes; to++)
            {
                int temp;
                cin >> temp;
                wait_graph.at(from).push_back(temp);
            }
        }
        cout << endl;
        cout << "The wait-for graph is : " << endl
             << endl;
        displayGraph(wait_graph);
        cout << endl;
        cout << "Enter the proccess initiating the probe (Between 1 and
no. of proccesses)" << endl;
        cin >> pid_probe;
        cout << endl;
        pid_probe = pid_probe - 1;
        cout << "Initiating Probe....." << endl
             << endl;
        cout << "DIRECTION"
             << "\t"
             << "PROBE" << endl;
        for (int col = 0; col < processes; col++)
        {
            if (wait_graph.at(pid_probe).at(col) == 1)
            {
                cout << " S" << (pid_probe + 1) << " --> S" << (col +
1) << " (" << (pid_probe + 1) << "," << (pid_probe + 1) << "," << (col
+ 1) << ")" << endl;
                detectDeadlock(wait_graph, pid_probe, col);
            }
        }
    }
    else
    {
        cout << "Deadlock detection not possbile. No proccess running
in the system" << endl;
    }
    return 0;
}
void displayGraph(vector<vector<int>> wait_graph)
{
    int n = wait_graph.at(0).size();
    int m = wait_graph.size();
    cout << "\t";
    for (int j = 0; j < m; j++)
    {
        cout << "S" << (j + 1) << "\t";
    }
    cout << endl;
    for (int i = 0; i < m; i++)
    {
        cout << "S" << (i + 1) << "\t";
        for (int j = 0; j < n; j++)
        {
            cout << wait_graph.at(i).at(j) << "\t";
        }
        cout << "\n";
    }
}
void detectDeadlock(vector<std::vector<int>> &graph, int init, int
dest)
{
    int end = processes;
    for (int col = 0; col < end; col++)
    {
        if (graph[dest][col] == 1)
        {
            if (init == col)
            {
                cout << " S" << (dest + 1) << " --> S" << (col + 1) <<
" (" << (init + 1) << "," << (dest + 1) << "," << (col + 1) << ")"
                     << " ---------> DEADLOCK DETECTED HERE " << endl;
                    deadlocked = true;
                break;
            }
            cout << " S" << (dest + 1) << " --> S" << (col + 1) << " ("
<< (init + 1)
                 << "," << (dest + 1) << "," << (col + 1) << ")" <<
endl;
            detectDeadlock(graph, init, col);
        }
    }
}
```

Output



```
anish@laptop:/mnt/c/Users/anish/prac$ g++ code.cpp && ./a.out
Enter the number of processes (minimum value greater than 1)
3
Enter the wait graph
0 1 1
1 0 0
1 1 0

The wait-for graph is :

        S1      S2      S3
S1      0       1       1
S2      1       0       0
S3      1       1       0

Enter the proccess initiating the probe (Between 1 and no. of proccesses)
2

Initiating Probe.....

DIRECTION       PROBE
 S2 --> S1 (2,2,1)
 S1 --> S2 (2,1,2) ---------> DEADLOCK DETECTED HERE
anish@laptop:/mnt/c/Users/anish/prac$
```

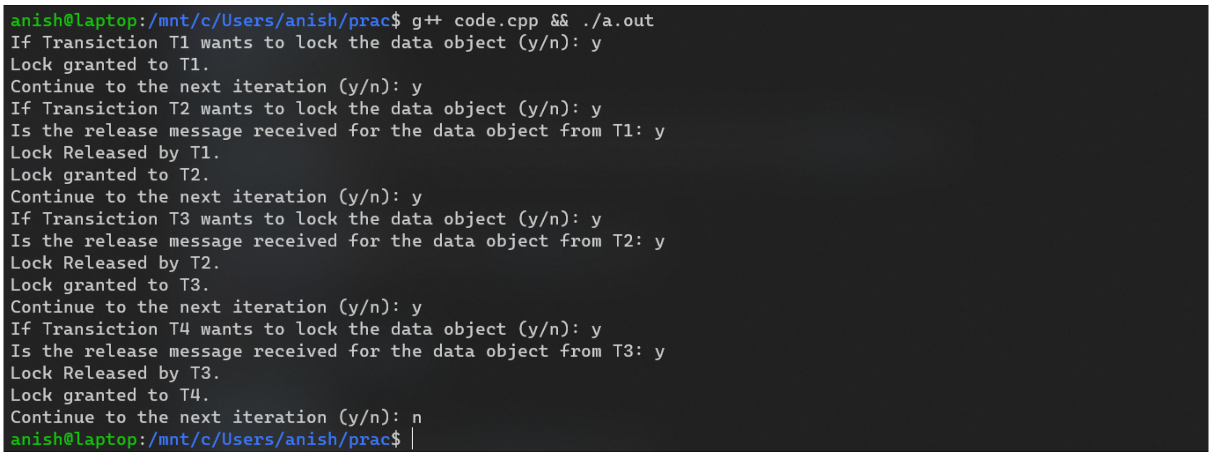
Practical 4

Aim - Program to implement locking algorithm.

Code

```cpp
#include <iostream>
using namespace std;
bool locked = false;
int holder;
bool check_lock_release()
{
    char choice;
    cout << "Is the release message received for the data object from T" << holder << ": ";
    cin >> choice;
    if (choice == 'y')
    {
        locked = false;
        cout << "Lock Released by T" << holder << ".\n";
        return true;
    }
    return false;
}
void get_lock(int transaction)
{
    char choice;
    locked = true;
    choice = 'n';
    cout << "Lock granted to T" << transaction << ".\n";
    holder = transaction;
}
int main()
{
    char choice, next = 'n';
    int iter = 1;
    do
    {
        cout << "If Transiction T" << iter << " wants to lock the data object (y/n): ";
        cin >> choice;
        if (choice == 'y' && !locked)
        {
            get_lock(iter);
        }
        else if (locked)
        {
            if (check_lock_release())
            {
                get_lock(iter);
            }
            else
                cout << "Data object is locked \n";
        }
        else
        {
            check_lock_release();
        }
        cout << "Continue to the next iteration (y/n): ";
        cin >> next;
        iter++;
    } while (next == 'y');
}
```

Output



```
anish@laptop:/mnt/c/Users/anish/prac$ g++ code.cpp && ./a.out
If Transiction T1 wants to lock the data object (y/n): y
Lock granted to T1.
Continue to the next iteration (y/n): y
If Transiction T2 wants to lock the data object (y/n): y
Is the release message received for the data object from T1: y
Lock Released by T1.
Lock granted to T2.
Continue to the next iteration (y/n): y
If Transiction T3 wants to lock the data object (y/n): y
Is the release message received for the data object from T2: y
Lock Released by T2.
Lock granted to T3.
Continue to the next iteration (y/n): y
If Transiction T4 wants to lock the data object (y/n): y
Is the release message received for the data object from T3: y
Lock Released by T3.
Lock granted to T4.
Continue to the next iteration (y/n): n
anish@laptop:/mnt/c/Users/anish/prac$
```

Practical 5

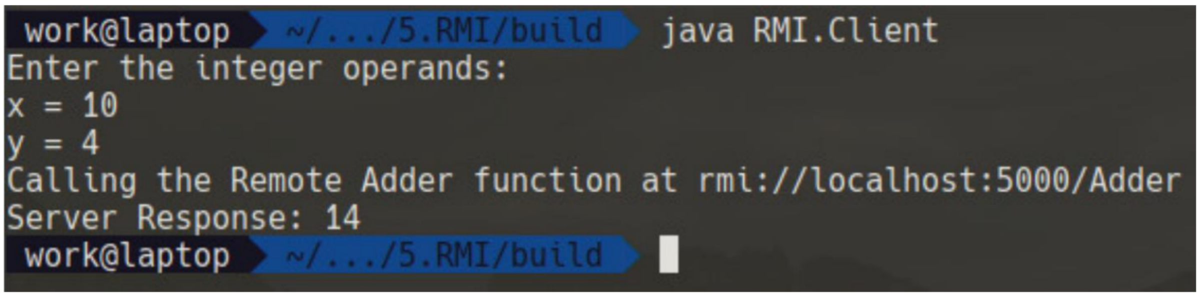
Aim - Program to implement Remote Method Invocation.

Code

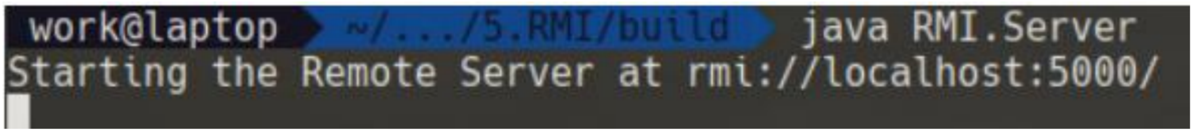
```
//Adder.java
package RMI;
import java.rmi.Remote;
import java.rmi.RemoteException;
public interface Adder extends Remote {
 int add(int x, int y) throws RemoteException;
}
//Adder_Remote.java
package RMI;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
public class AdderRemote extends UnicastRemoteObject implements Adder {
 AdderRemote() throws RemoteException {
 super();
 }
 public int add(int x, int y) {
 return x + y;
 }
}
//Client.java
package RMI;
import java.rmi.Naming;
import java.util.Scanner;
public class Client {
 private static final String URL = "rmi://localhost:5000/Adder";
 public static void main(String[] args) {
 Scanner input = new Scanner(System.in);
 System.out.println("Enter the integer operands: ");
 System.out.print("x = "); int x = input.nextInt();
 System.out.print("y = "); int y = input.nextInt();
 try {
 System.out.println("Calling the Remote Adder function at " + URL);
 Adder stub = (Adder) Naming.lookup(URL);
 System.out.println("Server Response: " + stub.add(x, y));
 } catch (Exception e) { System.out.println("Exception: " + e); } }}
//Server.java
package RMI;
import java.rmi.Naming;
```

```java
public class Server {
 public static void main(String[] args) {
 System.out.println("Starting the Remote Server at
rmi://localhost:5000/");
 try {
 Adder stub = new AdderRemote();
 Naming.bind("rmi://localhost:5000/Adder", stub);
 } catch (Exception e) {
 System.out.println("Exception: " + e); } }}
```

Output



```
 work@laptop  ~/.../5.RMI/build  java RMI.Client
Enter the integer operands:
x = 10
y = 4
Calling the Remote Adder function at rmi://localhost:5000/Adder
Server Response: 14
 work@laptop  ~/.../5.RMI/build 
```



```
 work@laptop  ~/.../5.RMI/build  java RMI.Server
Starting the Remote Server at rmi://localhost:5000/
```

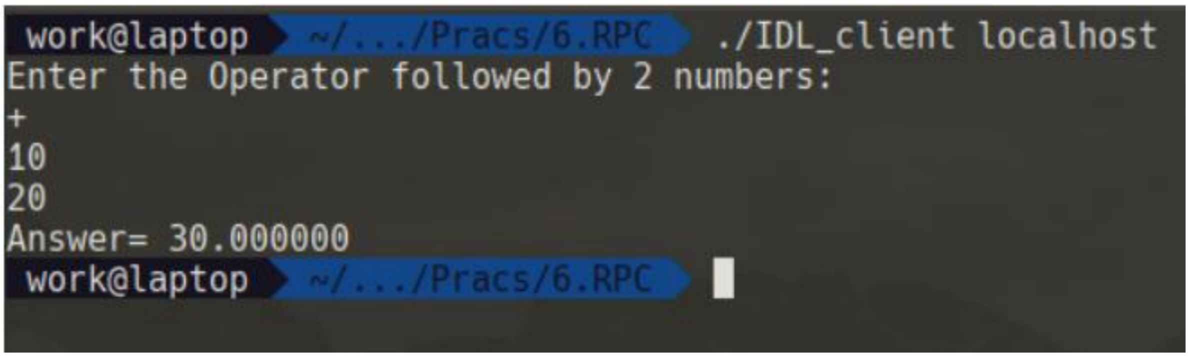
Practical 6

Aim - Program to implement Remote Procedure Call.

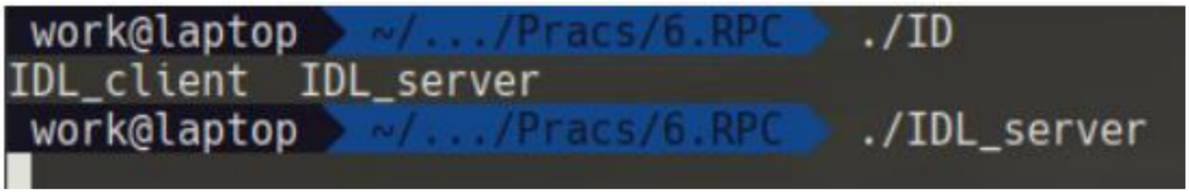
Code

```
//Client.c
#include "IDL.h"
#include <stdio.h>
float compute_6(char *host,float a,float b,char op) {
CLIENT *clnt;float *result_1;
values add_6_arg; float *result_2;
values sub_6_arg;float *result_3;values mul_6_arg;float *result_4;
values div_6_arg;
if(op=='+')
{add_6_arg.num[0]=a;add_6_arg.num[1]=b;
add_6_arg.operation=op;clnt = clnt_create (host, COMPUTE, COMPUTE_VERS,
"udp"); if (clnt == NULL) {clnt_pcreateerror (host);exit (1);
}result_1 = add_6(&add_6_arg, clnt);
if (result_1 == (float *) NULL) { clnt_perror (clnt, "call failed");
}clnt_destroy (clnt);
return (*result_1);
}
else if(op=='-'){
sub_6_arg.num[0]=a;sub_6_arg.num[1]=b;sub_6_arg.operation=op;
clnt = clnt_create (host, COMPUTE, COMPUTE_VERS, "udp");
if (clnt == NULL) { clnt_pcreateerror (host);exit (1);}
result_2 = sub_6(&sub_6_arg, clnt); if (result_2 == (float *) NULL) {
clnt_perror (clnt, "call failed");
} clnt_destroy (clnt);
return (*result_2); }else if(op=='*'){
mul_6_arg.num[0]=a; mul_6_arg.num[1]=b; mul_6_arg.operation=op;
clnt = clnt_create (host, COMPUTE, COMPUTE_VERS, "udp");
if (clnt == NULL) {clnt_pcreateerror (host);exit (1); }
result_3 = mul_6(&mul_6_arg, clnt); if (result_3 == (float *) NULL) {
clnt_perror (clnt, "call failed"); }
clnt_destroy (clnt);return (*result_3); }
else if(op=='/'){ if(b==0){printf("You are trying to divide by zero.
Please insert a valid
number.\n"); exit(0); }
else{ div_6_arg.num[0]=a;div_6_arg.num[1]=b;div_6_arg.operation=op;
clnt = clnt_create (host, COMPUTE, COMPUTE_VERS, "udp");if (clnt ==
NULL) {
clnt_pcreateerror (host);
exit (1); }result_4 = div_6(&div_6_arg, clnt);
if (result_4 == (float *) NULL) {
clnt_perror (clnt, "call failed"); }
clnt_destroy (clnt); return (*result_4);
} }}int main (int argc, char *argv[]) {char *host;
float number1,number2;char oper; printf("Enter the Operator followed by
2
numbers:\n");
scanf("%c",&oper); scanf("%f",&number1); scanf("%f",&number2); host =
argv[1];
printf("Answer= %f\n",compute_6 (host,number1,number2,oper));
exit(0);
}//Server.c
#include "IDL.h"
float *
add_6_svc(values *argp, struct svc_req *rqstp)
{ static float result;
result = argp->num[0] + argp->num[1];return &result;}
float *
sub_6_svc(values *argp, struct svc_req *rqstp)
{ static float result; result = argp->num[0] - argp->num[1];
return &result;}
float *
mul_6_svc(values *argp, struct svc_req *rqstp)
{
static float result;result = argp->num[0] * argp->num[1];
return &result;} float *
div_6_svc(values *argp, struct svc_req *rqstp)
{ static float result;result = argp->num[0] / argp->num[1];
return &result;}
//IDL.x
struct values {float num[2];char operation;};program COMPUTE {version
COMPUTE_VERS {
float ADD(values) = 1;float SUB(values) = 2;float MUL(values) = 3;float
DIV(values) = 4;
} = 6;} = 10;
```

Output



```
 work@laptop  ~/.../Pracs/6.RPC  ./IDL_client localhost
Enter the Operator followed by 2 numbers:
+
10
20
Answer= 30.000000
 work@laptop  ~/.../Pracs/6.RPC 
```



```
 work@laptop  ~/.../Pracs/6.RPC  ./ID
IDL_client  IDL_server
 work@laptop  ~/.../Pracs/6.RPC  ./IDL_server
```

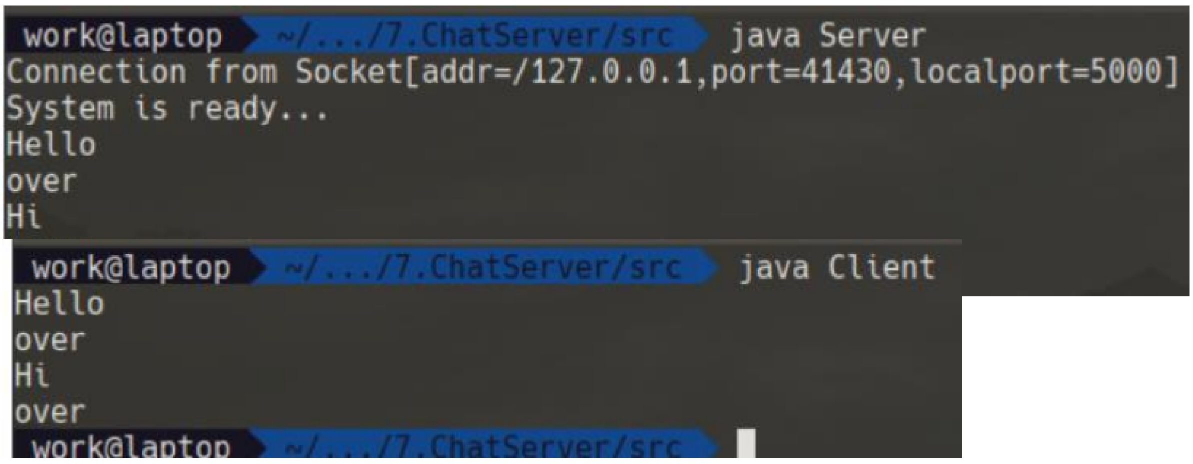
Practical 7

Aim - Program to implement Chat Server.

Code

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.InetAddress;
import java.net.Socket;
@SuppressWarnings("InfiniteLoopStatement")
public class Client {
 public static void main(String[] args) {
 try {
 Socket socket = new Socket(InetAddress.getLocalHost(), 5000);
 PrintWriter printWriter = new PrintWriter(socket.getOutputStream(),
true);
 BufferedReader bufferedReaderSocket = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
 BufferedReader bufferedReaderIO = new BufferedReader((new
InputStreamReader(System.in)));
 String input, output;
 while (true) {
 do {
 output = bufferedReaderSocket.readLine();
 System.out.println(output);
 } while (!output.equals("over"));
 do {
 input = bufferedReaderIO.readLine();
 printWriter.println(input);
 } while (!input.equals("over"));
 return;
 } } catch (Exception e) {
 System.out.println("Exception: " + e);
 } }}
//Server.java
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
@SuppressWarnings("InfiniteLoopStatement")
public class Server {
 public static void main(String[] args) {
 try {
 ServerSocket socketServer = new ServerSocket(5000);
 Socket socket = socketServer.accept();
 System.out.println("Connection from " + socket);
 PrintWriter printWriter = new PrintWriter(socket.getOutputStream(),
true);
 BufferedReader bufferedReaderSocket = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
 BufferedReader bufferedReaderIO = new BufferedReader(new
InputStreamReader(System.in));
 System.out.println("System is ready...");
 String input, output;
 while (true) {
 do {
 output = bufferedReaderIO.readLine();
 printWriter.println(output);
 } while (!output.equals("over"));
 do {
 input = bufferedReaderSocket.readLine();
 System.out.println(input);
 } while (!input.equals("over"));
 } } catch (Exception e) {
 System.out.println("Exception: " + e);
 } }}
```

Output



```
work@laptop  ~/.../7.ChatServer/src  java Server
Connection from Socket[addr=/127.0.0.1,port=41430,localport=5000]
System is ready...
Hello
over
Hi
```

```
work@laptop  ~/.../7.ChatServer/src  java Client
Hello
over
Hi
over
work@laptop  ~/.../7.ChatServer/src
```

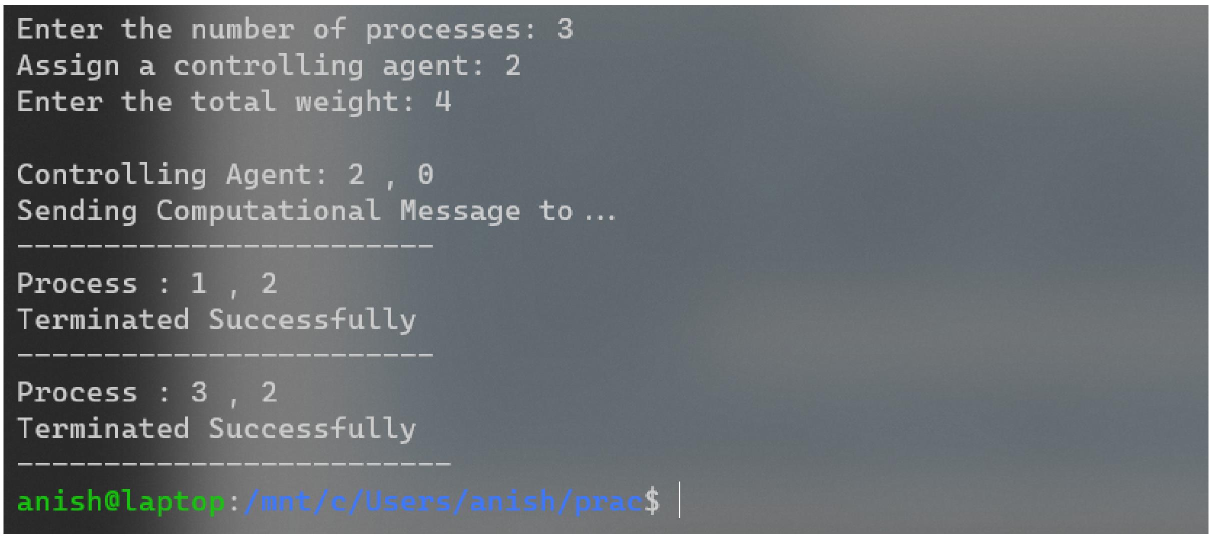
Practical 8

Aim - Program to implement termination detection

Code

```
#include <iostream>
#include <random>
using namespace std;
int main()
{
    system("clear");
    size_t processes, controlling, total_weight;
    cout << "Enter the number of processes: ";
    cin >> processes;
    cout << "Assign a controlling agent: ";
    cin >> controlling;
    cout << "Enter the total weight: ";
    cin >> total_weight;
    random_device rd;
    default_random_engine e(rd());
    uniform_int_distribution<> random(0, total_weight);
    vector<size_t> weight(processes);
    size_t current_max = total_weight;
    for (size_t i = 0; i < processes; i++)
    {
        weight[i] = random(e) % current_max;
        current_max -= weight[i];
    }
    weight[processes - 1] += current_max;
    cout << "\nControlling Agent: " << controlling << " , " <<
weight[controlling - 1] << "\n";
    cout << "Sending Computational Message to...\n";
    for (size_t j = 0; j < processes; j++)
    {
        if (j != (controlling - 1))
        {
            cout << "-------------------------\n";
            cout << "Process : " << j + 1 << " , " << weight[j] <<
"\n";
            cout << "Terminated Successfully \n";
        }
    }
    cout << "-------------------------\n";
    return 0;
}
```

Output



```
Enter the number of processes: 3
Assign a controlling agent: 2
Enter the total weight: 4

Controlling Agent: 2 , 0
Sending Computational Message to ...
------------------------
Process : 1 , 2
Terminated Successfully
------------------------
Process : 3 , 2
Terminated Successfully
-------------------------
anish@laptop:/mnt/c/Users/anish/prac$
```

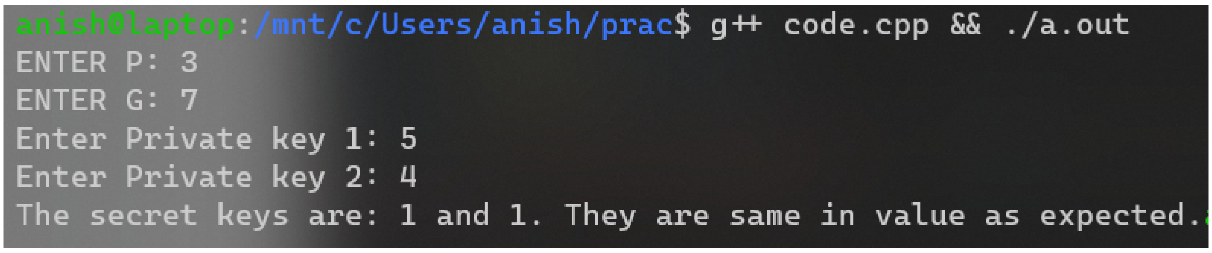
Practical 9

Aim - Diff Hellman key exchange algorithm

Code

```cpp
#include <iostream>
#include <cmath>
using namespace std;
long int exponent(long int, long int, long int);
int main()
{
    long int P, G, x, a, y, b, ka, kb;
    cout << "ENTER P: ";
    cin >> P;
    cout << "ENTER G: ";
    cin >> G;
    cout << "Enter Private key 1: ";
    cin >> a;
    cout << "Enter Private key 2: ";
    cin >> b;
    x = exponent(G, a, P);
    y = exponent(G, b, P);
    ka = exponent(y, a, P);
    kb = exponent(x, b, P);
    cout << "The secret keys are: " << ka << " and " << kb << ". They
are same in value as expected.";
    return 0;
}
long int exponent(long int a, long int b, long int P)
{
    if (b == 1)
        return a;
    else
        return (((long int)pow(a, b)) % P);
}
```

Output



```
anish@laptop:/mnt/c/Users/anish/prac$ g++ code.cpp && ./a.out
ENTER P: 3
ENTER G: 7
Enter Private key 1: 5
Enter Private key 2: 4
The secret keys are: 1 and 1. They are same in value as expected.
```

Practical 10

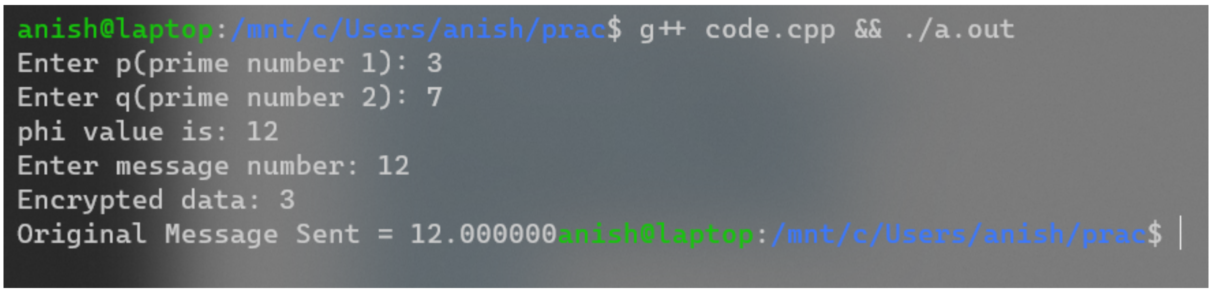
Aim - RSA Algorithm

Code

```
#include <iostream>
#include <cmath>
using namespace std;
int hcf(int, int);
int main()
{
    int p;
    int q;
    cout << "Enter p(prime number 1): ";
    cin >> p;
    cout << "Enter q(prime number 2): ";
    cin >> q;
    double n = p * q;
    double e = 2;
    double phi = (p - 1) * (q - 1);
    cout << "phi value is: " << phi << endl;
    while (e < phi)
    {
        if (hcf(e, phi) == 1)
            break;
        else
            e++;
    }
    int k = 2;
    double d = (1 + (k * phi)) / e;
    char ch;
    int msg;
    cout << "Enter message number: ";
    cin >> msg;
    double c = pow(msg, e);
    c = fmod(c, n);
    cout << "Encrypted data: " << c;
    double m = pow(c, d);
    m = fmod(m, n);
    printf("\nOriginal Message Sent = %lf", m);
    return 0;
}
int hcf(int a, int h)
```

```
{
    int temp;
    while (1)
    {
        temp = a % h;
        if (temp == 0)
            return h;
        a = h;
        h = temp;
    }
}
```

Output



```
anish@laptop:/mnt/c/Users/anish/prac$ g++ code.cpp && ./a.out
Enter p(prime number 1): 3
Enter q(prime number 2): 7
phi value is: 12
Enter message number: 12
Encrypted data: 3
Original Message Sent = 12.000000anish@laptop:/mnt/c/Users/anish/prac$ 
```