Ex.No: 01	NETWORK COMMANDS
Date:	

AIM:

NETWORKING COMMANDS:

- **NSLOOKUP:**

  NSLOOKUP command is used to troubleshoot network connectivity issiues in system. Using NSLOOKUP command , we can access the information related to system’s DNS server that is domain name and IP address.

- **NETSTAT:**

  NETSTAT command displays an overview of all connections in the device. The table shows details about connection protocol, address and current state of network.

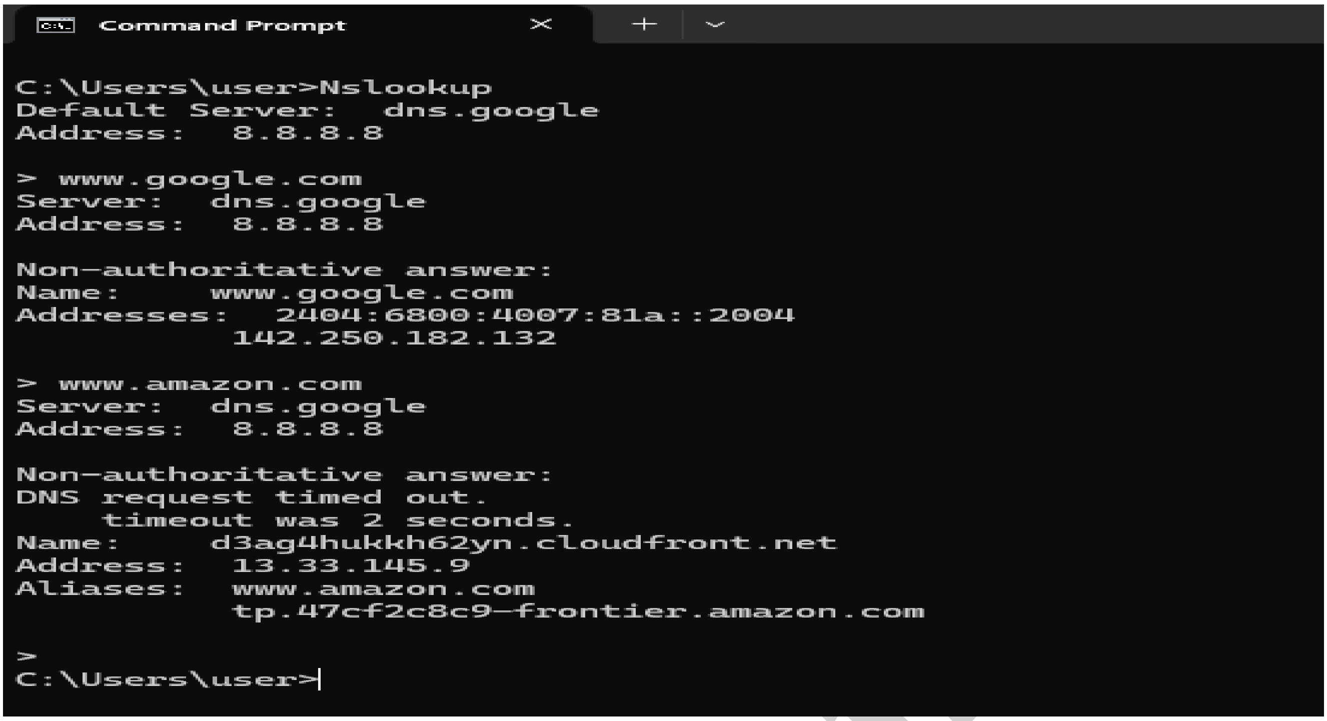
- **ROUTE:**

  Routing tables are used to direct packets from one subnet to another. ROUTE command returns routing table and one can make changes by commands such as Route add, Route delete, Route change which allows modifying Routing tables as a requirement.

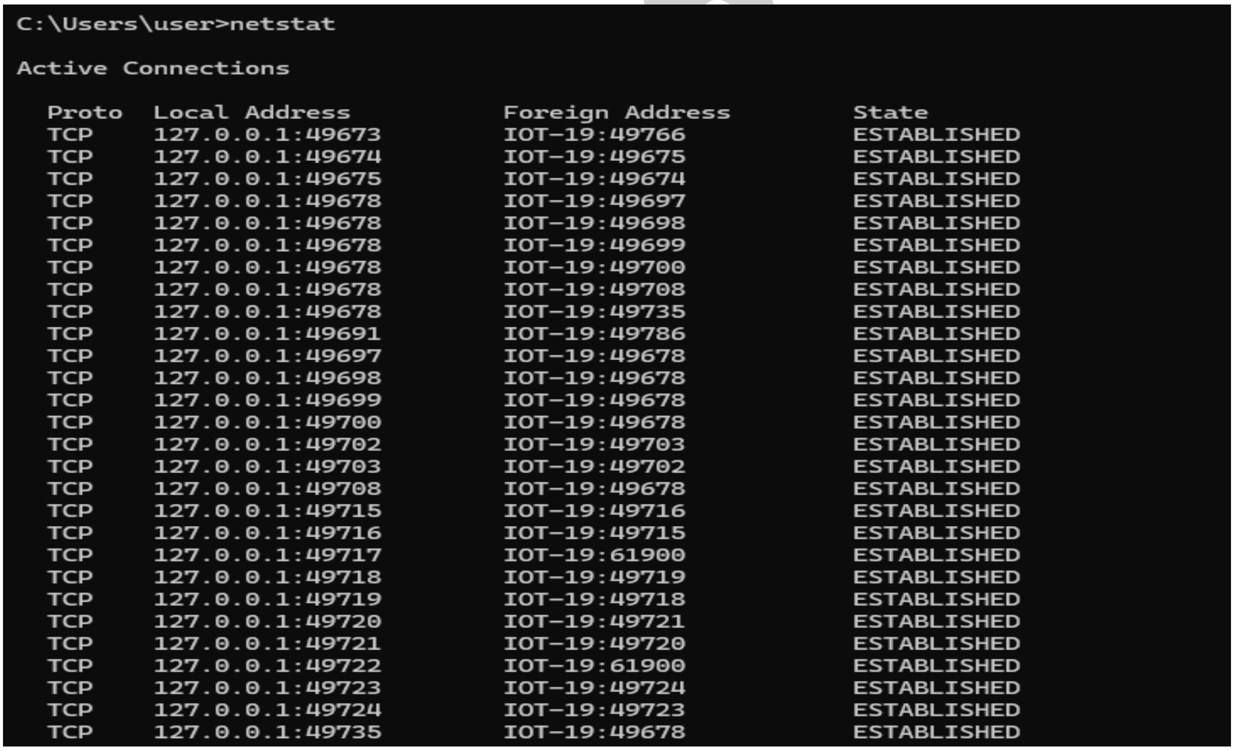
- **HOSTNAME:**

  HOSTNAME command displays hostname of the system.

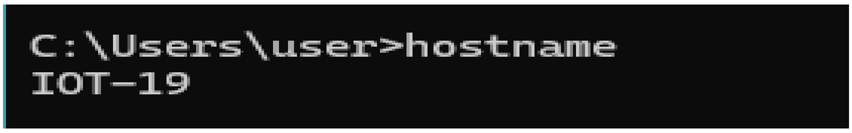
OUTPUT:



```
C:\Users\user>Nslookup
Default Server:  dns.google
Address:  8.8.8.8

> www.google.com
Server:  dns.google
Address:  8.8.8.8

Non-authoritative answer:
Name:    www.google.com
Addresses:  2404:6800:4007:81a::2004
          142.250.182.132

> www.amazon.com
Server:  dns.google
Address:  8.8.8.8

Non-authoritative answer:
DNS request timed out.
    timeout was 2 seconds.
Name:    d3ag4hukkh62yn.cloudfront.net
Address:  13.33.145.9
Aliases:  www.amazon.com
          tp.47cf2c8c9-frontier.amazon.com

>
C:\Users\user>
```



```
C:\Users\user>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:49673        IOT-19:49766           ESTABLISHED
  TCP    127.0.0.1:49674        IOT-19:49675           ESTABLISHED
  TCP    127.0.0.1:49675        IOT-19:49674           ESTABLISHED
  TCP    127.0.0.1:49678        IOT-19:49697           ESTABLISHED
  TCP    127.0.0.1:49678        IOT-19:49698           ESTABLISHED
  TCP    127.0.0.1:49678        IOT-19:49699           ESTABLISHED
  TCP    127.0.0.1:49678        IOT-19:49700           ESTABLISHED
  TCP    127.0.0.1:49678        IOT-19:49708           ESTABLISHED
  TCP    127.0.0.1:49678        IOT-19:49735           ESTABLISHED
  TCP    127.0.0.1:49691        IOT-19:49786           ESTABLISHED
  TCP    127.0.0.1:49697        IOT-19:49678           ESTABLISHED
  TCP    127.0.0.1:49698        IOT-19:49678           ESTABLISHED
  TCP    127.0.0.1:49699        IOT-19:49678           ESTABLISHED
  TCP    127.0.0.1:49700        IOT-19:49678           ESTABLISHED
  TCP    127.0.0.1:49702        IOT-19:49703           ESTABLISHED
  TCP    127.0.0.1:49703        IOT-19:49702           ESTABLISHED
  TCP    127.0.0.1:49708        IOT-19:49678           ESTABLISHED
  TCP    127.0.0.1:49715        IOT-19:49716           ESTABLISHED
  TCP    127.0.0.1:49716        IOT-19:49715           ESTABLISHED
  TCP    127.0.0.1:49717        IOT-19:61900           ESTABLISHED
  TCP    127.0.0.1:49718        IOT-19:49719           ESTABLISHED
  TCP    127.0.0.1:49719        IOT-19:49718           ESTABLISHED
  TCP    127.0.0.1:49720        IOT-19:49721           ESTABLISHED
  TCP    127.0.0.1:49721        IOT-19:49720           ESTABLISHED
  TCP    127.0.0.1:49722        IOT-19:61900           ESTABLISHED
  TCP    127.0.0.1:49723        IOT-19:49724           ESTABLISHED
  TCP    127.0.0.1:49724        IOT-19:49723           ESTABLISHED
  TCP    127.0.0.1:49735        IOT-19:49678           ESTABLISHED
```



```
C:\Users\user>hostname
IOT-19
```

- **ARP**:

  ARP displays and modifies the IP-to-Physical address translation tables used by address resolution protocol (ARP).

  The ARP command is used to access mapping structure of IP address to MAC address. This provides better understanding of transmission of packets in network channel.

  -a : displays ARP entities by interrogating current protocol data configuration.

- **TRACERT:**

  TRACERT command is used to trace the route during the transmission of data packet over to destination host and also provides us with 'hop' count during transmission.

  Using number of hops and hop Ip address , we can troubleshoot network issues and identify the point of the problem during transmission of data packet.

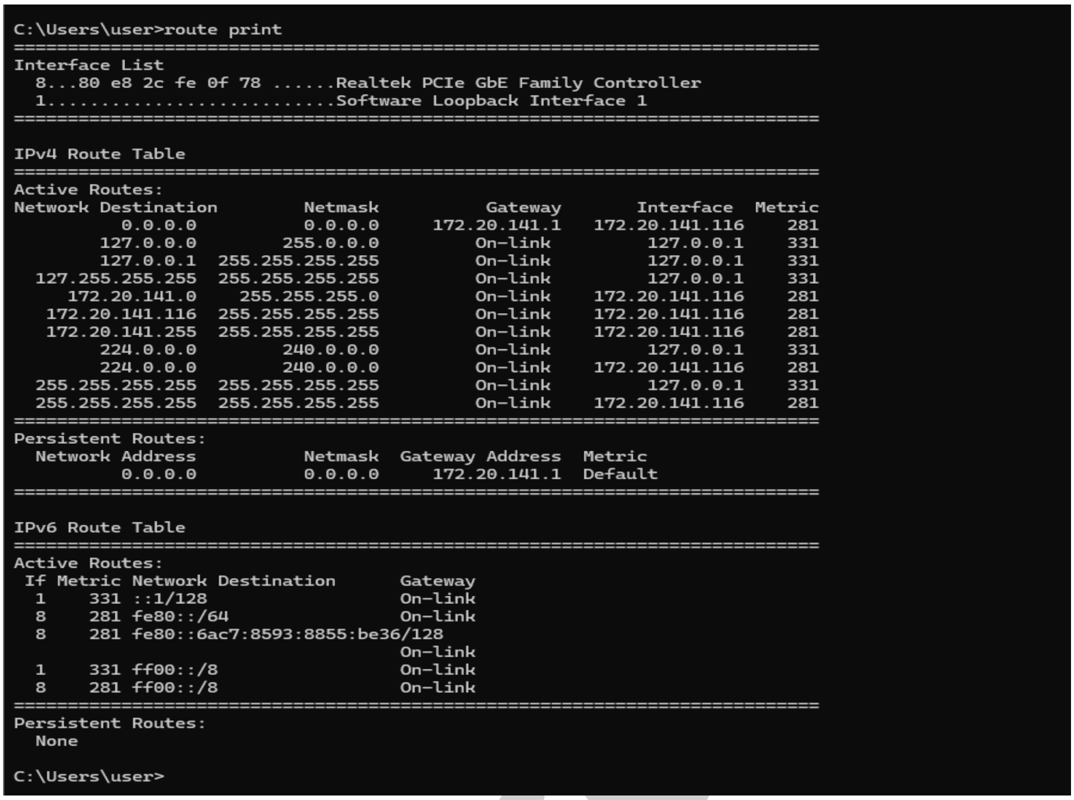
- **PING:**

  It allows the user to check the connectivity of our system to another host.

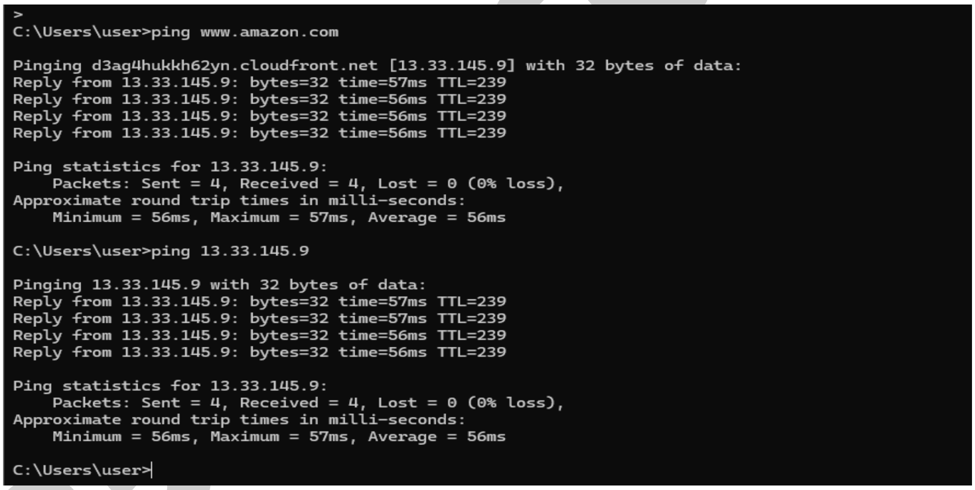
- **IPCONFIG:**

  Command IPCONFIG will display basic details about device's IP address

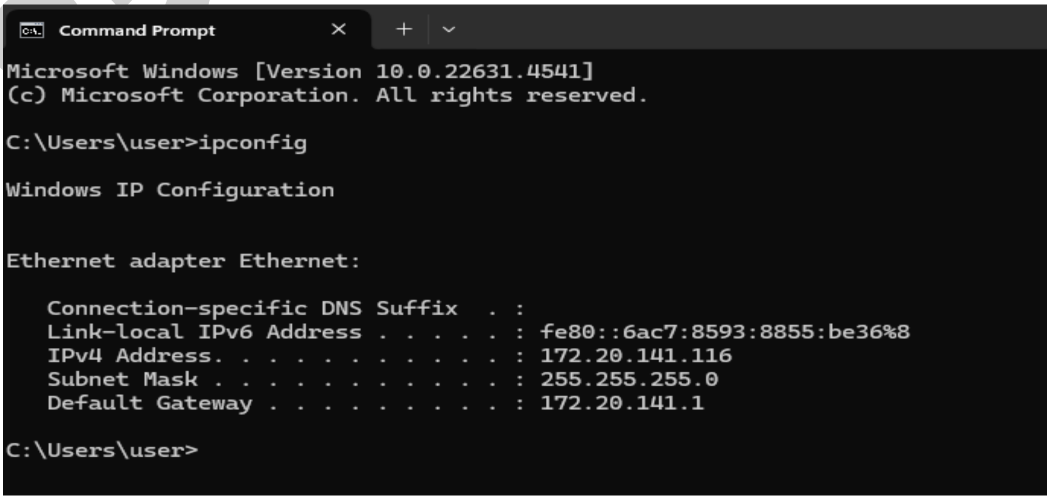
OUTPUT:



```
C:\Users\user>route print
===========================================================================
Interface List
  8...80 e8 2c fe 0f 78 ......Realtek PCIe GbE Family Controller
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0     172.20.141.1   172.20.141.116    281
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
     172.20.141.0    255.255.255.0         On-link    172.20.141.116    281
   172.20.141.116  255.255.255.255         On-link    172.20.141.116    281
   172.20.141.255  255.255.255.255         On-link    172.20.141.116    281
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link    172.20.141.116    281
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
  255.255.255.255  255.255.255.255         On-link    172.20.141.116    281
===========================================================================
Persistent Routes:
  Network Address          Netmask  Gateway Address  Metric
          0.0.0.0          0.0.0.0     172.20.141.1  Default
===========================================================================

IPv6 Route Table
===========================================================================
Active Routes:
 If Metric Network Destination      Gateway
  1    331 ::1/128                  On-link
  8    281 fe80::/64                On-link
  8    281 fe80::6ac7:8593:8855:be36/128
                                    On-link
  1    331 ff00::/8                 On-link
  8    281 ff00::/8                 On-link
===========================================================================
Persistent Routes:
  None

C:\Users\user>
```



```
>
C:\Users\user>ping www.amazon.com

Pinging d3ag4hukkh62yn.cloudfront.net [13.33.145.9] with 32 bytes of data:
Reply from 13.33.145.9: bytes=32 time=57ms TTL=239
Reply from 13.33.145.9: bytes=32 time=56ms TTL=239
Reply from 13.33.145.9: bytes=32 time=56ms TTL=239
Reply from 13.33.145.9: bytes=32 time=56ms TTL=239

Ping statistics for 13.33.145.9:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 56ms, Maximum = 57ms, Average = 56ms

C:\Users\user>ping 13.33.145.9

Pinging 13.33.145.9 with 32 bytes of data:
Reply from 13.33.145.9: bytes=32 time=57ms TTL=239
Reply from 13.33.145.9: bytes=32 time=57ms TTL=239
Reply from 13.33.145.9: bytes=32 time=56ms TTL=239
Reply from 13.33.145.9: bytes=32 time=56ms TTL=239

Ping statistics for 13.33.145.9:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 56ms, Maximum = 57ms, Average = 56ms

C:\Users\user>
```



```
Microsoft Windows [Version 10.0.22631.4541]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::6ac7:8593:8855:be36%8
   IPv4 Address. . . . . . . . . . . : 172.20.141.116
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 172.20.141.1

C:\Users\user>
```

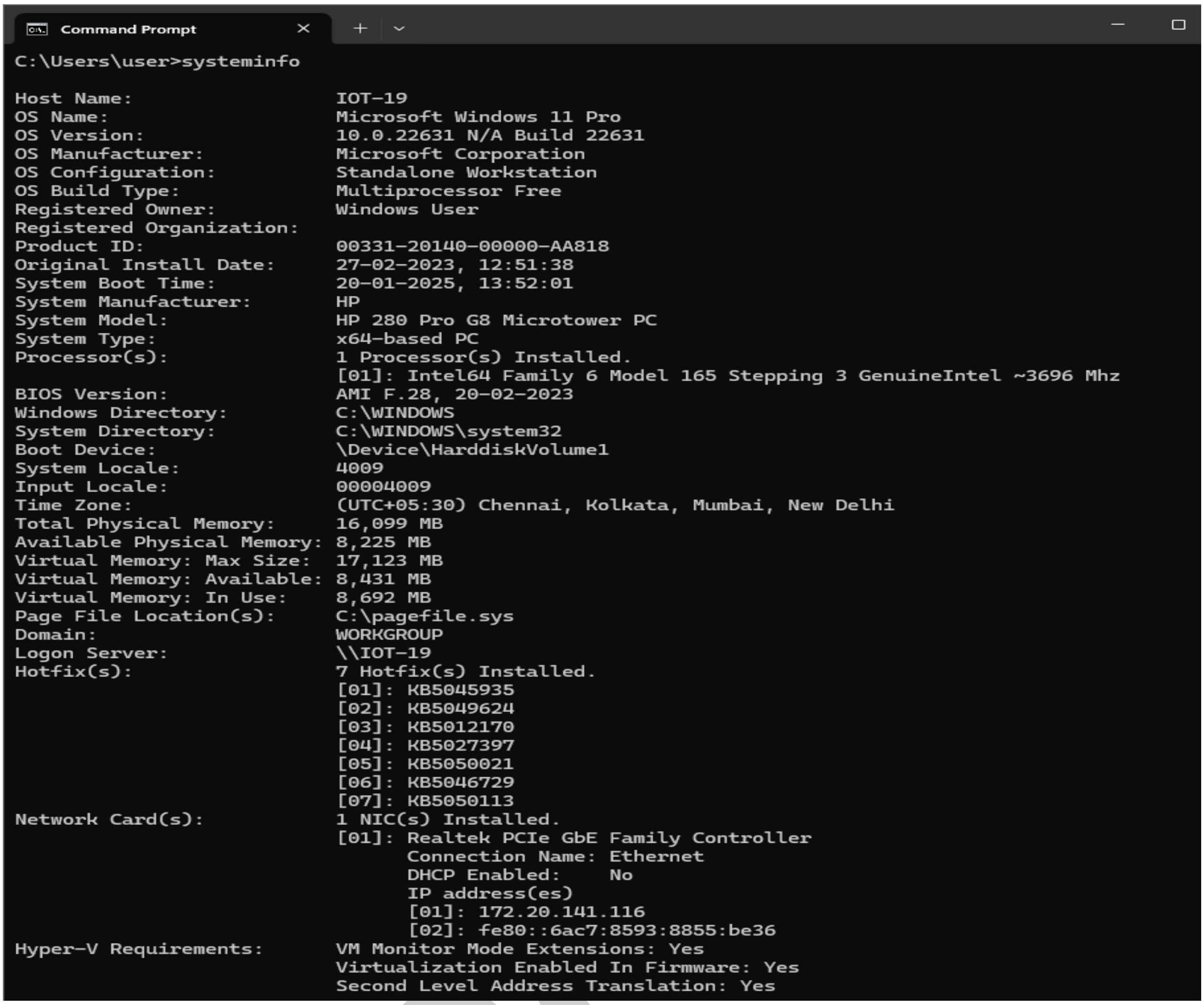
- **SYSTEMINFO:**

  Using SYSTEMINFO we can access the system’s hardware and software details such as processor data, booting data, window’s version etc..

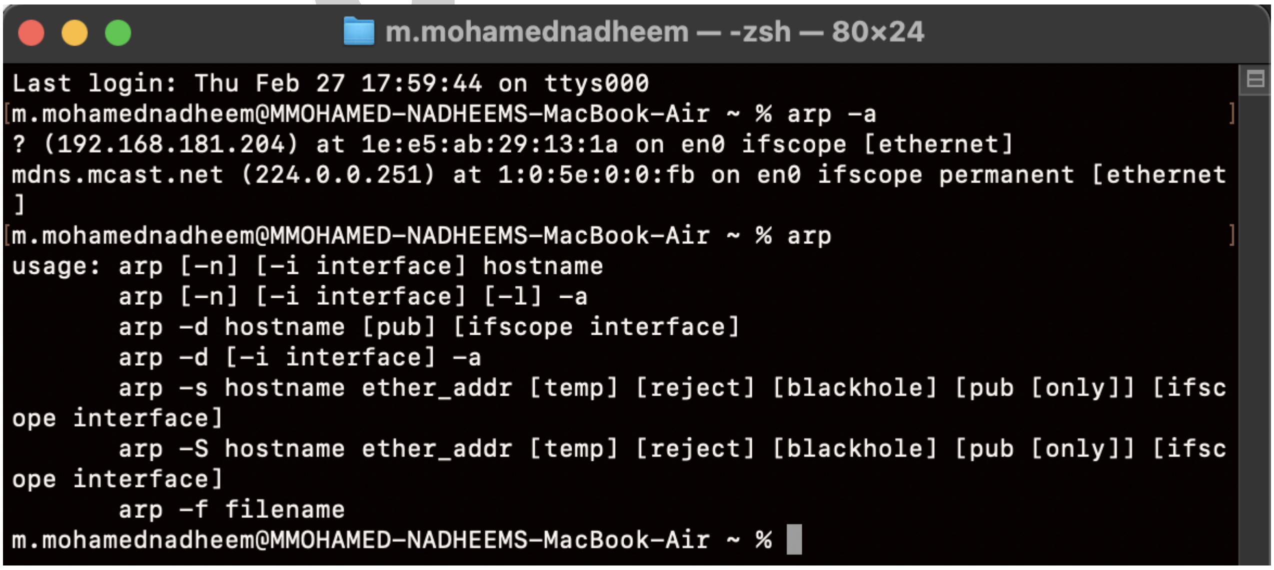
- **HELP:**

  HELP command allows list of available commands or detailed help information on a specified command . Without parameters , HELP lists and briefly describes every system command.

OUTPUT:



```
C:\Users\user>systeminfo

Host Name:                 IOT-19
OS Name:                   Microsoft Windows 11 Pro
OS Version:                10.0.22631 N/A Build 22631
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:          Windows User
Registered Organization:
Product ID:                00331-20140-00000-AA818
Original Install Date:     27-02-2023, 12:51:38
System Boot Time:          20-01-2025, 13:52:01
System Manufacturer:       HP
System Model:              HP 280 Pro G8 Microtower PC
System Type:               x64-based PC
Processor(s):              1 Processor(s) Installed.
                           [01]: Intel64 Family 6 Model 165 Stepping 3 GenuineIntel ~3696 Mhz
BIOS Version:              AMI F.28, 20-02-2023
Windows Directory:         C:\WINDOWS
System Directory:          C:\WINDOWS\system32
Boot Device:               \Device\HarddiskVolume1
System Locale:             4009
Input Locale:              00004009
Time Zone:                 (UTC+05:30) Chennai, Kolkata, Mumbai, New Delhi
Total Physical Memory:     16,099 MB
Available Physical Memory: 8,225 MB
Virtual Memory: Max Size:  17,123 MB
Virtual Memory: Available: 8,431 MB
Virtual Memory: In Use:    8,692 MB
Page File Location(s):     C:\pagefile.sys
Domain:                    WORKGROUP
Logon Server:              \\IOT-19
Hotfix(s):                 7 Hotfix(s) Installed.
                           [01]: KB5045935
                           [02]: KB5049624
                           [03]: KB5012170
                           [04]: KB5027397
                           [05]: KB5050021
                           [06]: KB5046729
                           [07]: KB5050113
Network Card(s):           1 NIC(s) Installed.
                           [01]: Realtek PCIe GbE Family Controller
                                 Connection Name: Ethernet
                                 DHCP Enabled:    No
                                 IP address(es)
                                 [01]: 172.20.141.116
                                 [02]: fe80::6ac7:8593:8855:be36
Hyper-V Requirements:      VM Monitor Mode Extensions: Yes
                           Virtualization Enabled In Firmware: Yes
                           Second Level Address Translation: Yes
```



```
Last login: Thu Feb 27 17:59:44 on ttys000
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % arp -a
? (192.168.181.204) at 1e:e5:ab:29:13:1a on en0 ifscope [ethernet]
mdns.mcast.net (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet
]
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % arp
usage: arp [-n] [-i interface] hostname
       arp [-n] [-i interface] [-l] -a
       arp -d hostname [pub] [ifscope interface]
       arp -d [-i interface] -a
       arp -s hostname ether_addr [temp] [reject] [blackhole] [pub [only]] [ifsc
ope interface]
       arp -S hostname ether_addr [temp] [reject] [blackhole] [pub [only]] [ifsc
ope interface]
       arp -f filename
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ %
```

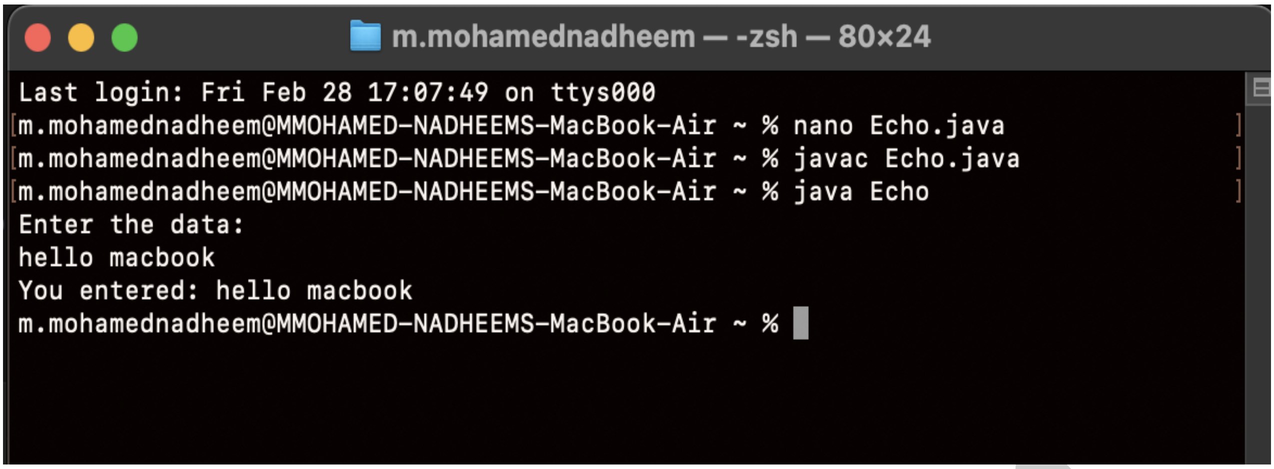
RESULT:

Ex.No: 02	**PROGRAM FOR ECHO**
Date:	

AIM:

ALGORITHM:

OUTPUT:



```
Last login: Fri Feb 28 17:07:49 on ttys000
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % nano Echo.java
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % javac Echo.java
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % java Echo
Enter the data:
hello macbook
You entered: hello macbook
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ %
```

PROGRAM:

```
 import java.util.Scanner;
 public class Echo
    {
      public static void main (String[] args)
      {
        String inData;
        Scanner scan = new Scanner( System.in );
        System.out.println("Enter the
        data:"); inData = scan.nextLine();
        System.out.println("You entered:" + inData );
      }
    }
```

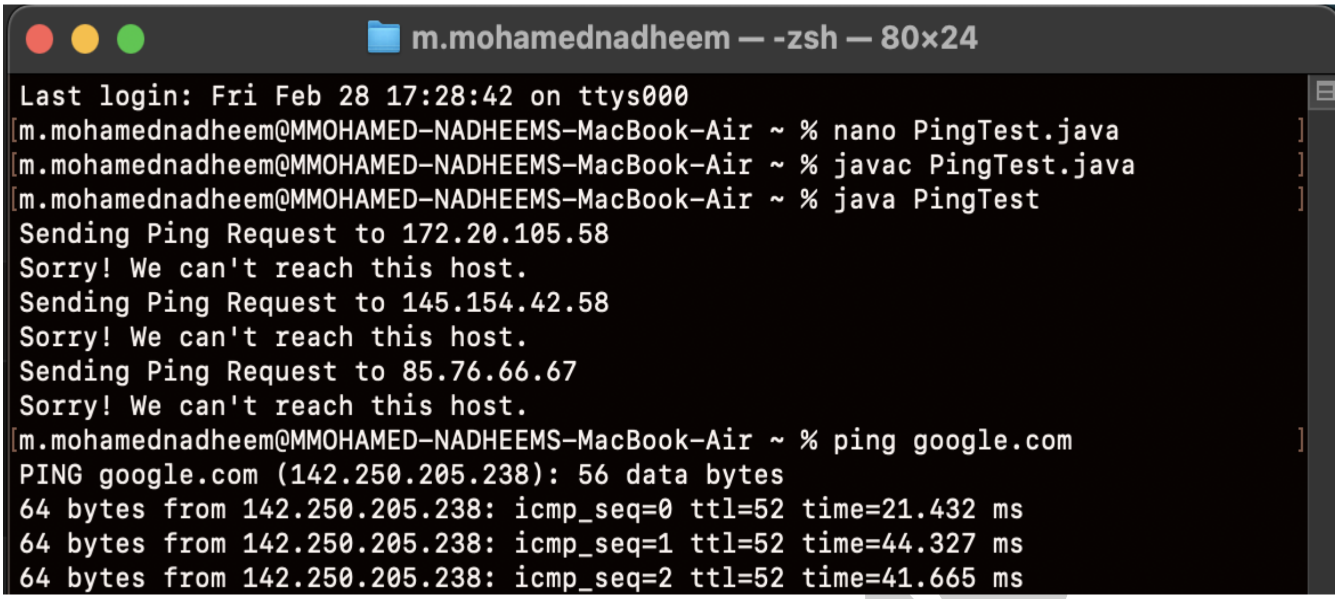
RESULT:

Ex.No: 03	**PROGRAM FOR PING**
Date:	

AIM:

ALGORITHM:

OUTPUT:



```
Last login: Fri Feb 28 17:28:42 on ttys000
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % nano PingTest.java
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % javac PingTest.java
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % java PingTest
Sending Ping Request to 172.20.105.58
Sorry! We can't reach this host.
Sending Ping Request to 145.154.42.58
Sorry! We can't reach this host.
Sending Ping Request to 85.76.66.67
Sorry! We can't reach this host.
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air ~ % ping google.com
PING google.com (142.250.205.238): 56 data bytes
64 bytes from 142.250.205.238: icmp_seq=0 ttl=52 time=21.432 ms
64 bytes from 142.250.205.238: icmp_seq=1 ttl=52 time=44.327 ms
64 bytes from 142.250.205.238: icmp_seq=2 ttl=52 time=41.665 ms
```

PROGRAM:

```
import java.io.IOException;
import java.net.InetAddress;
import java.net.UnknownHostException;

class PingTest {

    // Sends ping request to a provided IP address
    public static void sendPingRequest(String ipAddress) throws
UnknownHostException, IOException {
        InetAddress geek = InetAddress.getByName(ipAddress);
        System.out.println("Sending Ping Request to " + ipAddress);

        if (geek.isReachable(5000)) { // Timeout set to 5000 milliseconds
            System.out.println("Host is reachable.");
        } else {
            System.out.println("Sorry! We can't reach this host.");
        }
    }

    // Driver code
    public static void main(String[] args) throws UnknownHostException,
IOException {
        String ipAddress;

        ipAddress = "172.20.105.58";
        sendPingRequest(ipAddress);

        ipAddress = "145.154.42.58";
        sendPingRequest(ipAddress);

        ipAddress = "85.76.66.67";
        sendPingRequest(ipAddress);
```

RESULT:

Ex.No: 04	**SOCKET PROGRAM**
Date:	

AIM:

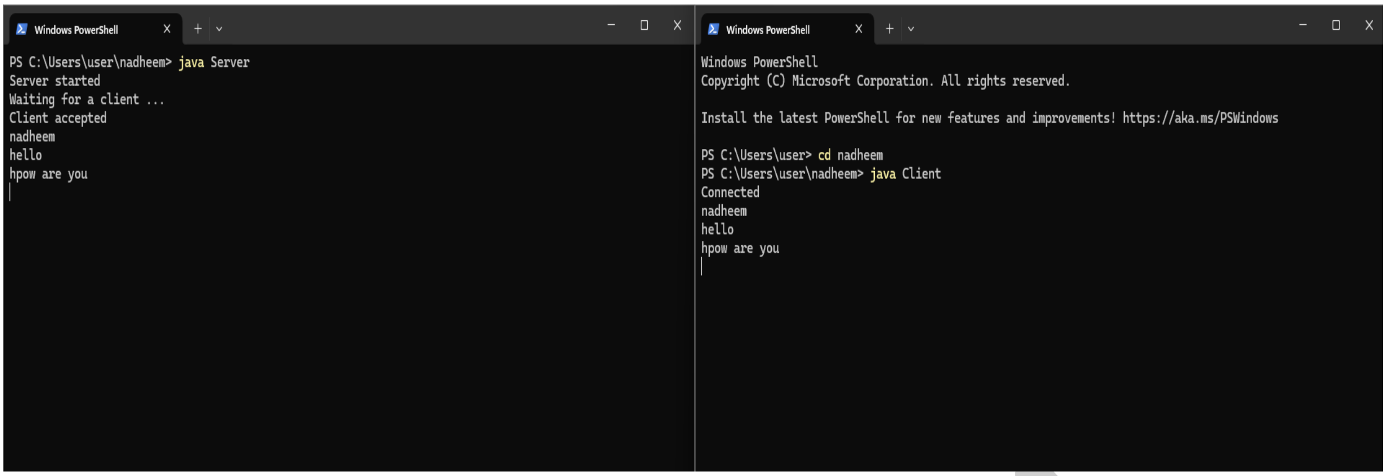
ALGORITHM:

PROGRAM:

Server program:

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

public class MyServer {
    public static void main(String[] args) {
        try {
            ServerSocket ss = new ServerSocket(6666);
            Socket s = ss.accept(); // establishes connection

            DataInputStream dis = new
     DataInputStream(s.getInputStream());
            String str = (String) dis.readUTF();

            System.out.println("message= " + str);
            ss.close();
        }
        catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

OUTPUT:



```
PS C:\Users\user\nadheem> java Server
Server started
Waiting for a client ...
Client accepted
nadheem
hello
hpow are you
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\user> cd nadheem
PS C:\Users\user\nadheem> java Client
Connected
nadheem
hello
hpow are you
```

Client program:

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

        public class MyClient {
           public static void main(String[] args) {
              try {
                 Socket s = new Socket("localhost", 6666);
                 DataOutputStream dout = new
           DataOutputStream(s.getOutputStream());

                 dout.writeUTF("Hello Server");
                 dout.flush();
                 dout.close();
                 s.close();
              }
              catch (Exception e) {
                 System.out.println(e);
              }
           }
}
```

RESULT:

Ex.No: 05	**CLIENT SERVER APPLICATION FOR CHAT**
Date:	

AIM:

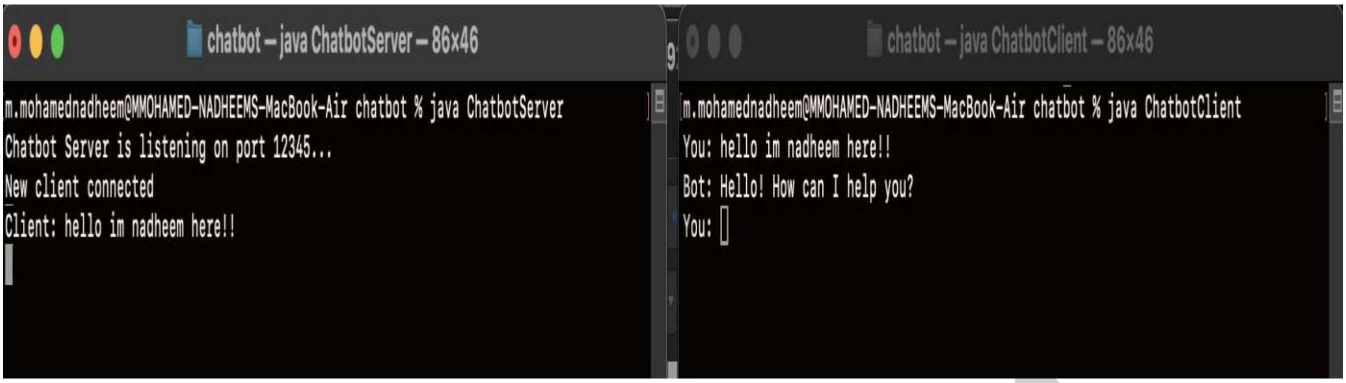
ALGORITHM:

PROGRAM:

Server.java

```
import java.net.*;
import java.io.*;
public class Server {
   public static void main(String[] args) {
      try {
         ServerSocket ss = new ServerSocket(1999);
         Socket s = ss.accept();
         DataOutputStream dos = new
DataOutputStream(s.getOutputStream());
         DataInputStream dis = new DataInputStream(s.getInputStream());
         BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
         while (!br.readLine().equals("quit")) {
            System.out.println("She says: " + dis.readUTF());
            dos.writeUTF(br.readLine());
         }
         ss.close();
      }
      catch (Exception ie) {
         ie.printStackTrace();
      }
   }}
```

OUTPUT:


chatbot – java ChatbotServer – 86×46
chatbot – java ChatbotClient – 86×46

```
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air chatbot % java ChatbotServer
Chatbot Server is listening on port 12345...
New client connected
Client: hello im nadheem here!!
```

```
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air chatbot % java ChatbotClient
You: hello im nadheem here!!
Bot: Hello! How can I help you?
You: 
```

Client.java

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

public class Client {
   public static void main(String[] args) {
      try {
         Socket s = new Socket("localhost", 1999);

         DataOutputStream dos = new
DataOutputStream(s.getOutputStream());
         DataInputStream dis = new DataInputStream(s.getInputStream());
         BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));

         while (!br.readLine().equals("quit")) {
            dos.writeUTF(br.readLine());
            System.out.println("He says: " + dis.readUTF());
            dos.flush();
         }

         s.close();
      }
      catch (IOException ie) {
         ie.printStackTrace();
      }
   }
```

RESULT:

Ex.No: 06	**PROGRAM TO IMPLEMENT REMOTE COMMAND EXECUTION**
Date:	

AIM:

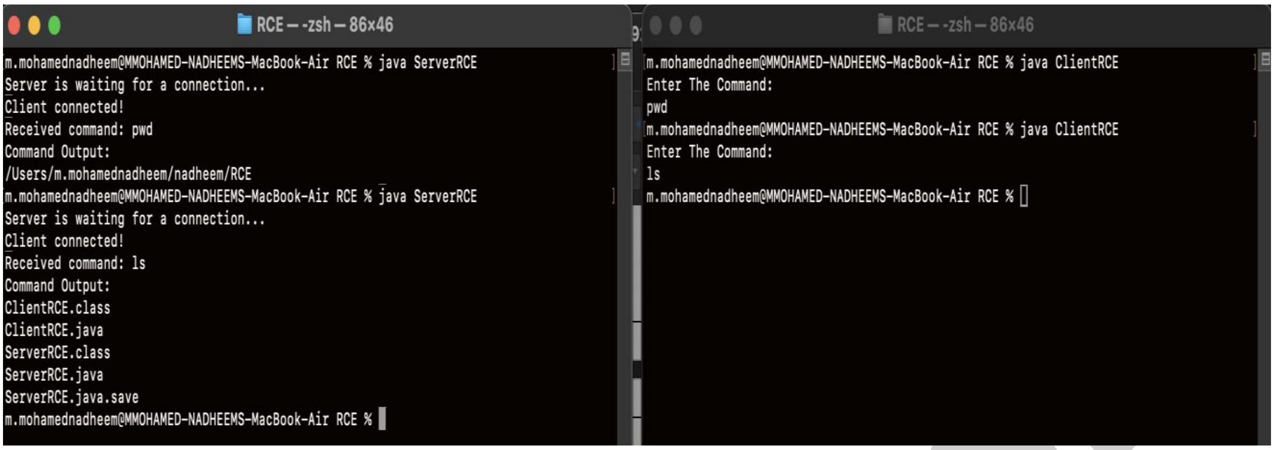
ALGORITHM:

PROGRAM:

CLIENTRCE

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


class ClientRCE {
    public static void main(String args[]) throws IOException {
        try {
            String str;
            Socket client = new Socket("127.0.0.1", 6555);
            PrintStream ps = new PrintStream(client.getOutputStream());
            BufferedReader br = new BufferedReader(new InputStreamReader(System.in));


            System.out.println("Enter The Command:");
            str = br.readLine();
            ps.println(str);
        } catch (IOException e) {
            System.out.println("Error");
        }
    }
}
```

OUTPUT:



```
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air RCE % java ServerRCE
Server is waiting for a connection...
Client connected!
Received command: pwd
Command Output:
/Users/m.mohamednadheem/nadheem/RCE
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air RCE % java ServerRCE
Server is waiting for a connection...
Client connected!
Received command: ls
Command Output:
ClientRCE.class
ClientRCE.java
ServerRCE.class
ServerRCE.java
ServerRCE.java.save
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air RCE %
```

```
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air RCE % java ClientRCE
Enter The Command:
pwd
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air RCE % java ClientRCE
Enter The Command:
ls
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air RCE %
```

SERVERRCE

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

class ServerRCE {
   public static void main(String args[]) throws IOException {
      try {
         String str;
         ServerSocket server = new ServerSocket(6555);
         Socket s = server.accept();
         BufferedReader br = new BufferedReader(new
InputStreamReader(s.getInputStream()));

         str = br.readLine();
         Runtime r = Runtime.getRuntime();
         Process p = r.exec(str);
      } catch (IOException e) {
         System.out.println("Error");
      }
   }
}
```

RESULT:

Ex.No: 07	**PROGRAM FOR FILE TRANSFER**
Date:	

AIM:

ALGORITHM:

PROGRAM:

SERVER

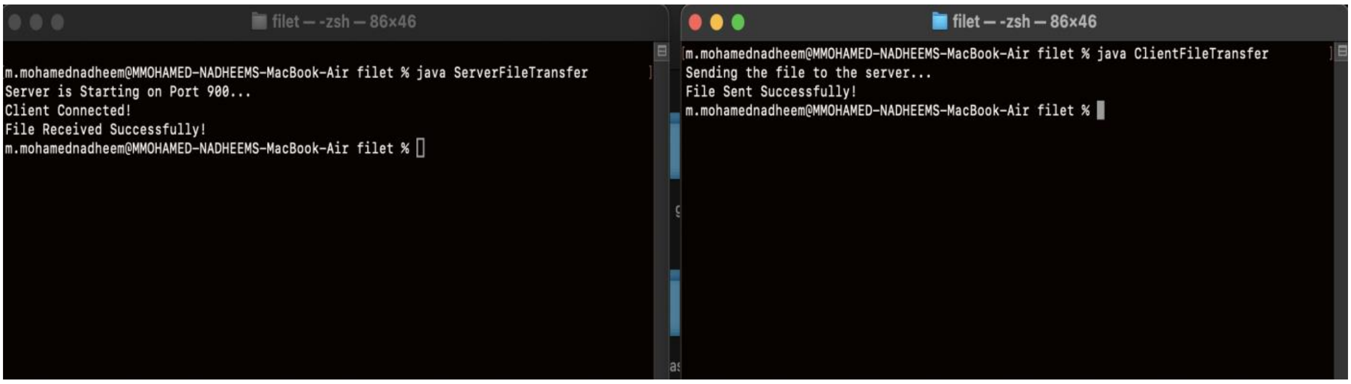
```
import java.io.*;
import java.net.*;

public class ServerFileTransfer {

    private static DataOutputStream dataOutputStream = null;
    private static DataInputStream dataInputStream = null;

    public static void main(String[] args) {
        try (ServerSocket serverSocket = new ServerSocket(900)) {
            System.out.println("Server is Starting on Port 900...");
            Socket clientSocket = serverSocket.accept();
            System.out.println("Client Connected!");

            dataInputStream = new 
DataInputStream(clientSocket.getInputStream());
            dataOutputStream = new 
DataOutputStream(clientSocket.getOutputStream());


receiveFile("/Users/m.mohamednadheem/Documents/received_useradd.png
");

            dataInputStream.close();
            dataOutputStream.close();
            clientSocket.close();
            serverSocket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void receiveFile(String fileName) throws Exception {
        int bytes = 0;
        FileOutputStream fileOutputStream = new FileOutputStream(fileName);
        long size = dataInputStream.readLong();
        byte[] buffer = new byte[4096];

        while (size > 0 && (bytes = dataInputStream.read(buffer, 0, (int) 
Math.min(buffer.length, size))) != -1) {
            fileOutputStream.write(buffer, 0, bytes);
```

```
size -= bytes;
        }
        System.out.println("File Received Successfully!");

        fileOutputStream.close();
    }
}
```

CLIENT

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

public class ClientFileTransfer {

    private static DataOutputStream dataOutputStream = null;
    private static DataInputStream dataInputStream = null;

    public static void main(String[] args) {
        try (Socket socket = new Socket("localhost", 900)) {
            dataInputStream = new DataInputStream(socket.getInputStream());
            dataOutputStream = new
DataOutputStream(socket.getOutputStream());

            System.out.println("Sending the file to the server...");
            sendFile("/Users/m.mohamednadheem/Documents/useradd.png");

            dataInputStream.close();
            dataOutputStream.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void sendFile(String filePath) throws Exception {
        int bytes = 0;
        File file = new File(filePath);
        FileInputStream fileInputStream = new FileInputStream(file);
        dataOutputStream.writeLong(file.length());

        byte[] buffer = new byte[4096];
        while ((bytes = fileInputStream.read(buffer)) != -1) {
            dataOutputStream.write(buffer, 0, bytes);
```

OUTPUT:



```
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air filet % java ServerFileTransfer
Server is Starting on Port 900...
Client Connected!
File Received Successfully!
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air filet %
```

```
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air filet % java ClientFileTransfer
Sending the file to the server...
File Sent Successfully!
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air filet %
```

```
   dataOutputStream.flush();
        }
        System.out.println("File Sent Successfully!");

        fileInputStream.close();
    }
}
```

RESULT:

Ex.No: 08	**PROGRAM TO IMPLEMENT CRC FOR ERROR HANDLING**
Date:	

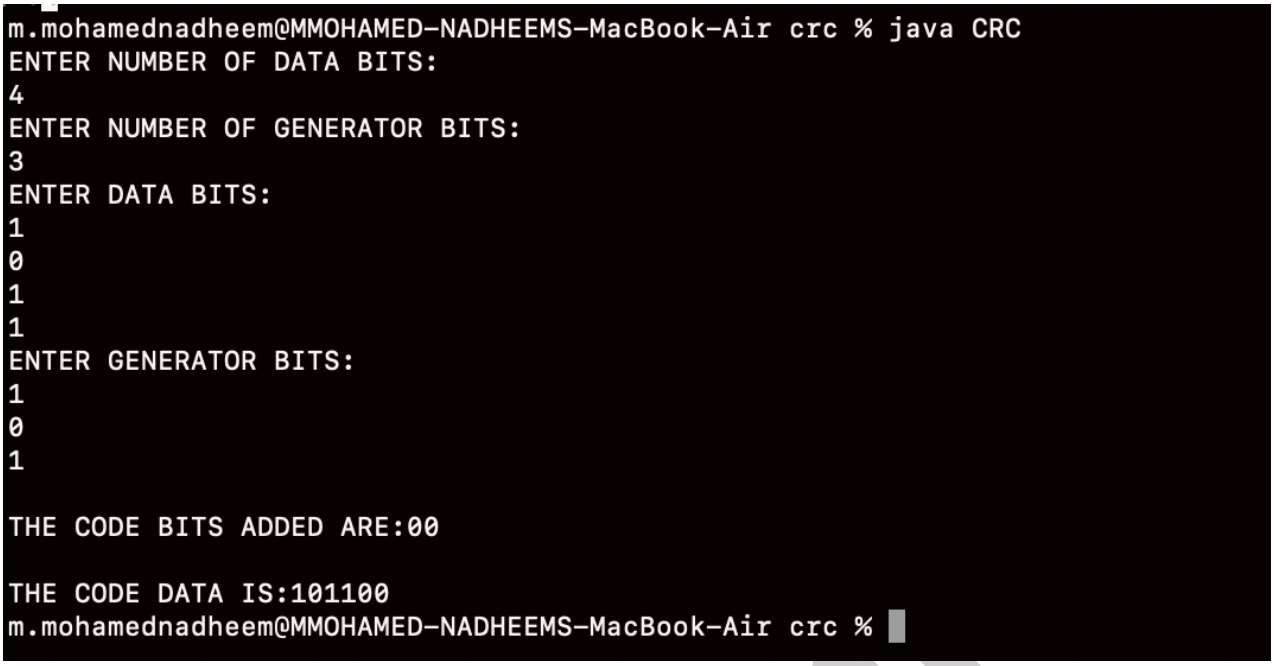
AIM:

ALGORITHM

PROGRAM:

```
import java.util.Scanner;

class CRC {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        int m, g[], n, d[], r[], msb, i, j, k;

        System.out.print("ENTER NUMBER OF DATA BITS:\n");
        n = sc.nextInt();
        System.out.print("ENTER NUMBER OF GENERATOR BITS:\n");
        m = sc.nextInt();

        d = new int[n + m];
        g = new int[m];

        System.out.print("ENTER DATA BITS:\n");
        for (i = 0; i < n; i++)
            d[i] = sc.nextInt();

        System.out.print("ENTER GENERATOR BITS:\n");
        for (j = 0; j < m; j++)
            g[j] = sc.nextInt();

        for (i = 0; i < m - 1; i++)
            d[n + i] = 0;

        r = new int[m + n];
        for (i = 0; i < m; i++)
            r[i] = d[i];

        int z[] = new int[m];
        for (i = 0; i < m; i++)
            z[i] = 0;

        for (i = 0; i < n; i++) {
            k = 0;
            msb = r[0];

            for (j = 1; j < m; j++) {
                if (msb == 0)
                    r[j - 1] = xor(r[j], z[k]);
                else
                    r[j - 1] = xor(r[j], g[k]);
```

OUTPUT:



```
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air crc % java CRC
ENTER NUMBER OF DATA BITS:
4
ENTER NUMBER OF GENERATOR BITS:
3
ENTER DATA BITS:
1
0
1
1
ENTER GENERATOR BITS:
1
0
1

THE CODE BITS ADDED ARE:00

THE CODE DATA IS:101100
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air crc %
```

```
       k++;


        }
            r[m - 1] = d[m + i];
        }

        System.out.print("\nTHE CODE BITS ADDED ARE:");
        for (i = n; i < n + m - 1; i++) {
            d[i] = r[i];
            System.out.print(d[i]);
        }

        System.out.println("");

        System.out.print("\nTHE CODE DATA IS:");
        for (i = 0; i < n + m - 1; i++) {
            System.out.print(d[i]);
        }

        System.out.println("");
    }

    public static int xor(int x, int y) {
        if (x == y)
            return 0;
        else
            return 1;
    }
}
```

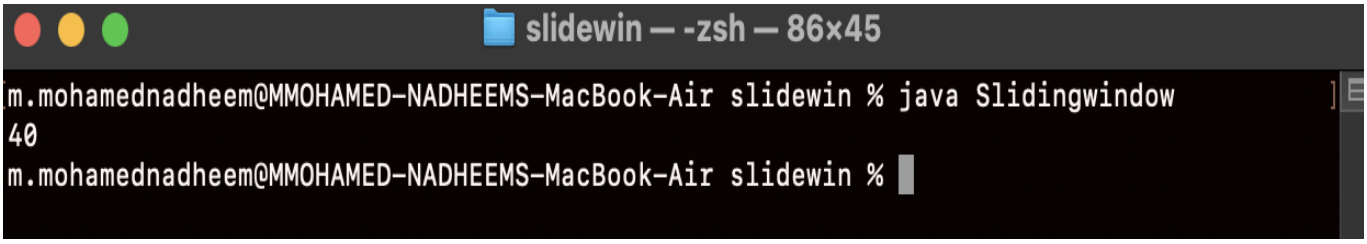
RESULT:

Ex.No: 09	**SLIDING WINDOW**
Date:	

AIM:

ALGORITHM:

OUTPUT:



```
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air slidewin % java Slidingwindow
40
m.mohamednadheem@MMOHAMED-NADHEEMS-MacBook-Air slidewin %
```

PROGRAM:

```
public class Slidingwindow {
    static int maxSum(int[] arr, int k) {
       int n = arr.length;
        if (n < k) {
            System.out.println("Invalid");
            return -1;
        }
        int window_sum = 0;
        for (int i = 0; i < k; i++)
            window_sum += arr[i];
        int max_sum = window_sum;
        for (int i = k; i < n; i++) {
            window_sum += (arr[i] - arr[i - k]);
            max_sum = Math.max(window_sum, max_sum);

        return max_sum;
    }
    public static void main(String[] args) {

        int k = 3;
        int[] arr = {16, 12, 9, 19, 11, 8};

        System.out.println(maxSum(arr, k));
    }
}
```

RESULT: