

# **REPORT on Computer Networks (V23CSL09)**

**B. TECH V Semester (2023 - 2027 Batch)**

*Academic Year: 2025-26*



**SRI VASAVI ENGINEERING COLLEGE (Autonomous)**

**Pedatadepalli, Tadepalligudem - 534101**

**Department of Computer Science & Engineering (Accredited by NBA)**

**SRI VASAVI ENGINEERING COLLEGE (Autonomous)**  
**PEDATADEPALLI, TADEPALLIGUDEM.**



## Certificate

This is to certify that this is a bona fide record of Practical Work done in **Computer Networks** Lab by Mr./Miss **Pamarthi Leela Venkata Siva Nageswararao** bearing Roll No. **24A85A0506** of **CSE** branch of **V** Semester during the academic year **2025-26**.

**No. of Experiments Done:**

**Faculty In charge:**

**Head of The Department:**

**Examiner 1:**

**Examiner 2:**

## INDEX

Experiment No.	Date	Name of the Experiment	Page	Signature of the Faculty
1	08/07/2025	Study of Network devices in detail and connect the computers in Local Area Network.	1-3	
2	08/07/2025	Develop a Program to implement the data link layer framing methods such as i) Character stuffing ii) bit stuffing	4-7	
3	15/07/2025	Develop a Program to implement data link layer framing method checksum.	8-9	
4	22/07/2025	Develop a program for Hamming Code generation for error detection and correction.	10-12	
5	29/07/2025	Develop a Program to implement on a data set of characters the three CRC polynomials – CRC 12, CRC 16 and CRC CCIP.	13-15	
6	29/07/2025	Develop a Program to implement Sliding window protocol for Goback N.	16-18	
7	05/08/2025	Develop a Program to implement Sliding window protocol for Selective repeat.	19-21	
8	05/08/2025	Develop a Program to implement Stop and Wait Protocol.	22-23	
9	14/10/2025	Develop a program for congestion control using leaky bucket algorithm	24-25	
10	23/09/2025	Develop a Program to implement Dijkstra's algorithm to compute the Shortest path through a graph.	26-28	
11	07/10/2025	Develop a Program to implement Distance vector routing algorithm by obtaining routing table at each node (Take an example subnet graph with weights indicating delay between nodes).	29-31	
12	09/09/2025	Wireshark i. Packet Capture Using Wire shark ii. Starting Wire shark iii. Viewing Captured Traffic iv. Analysis and Statistics & Filters.	32-36	
13	09/09/2025	Discover active hosts, open ports and running services in a network using Nmap	37-38	
14	09/09/2025	Find Operating System in a host using Nmap	39	

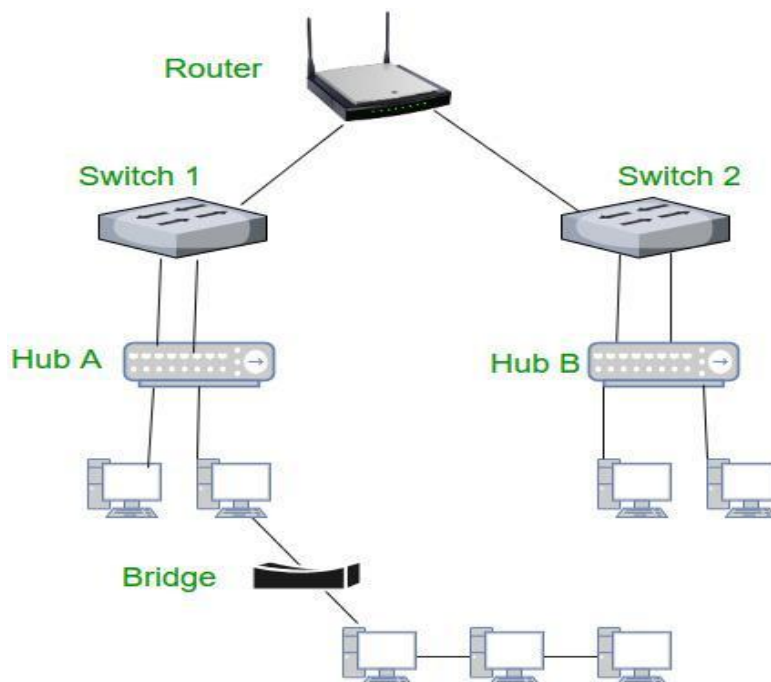
## 1. Study of Network devices in detail and connect the computers in Local Area Network.

### Description :-

Network devices are physical devices that allow hardware on a computer network to communicate and interact with one another. Examples include repeater, hub, bridge, switch, router, gateway, and NIC (Network Interface Card).

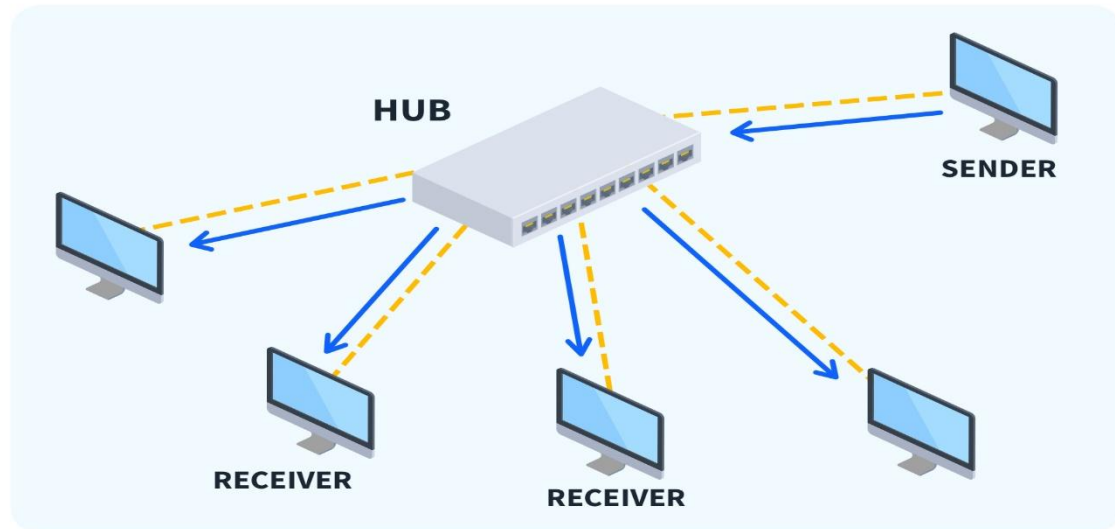
### Router :-

- A **router** is a networking device that forwards data packets between computer networks.
- It can connect one or more packet-switched networks or subnetworks.
- The router sends data packets to their intended IP addresses.
- It manages traffic between different networks.
- It also permits secure communication between networks.

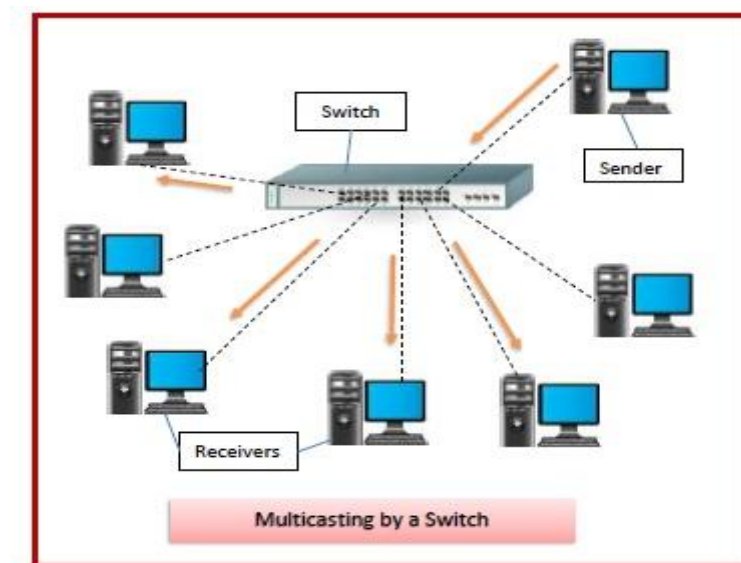


**HUB :-**

- A **hub** is a simple networking device that connects multiple devices in a network.
- It acts as a **central point** for data transmission.
- When data is received on one port, the hub **broadcasts it to all other ports**.

**Switch :-**

- A **network switch** is a crucial piece of networking hardware that connects wired devices within a network.
- It uses **packet switching** to receive, analyze, and forward data to the correct destination.
- This ensures **efficient communication** between devices.
- Essentially, a switch acts as a **traffic controller**, directing data to where it needs to go within a local network.



**Difference between hub and switch :-**

A **hub** and a **switch** are both networking devices used to connect multiple devices within a network, but they differ significantly in how they manage data. A hub is a basic device that simply broadcasts the data it receives to all connected devices, regardless of the intended recipient. This can lead to unnecessary network traffic, collisions, and slower performance. In contrast, a switch is more advanced and efficient. It uses the **MAC (Media Access Control) addresses** of devices to identify the correct destination and forwards data only to the intended device. This targeted communication reduces congestion, improves security, and ensures faster and more reliable data transfer. Because of this, switches are considered much more efficient and secure compared to hubs.

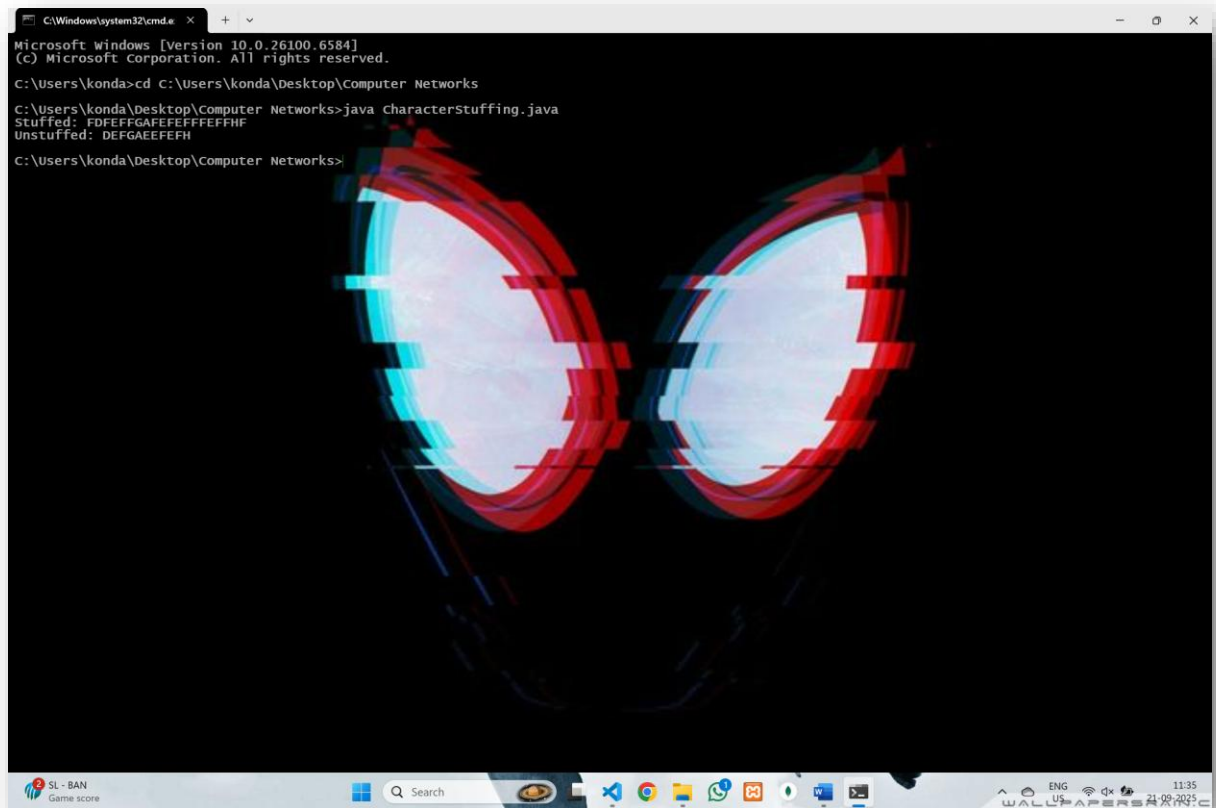
## 2. Develop a Program to implement the data link layer framing methods such as

### i) Character stuffing

### ii) bit stuffing.

#### i) Character stuffing :-

```
public class CharacterStuffing {
    public static void main(String[] args) {
        String input = "DEFGAEEFEFH";
        String stuffed = stuffData(input);
        String unstuffed = unstuffData(stuffed);
        System.out.println("Stuffed: " + stuffed);
        System.out.println("Unstuffed: " + unstuffed);
    }
    private static String stuffData(String s) {
        StringBuilder result = new StringBuilder();
        result.append('F');
        for (int i = 0; i < s.length(); i++) {
            char ch = s.charAt(i);
            if (ch == 'E' || ch == 'F')
                result.append('F');
            result.append(ch);
        }
        result.append('F');
        return result.toString();
    }
    private static String unstuffData(String s) {
        StringBuilder result = new StringBuilder();
        for (int i = 1; i < s.length() - 1; i++) {
            char ch = s.charAt(i);
            if (ch == 'F') {
                if (i + 1 < s.length() - 1) {
                    i++;
                    result.append(s.charAt(i));
                }
            } else {
                result.append(ch);
            }
        }
        return result.toString();
    }
}
```

**Output:-**

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.

C:\Users\konda>cd C:\Users\konda\Desktop\Computer Networks
C:\Users\konda\Desktop\Computer Networks>java characterStuffing.java
Stuffed: FDFEFGAFEFEEEEFFMF
Unstuffed: DEFGAEEFEFH
C:\Users\konda\Desktop\Computer Networks>
```



**ii) bit stuffing :-**

```

public class BitStuffing {
    public static void main(String[] args) {
        String s = "011101110111101111";
        Stuff s1 = new Stuff();
        s1.bitStuff(s);
        Unstuff s2 = new Unstuff();
        s2.bitUnstuff("01110111011110111110");
    }
}

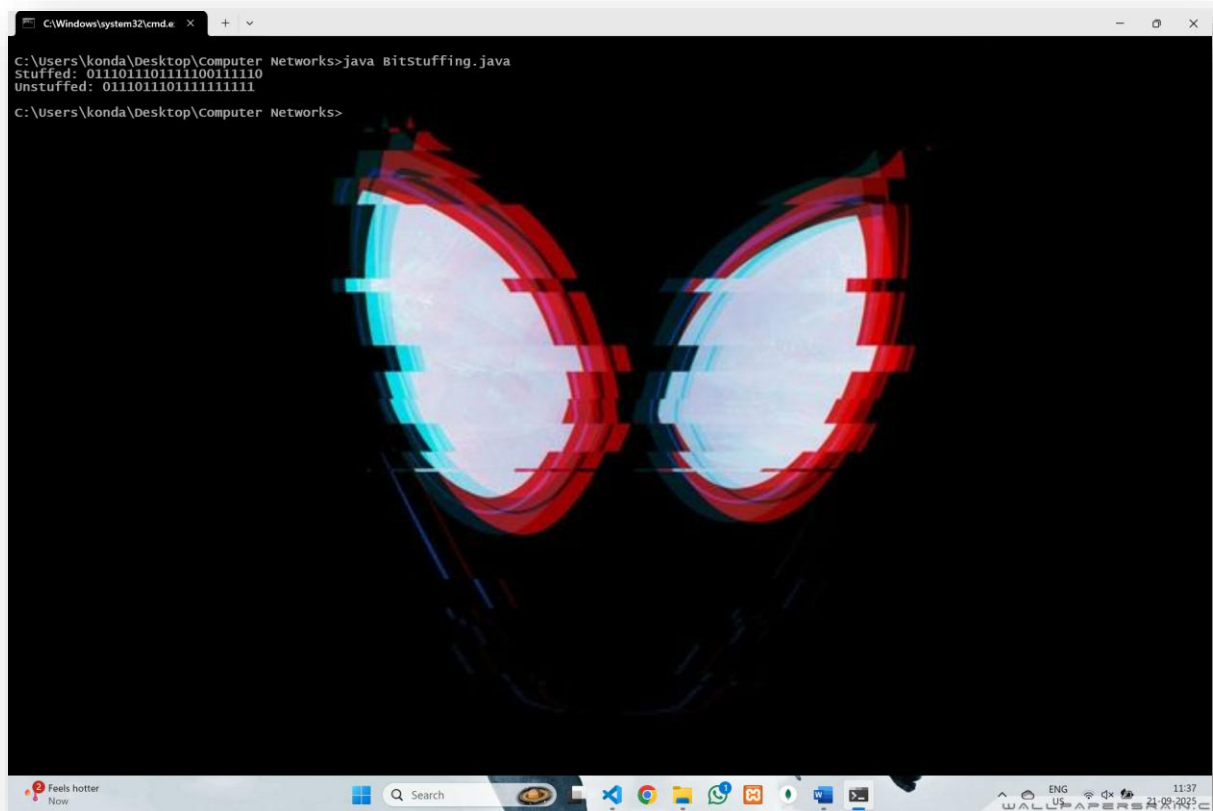
class Stuff {
    String bitStuff(String s) {
        int c = 0;
        StringBuilder res = new StringBuilder();
        for (int i = 0; i < s.length(); i++) {
            res.append(s.charAt(i));
            if (s.charAt(i) == '1') {
                c++;
                if (c == 5) {
                    res.append("0");
                    c = 0;
                }
            } else {
                c = 0;
            }
        }
        String result = res.toString();
        System.out.println("Stuffed: " + result);
        return result;
    }
}

class Unstuff {
    String bitUnstuff(String s) {
        int c = 0;
        StringBuilder res = new StringBuilder();
        for (int i = 0; i < s.length(); i++) {
            char ch = s.charAt(i);
            res.append(ch);
            if (ch == '1') {
                c++;
            }
        }
    }
}

```

```
        if (c == 5 && i + 1 < s.length() && s.charAt(i + 1) == '0') {  
            i++;  
            c = 0;  
        }  
    } else {  
        c = 0;  
    }  
}  
String result = res.toString();  
System.out.println("Unstuffed: " + result);  
return result;  
}  
}
```

### Output:-



### 3. Develop a Program to implement data link layer farming method checksum.

```
import java.util.*;
public class Checksum {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the length of the boolean string: ");
        int n = sc.nextInt();
        System.out.print("Enter boolean string: ");
        String data = sc.next();
        System.out.print("Enter size of blocks to divide: ");
        int blockSize = sc.nextInt();
        if (n % blockSize != 0) {
            System.out.println("Error: String length must be divisible by block
size");
            return;
        }
        if (data.length() != n) {
            System.out.println("Error: Actual string length doesn't match
specified length");
            return;
        }

        System.out.println("Divided blocks (sender): ");
        int numBlocks = n / blockSize;
        String[] blocks = new String[numBlocks];
        for (int i = 0; i < numBlocks; i++) {
            blocks[i] = data.substring(i * blockSize, (i + 1) * blockSize);
            System.out.println(blocks[i]);
        }
        String sum = blocks[0];
        for (int i = 1; i < numBlocks; i++) {
            sum = addBinary(sum, blocks[i]);
            System.out.println("Sum after adding block " + (i + 1) + ": " + sum);
        }
        String checksum = onesComplement(sum);
        System.out.println("Checksum: " + checksum);
        String finalSum = addBinary(sum, checksum);
        System.out.println("Receiver final sum: " + finalSum);
        if (allOnes(finalSum)) {
            System.out.println("Message received (No error)");
        } else {
            System.out.println("Message received (Error detected)");
        }
    }
}
```

```

        sc.close();
    }
    private static String addBinary(String a, String b) {
        int n = Math.max(a.length(), b.length());
        while (a.length() < n) a = "0" + a;
        while (b.length() < n) b = "0" + b;
        int carry = 0;
        StringBuilder result = new StringBuilder();
        for (int i = n - 1; i >= 0; i--) {
            int bit1 = a.charAt(i) - '0';
            int bit2 = b.charAt(i) - '0';
            int sum = bit1 + bit2 + carry;
            result.insert(0, sum % 2);
            carry = sum / 2;
        }
        if (carry > 0) {
            return addBinary(result.toString(), "1");
        }

        return result.toString();
    }
    private static String onesComplement(String s) {
        StringBuilder sb = new StringBuilder();
        for (char c : s.toCharArray()) {
            sb.append(c == '0' ? '1' : '0');
        }
        return sb.toString();
    }
    private static boolean allOnes(String s) {
        for (char c : s.toCharArray()) {
            if (c != '1') return false;
        }
        return true;
    }
}

```

### **Output :-**

```

C:\Windows\system32\cmd.exe
c:\Users\konda\Desktop\Computer Networks>java Checksum.java
Enter the length of the boolean string: 8
Enter boolean string: 11001111
Enter size of blocks to divide: 4
Divided blocks (sender):
1100
1111
Sum after adding block 2: 1100
Checksum: 0011
Receiver final sum: 1111
Message received (No error)
c:\Users\konda\Desktop\Computer Networks>

```

#### 4. Develop a program for Hamming Code generation for error detection and correction.

```
import java.util.*;

public class HammingCode {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Sender Side :");
        System.out.print("Enter no of data bits : ");
        int m = sc.nextInt();
        int[] data = new int[m];
        System.out.println("Enter data Bits : ");
        for (int i = 0; i < m; i++) {
            data[i] = sc.nextInt();
        }

        int r = 0;
        while (Math.pow(2, r) < m + r + 1) {
            r++;
        }

        int[] hamming = new int[m + r + 1];
        int dataIndex = 0;

        for (int i = 1; i < hamming.length; i++) {
            if (isPowerOfTwo(i)) {
                hamming[i] = 0;
            } else {
                hamming[i] = data[dataIndex];
                dataIndex++;
            }
        }

        for (int i = 0; i < r; i++) {
            int pos = (int) Math.pow(2, i);
```

```

    int val = 0;
    for (int k = 1; k < hamming.length; k++) {
        if (((k >> i) & 1) == 1 && k != pos) {
            val ^= hamming[k];
        }
    }
    hamming[pos] = val;
}

```

```

System.out.print("Genrated hamming code : ");
for (int i = 1; i < hamming.length; i++) {
    System.out.print(hamming[i]);
}
System.out.println();

```

```

System.out.println("Reciver side:");
System.out.print("Enter Recieved message : ");
String recv = sc.next();
int[] recvd = new int[recv.length() + 1];

```

```

for (int i = 1; i <= recv.length(); i++) {
    recvd[i] = recv.charAt(i - 1) - '0';
}

```

```

int errorPos = 0;
for (int i = 0; i < r; i++) {
    int pos = (int) Math.pow(2, i);
    int val = 0;
    for (int k = 1; k < recvd.length; k++) {
        if (((k >> i) & 1) == 1) {
            val ^= recvd[k];
        }
    }
    if (val != 0) {
        errorPos += pos;
    }
}

```

```

    }
}

if (errorPos == 0) {
    System.out.println("Message Recieved correctly ");
} else {
    System.out.println("Error Detected at position " + errorPos);
    System.out.print("Corrected message : ");
    recvd[errorPos] = 1 - recvd[errorPos];
    for (int i = 1; i < recvd.length; i++) {
        System.out.print(recvd[i]);
    }
    System.out.println();
}

sc.close();
}

private static boolean isPowerOfTwo(int n) {
    return n > 0 && (n & (n - 1)) == 0;
}
}

```

### Output :-



```

C:\Windows\system32\cmd.exe
C:\Users\konda\Desktop\Computer Networks>java HammingCode.java
Sender side :
Enter no of data bits : 4
Enter data Bits :
1
0
1
1
Genrated hamming code : 0110011
Receiver side:
Enter Recieved message : 0110011
Message Recieved correctly

C:\Users\konda\Desktop\Computer Networks>java HammingCode.java
Sender Side :
Enter no of data bits : 4
Enter data Bits :
1
0
1
1
Genrated hamming code : 0110011
Receiver side:
Enter Recieved message : 0010011
Error Detected at position 2
Corrected message : 0110011
C:\Users\konda\Desktop\Computer Networks>

```

### 5. Develop a Program to implement on a data set of characters the three CRC polynomials – CRC 12, CRC 16 and CRC CCIP.

```
import java.util.*;

public class CRC {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter Frame Size : ");

        int frameSize = sc.nextInt();

        System.out.print("Enter Frame bits (Space Separated 0/1) : ");

        int[] frame = new int[frameSize];

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

            frame[i] = sc.nextInt();

        }

        System.out.print("Enter the Highest Power of x in the generator : ");

        int gp = sc.nextInt();

        int[] generator = new int[gp + 1];

        System.out.println("Enter Coefficient for Generator Polynomial :");

        for (int i = gp; i >= 0; i--) {

            System.out.print("Coefficient of x^" + i + " : ");

            generator[i] = sc.nextInt();

        }

        int[] transmitted = new int[frameSize + gp];

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

            transmitted[i] = frame[i];

        }

        int[] remainder = divide(transmitted.clone(), generator);

        System.out.println("Sender Side : ");

        System.out.print("Frame : ");

        for (int i = 0; i < frameSize; i++) System.out.print(frame[i]);

        System.out.println();

        System.out.print("Generator : ");

        for (int i = gp; i >= 0; i--) System.out.print(generator[i]);

    }

}
```



```
System.out.println();
System.out.print("CRC : ");
for (int i = 0; i < gp; i++) System.out.print(remainder[i]);
System.out.println();
for (int i = 0; i < gp; i++) {
    transmitted[frameSize + i] = remainder[i];
}
System.out.print("Transmitted Frame : ");
for (int i = 0; i < transmitted.length; i++)
    System.out.print(transmitted[i]);
System.out.println();
System.out.println("Receiver Side:");
System.out.print("Enter Received Frame Size : ");
int rsize = sc.nextInt();
System.out.print("Enter Received Frame : ");
int[] received = new int[rsize];
for (int i = 0; i < rsize; i++) {
    received[i] = sc.nextInt();
}
int[] rem2 = divide(received.clone(), generator);
System.out.print("Remainder :");
for (int i = 0; i < gp; i++) System.out.print(rem2[i]);
System.out.println();
boolean error = false;
for (int i = 0; i < gp; i++) {
    if (rem2[i] != 0) {
        error = true;
        break;
    }
}
if (error) {
    System.out.println("Error Detected");
}
```

```

    } else {

        System.out.println("NO error Detected");

    }

    sc.close();

}

private static int[] divide(int[] dividend, int[] divisor) {

    int gp = divisor.length - 1;

    for (int i = 0; i <= dividend.length - divisor.length; i++) {

        if (dividend[i] == 1) {

            for (int j = 0; j < divisor.length; j++) {

                dividend[i + j] ^= divisor[j];

            }

        }

    }

    int[] remainder = new int[gp];

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

        remainder[i] = dividend[dividend.length - gp + i];

    }

    return remainder;

} }

```

### Output :-

```

C:\Users\konda\Desktop\Computer Networks>java CRC.java
Enter Frame Size : 12
Enter Frame bits (Space Separated 0/1) : 1
1
0
1
1
1
0
1
1
1
1
0
1
Enter the Highest Power of x in the generator : 4
Enter Coefficient for Generator Polynomial :
Coefficient of x^4 : 1
Coefficient of x^3 : 0
Coefficient of x^2 : 1
Coefficient of x^1 : 0
Coefficient of x^0 : 1
Sender Side :
Frame : 110111011101
Generator : 10101
CRC : 0000
Transmitted Frame : 1101110111010000
Receiver Side:
Enter Recieved Frame Size : 16
Enter Recieved Frame : 1
1
0
1
1
1
0
1
1
1
0
1
0
0
0
0
Remainder :0000
NO error Detected
C:\Users\konda\Desktop\Computer Networks>

```

## 6. Develop a Program to implement Sliding window protocol for Goback N.

```
import java.util.*;

public class GoBackNARQ {
    private int totalFrames;
    private int windowSize;
    private int base = 0;
    private int nextFrame = 0;
    private long timerStart;
    private final int TIMEOUT = 3000;
    private boolean timerRunning = false;
    private Scanner sc = new Scanner(System.in);

    public GoBackNARQ(int totalFrames, int windowSize) {
        this.totalFrames = totalFrames;
        this.windowSize = windowSize;
    }

    public void SendFrames() {
        while (base < totalFrames) {
            while (nextFrame < base + windowSize && nextFrame < totalFrames) {
                System.out.println("Sending frame " + nextFrame);
                nextFrame++;
                if (!timerRunning) {
                    timerRunning = true;
                    timerStart = System.currentTimeMillis();
                }
            }

            System.out.print("Was Ack for frame " + base + " received ? (y/n) : ");

            String ackInput = sc.next();
            if (ackInput.equals("y")) {
                System.out.println("Ack received for frame " + base);
                base++;
            }
        }
    }
}
```

```
        if (base == nextFrame) {
            timerRunning = false;
        } else {
            timerStart = System.currentTimeMillis();
        }
    } else if (ackInput.equals("n")) {
        System.out.println("Ack for frame " + base + " is lost waiting for
timeout...");

        try {
            Thread.sleep(TIMEOUT);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        int end = Math.min(base + windowSize, totalFrames);
        System.out.println("Time Out ! Resending frames from " + base + "
to " + (end - 1));

        for (int i = base; i < end; i++) {
            System.out.println("Sending frame " + i);
        }
        nextFrame = end;
        timerStart = System.currentTimeMillis();
    } else {
        System.out.println("Invalid input please enter y or n");
    }
}

System.out.println("All frames are Acknowledged");
}

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

```

System.out.print("Enter total number of frames to send : ");

int totalFrames = sc.nextInt();

System.out.print("Enter Window Size : ");

int windowSize = sc.nextInt();

if (windowSize > totalFrames)
    windowSize = totalFrames;

GoBackNARQ gbn = new GoBackNARQ(totalFrames, windowSize);

gbn.SendFrames();

sc.close();

}}

```

### Output :-

```

C:\Windows\system32\cmd.exe
C:\Users\konda\Desktop\Computer Networks>java GoBackNARQ.java
Enter total number of frames to send : 7
Enter window size : 3
Sending frame 0
Sending frame 1
Sending frame 2
Was Ack for frame 0 received ? (y/n) : y
Ack received for frame 0
Sending frame 3
Was Ack for frame 1 received ? (y/n) : n
Ack for frame 1 is lost waiting for timeout...
Time Out ! Resending frames from 1 to 3
Sending frame 1
Sending frame 2
Sending frame 3
Was Ack for frame 1 received ? (y/n) : y
Ack received for frame 1
Sending frame 4
Was Ack for frame 2 received ? (y/n) : y
Ack received for frame 2
Sending frame 5
Was Ack for frame 3 received ? (y/n) : y
Ack received for frame 3
Sending frame 6
Was Ack for frame 4 received ? (y/n) : y
Ack received for frame 4
Was Ack for frame 5 received ? (y/n) : y
Ack received for frame 5
Was Ack for frame 6 received ? (y/n) : y
Ack received for frame 6
All frames are Acknowledged
C:\Users\konda\Desktop\Computer Networks>

```

## 7. Develop a Program to implement Sliding window protocol for Selective repeat.

```
import java.util.*;

public class SelectiveRepeatProtocol {
    private final int totalFrames, windowSize;
    private final boolean[] sent, acked;
    private int base = 1;

    public SelectiveRepeatProtocol(int totalFrames, int windowSize) {
        this.totalFrames = totalFrames;
        this.windowSize = windowSize;
        this.sent = new boolean[totalFrames + 1];
        this.acked = new boolean[totalFrames + 1];
    }

    private void Sender(Scanner sc) {
        while (base <= totalFrames) {
            int windowEnd = Math.min(base + windowSize - 1, totalFrames);

            for (int i = base; i <= windowEnd; i++) {
                if (!sent[i]) {
                    Send(i);
                }
            }

            for (int f = base; f <= windowEnd; f++) {
                if (!acked[f]) {
                    System.out.print("Ack for frame " + f + " received ? (y/n) :");
                    String response = sc.next().trim();

                    if (response.equalsIgnoreCase("y")) {
                        acked[f] = true;
                        System.out.println("Sender : Ack received for frame " + f);
                    }
                }
            }
        }
    }
}
```

```

        while (base <= totalFrames && acked[base]) {
            base++;
        }

        int nextToSend = Math.min(base + windowSize - 1, totalFrames);
        if (nextToSend > windowEnd && nextToSend <= totalFrames) {
            if (!sent[nextToSend]) {
                Send(nextToSend);
            }
        } else if (base > totalFrames) {
            System.out.println("Sender : there is no frame to send");
        }

    } else if (response.equalsIgnoreCase("n")) {
        System.out.println("Sender:ack lost for frame " + f);
        System.out.println("Resending frame " + f);
        Send(f);
    }
}

}

}

}

private void Send(int frameNo) {
    System.out.println("Sender : Sending frame " + frameNo);
    sent[frameNo] = true;
    delay(100);
}

private void delay(int ms) {
    try {
        Thread.sleep(ms);
    }
}

```

```

    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter total number of frames : ");
    int totalFrames = sc.nextInt();
    System.out.print("Enter Window Size : ");
    int windowSize = sc.nextInt();

    SelectiveRepeatProtocol srp = new SelectiveRepeatProtocol(totalFrames,
windowSize);

    srp.Sender(sc);
    sc.close(); } }

```

### Output :-

```

C:\Windows\system32\cmd.exe
C:\Users\konda\Desktop\Computer Networks>java SelectiveRepeatProtocol.java
Enter total number of frames : 6
Enter window size : 2
Sender : Sending frame 1
Sender : Sending frame 2
Ack for frame 1 received ? (y/n) : y
Sender : Ack received for frame 1
Sender : Sending frame 3
Ack for frame 2 received ? (y/n) : n
Sender:ack lost for frame2
Resending frame 2
Sender : Sending frame 2
Ack for frame 2 received ? (y/n) : y
Sender : Ack received for frame 2
Sender : Sending frame 4
Ack for frame 3 received ? (y/n) : y
Sender : Ack received for frame 3
Sender : Sending frame 5
Ack for frame 4 received ? (y/n) : y
Sender : Ack received for frame 4
Sender : Sending frame 6
Ack for frame 5 received ? (y/n) : y
Sender : Ack received for frame 5
Ack for frame 6 received ? (y/n) : y
Sender : Ack received for frame 6
Sender : there is no frame to send
C:\Users\konda\Desktop\Computer Networks>

```



## 8. Develop a Program to implement Stop and Wait Protocol.

```
import java.util.*;

public class StopAndWaitProtocol {
    private final int totalFrames;
    private final int TIMEOUT = 3000;

    public StopAndWaitProtocol(int totalFrames) {
        this.totalFrames = totalFrames;
    }

    public void sender(Scanner sc) {
        for (int frameNo = 1; frameNo <= totalFrames; frameNo++) {
            boolean ackReceived = false;
            while (!ackReceived) {
                System.out.println("Sender : Sending frame " + frameNo);
                delay(500);
                System.out.print("Was Ack for frame " + frameNo + " received ?
(y/n) : ");

                String response = sc.next().trim();
                if (response.equalsIgnoreCase("y")) {
                    System.out.println("Sender : Ack received for frame " +
frameNo);

                    ackReceived = true;
                } else if (response.equalsIgnoreCase("n")) {
                    System.out.println("Sender : Ack lost for frame " + frameNo +
" waiting for timeout...");
                    try {
                        Thread.sleep(TIMEOUT);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                    System.out.println("Time Out ! Resending frame " + frameNo);
                } else {
                    System.out.println("Invalid input please enter y or n");
                }
            }
        }
    }
}
```

```

    }

}

System.out.println("All frames sent and acknowledged successfully");

}

private void delay(int ms) {

    try {

        Thread.sleep(ms);

    } catch (InterruptedException e) {

        e.printStackTrace();

    }

}

public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);

    System.out.print("Enter total number of frames to send : ");

    int totalFrames = sc.nextInt();

    StopAndWaitProtocol sap = new StopAndWaitProtocol(totalFrames);

    sap.sender(sc);

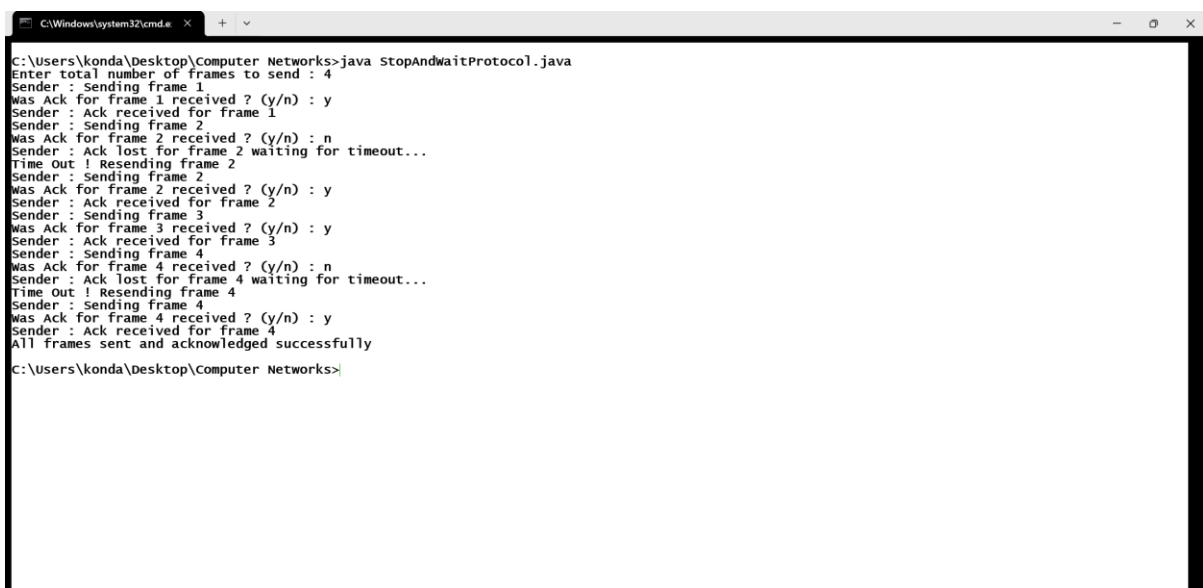
    sc.close();

}

}

```

### Output :-



```

C:\Windows\system32\cmd.exe
C:\Users\konda\Desktop\Computer Networks>java StopAndWaitProtocol.java
Enter total number of frames to send : 4
Sender : Sending frame 1
was Ack for frame 1 received ? (y/n) : y
Sender : Ack received for frame 1
Sender : Sending frame 2
was Ack for frame 2 received ? (y/n) : n
Sender : Ack lost for frame 2 waiting for timeout...
Time Out ! Resending frame 2
Sender : Sending frame 2
was Ack for frame 2 received ? (y/n) : y
Sender : Ack received for frame 2
Sender : Sending frame 3
was Ack for frame 3 received ? (y/n) : y
Sender : Ack received for frame 3
Sender : Sending frame 4
was Ack for frame 4 received ? (y/n) : n
Sender : Ack lost for frame 4 waiting for timeout...
Time Out ! Resending frame 4
Sender : Sending frame 4
was Ack for frame 4 received ? (y/n) : y
Sender : Ack received for frame 4
All frames sent and acknowledged successfully
C:\Users\konda\Desktop\Computer Networks>

```

**9. Develop a program for congestion control using leaky bucket algorithm .**

```

import java.util.LinkedList;
import java.util.Queue;
import java.util.Scanner;
public class LeakyBucket
{
    private int capacity;
    private int leakRate;
    private Queue<Integer> buffer;
    public LeakyBucket(int capacity, int leakRate) {
        this.capacity = capacity;
        this.leakRate = leakRate;
        this.buffer = new LinkedList<>();
    }
    public void addPackets(int packets) {
        System.out.println("Incoming packets: " + packets);
        System.out.println("Buffer size before adding: " + buffer.size());
        int dropped = 0;
        int added = 0;
        for (int i = 0; i < packets; i++) {
            if (buffer.size() < capacity) {
                buffer.add(1);
                added++;
                System.out.println("Packet added to bucket. Current size: " +
buffer.size());
            } else {
                dropped++;
            }
        }
        System.out.println("Packets added: " + added);
        if (dropped > 0) {
            System.out.println("Bucket overflow! Packets dropped: " + dropped);
        }
    }
    public void leak() {
        int leaked = 0;
        for (int i = 0; i < leakRate; i++) {
            if (!buffer.isEmpty()) {
                buffer.poll();
                leaked++;
            }
        }
        System.out.println("Packets processed (leaked): " + leaked);
        System.out.println("Buffer size after leaking: " + buffer.size());
    }
}

```

```

        System.out.println("-----");
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter bucket size (capacity): ");
        int capacity = sc.nextInt();
        System.out.print("Enter leak rate (packets processed per unit time): ");
        int leakRate = sc.nextInt();
        LeakyBucket leakyBucket = new LeakyBucket(capacity, leakRate);
        System.out.println("\nStarting simulation. Enter -1 to stop.");
        while (true) {
            System.out.print("Enter incoming packets: ");
            int packets = sc.nextInt();
            if (packets == -1) {
                System.out.println("Simulation ended.");
                break;
            }
            leakyBucket.addPackets(packets);
            leakyBucket.leak();
        }
        sc.close();
    }
}

```

### Output :-

```

PS C:\Users\konda\Desktop\Computer Networks> java .\LeakyBucket.java
Enter bucket size (capacity): 10
Enter leak rate (packets processed per unit time): 2

Starting simulation. Enter -1 to stop.
Enter incoming packets: 3
Incoming packets: 3
Buffer size before adding: 0
Packet added to bucket. Current size: 1
Packet added to bucket. Current size: 2
Packet added to bucket. Current size: 3
Packets added: 3
Packets processed (leaked): 2
Buffer size after leaking: 1
-----
Enter incoming packets: 2
Incoming packets: 2
Buffer size before adding: 1
Packet added to bucket. Current size: 2
Packet added to bucket. Current size: 3
Packets added: 2
Packets processed (leaked): 2
Buffer size after leaking: 1
-----
Enter incoming packets: 4
Incoming packets: 4
Buffer size before adding: 1
Packet added to bucket. Current size: 2
Packet added to bucket. Current size: 3
Packet added to bucket. Current size: 4
Packet added to bucket. Current size: 5
Packets added: 4
Packets processed (leaked): 2
Buffer size after leaking: 3
-----
Enter incoming packets: 2
Incoming packets: 2
Buffer size before adding: 3
Packet added to bucket. Current size: 4
Packet added to bucket. Current size: 5
Packets added: 2
Packets processed (leaked): 2
Buffer size after leaking: 3
-----
Enter incoming packets: 3
Incoming packets: 3
Buffer size before adding: 3
Packet added to bucket. Current size: 4
Packet added to bucket. Current size: 5
Packet added to bucket. Current size: 6
Packets added: 3
Packets processed (leaked): 2
Buffer size after leaking: 4
-----
Enter incoming packets: -1
Simulation ended.
PS C:\Users\konda\Desktop\Computer Networks>

```

## 10. Develop a Program to implement Dijkstra's algorithm to compute the Shortest path through a graph.

```
import java.util.*;

public class Dijkstra {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter No of Vertices :");
        int n = sc.nextInt();
        sc.nextLine();
        char[] arr = new char[n];
        Map<Character, Integer> vertexMap = new HashMap<>();
        for (int i = 0; i < n; i++) {
            System.out.println("Enter the name of Vertex " + i + ":");
            arr[i] = sc.next().charAt(0);
            vertexMap.put(arr[i], i);
        }
        List<int[]> edges = new ArrayList<>();
        System.out.println("Enter the number of edges:");
        int e = sc.nextInt();
        sc.nextLine();
        for (int i = 0; i < e; i++) {
            System.out.print("Enter edge (e.g., a -> b): ");
            String edge = sc.nextLine();
            String[] parts = edge.split("->");
            if (parts.length != 2) {
                System.out.println("Invalid input format. Use 'a -> b'.");
                return;
            }
            char u = parts[0].trim().charAt(0);
            char v = parts[1].trim().charAt(0);
            if (!vertexMap.containsKey(u) || !vertexMap.containsKey(v)) {
                System.out.println("Invalid vertices. Ensure they are within the vertex list.");
                return;
            }
            System.out.print("Enter the weight for edge " + u + " -> " + v + ":");
            int w = sc.nextInt();
            sc.nextLine();
            if (w < 0) {
                System.out.println("Negative weights are not allowed in Dijkstra's algorithm.");
                return;
            }
        }
    }
}
```

```

        edges.add(new int[]{vertexMap.get(u), vertexMap.get(v), w});
    }
    if (n <= 0) {
        System.out.println("Number of vertices must be positive.");
        return;
    }
    int[] dist = new int[n];
    boolean[] visited = new boolean[n];
    int[] prev = new int[n];
    Arrays.fill(dist, Integer.MAX_VALUE);
    Arrays.fill(visited, false);
    Arrays.fill(prev, -1);
    System.out.print("Enter the source vertex (e.g., a): ");
    char sourceChar = sc.next().charAt(0);
    int src = vertexMap.get(sourceChar);
    dist[src] = 0;
    PriorityQueue<int[]> pq = new PriorityQueue<>((a, b) -> a[1] - b[1]);
    pq.add(new int[]{src, 0});
    List<List<int[]>> adj = new ArrayList<>(n);
    for (int i = 0; i < n; i++) {
        adj.add(new ArrayList<>());
    }
    for (int[] edge : edges) {
        adj.get(edge[0]).add(new int[]{edge[1], edge[2]});
    }
    while (!pq.isEmpty()) {
        int[] curr = pq.poll();
        int u = curr[0];
        int d = curr[1];
        if (visited[u]) continue;
        visited[u] = true;
        for (int[] neighbor : adj.get(u)) {
            int v = neighbor[0];
            int w = neighbor[1];
            if (dist[v] > dist[u] + w) {
                dist[v] = dist[u] + w;
                prev[v] = u;
                pq.add(new int[]{v, dist[v]});
            }
        }
    }
    System.out.print("Enter the destination vertex (e.g., b): ");
    char destChar = sc.next().charAt(0);
    int dest = vertexMap.get(destChar);
    if (dist[dest] == Integer.MAX_VALUE) {

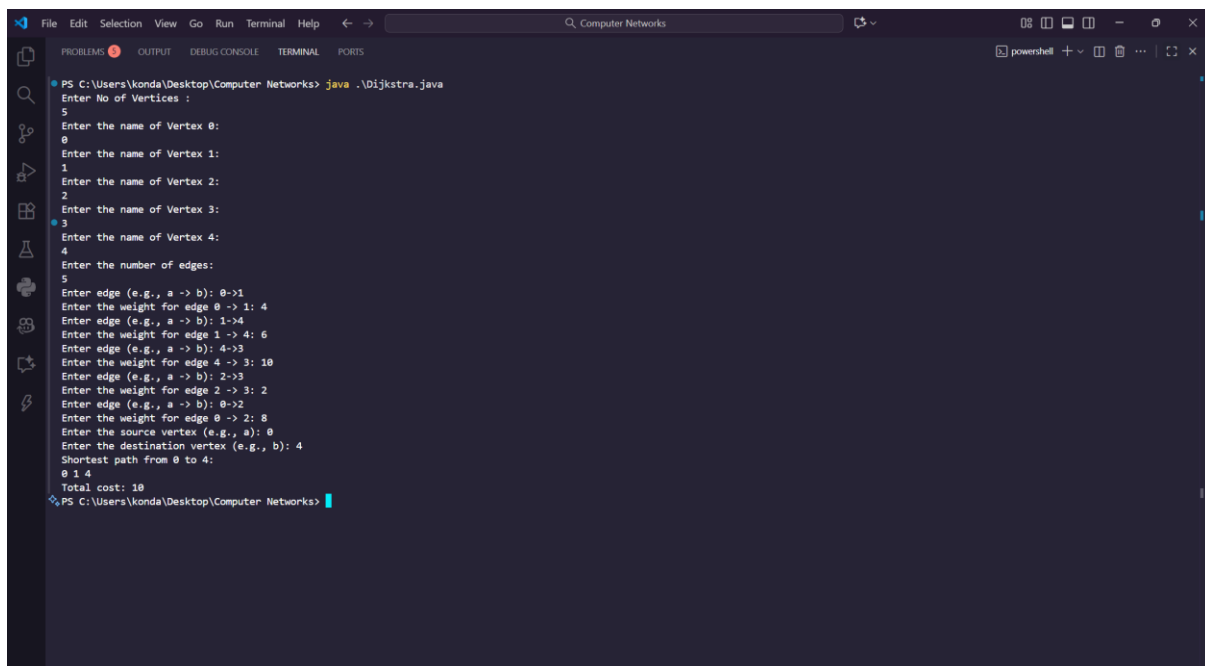
```

```

        System.out.println("Destination " + destChar + " is unreachable from "
+ sourceChar);
    } else {
        System.out.println("Shortest path from " + sourceChar + " to " +
destChar + ":");
        List<Character> path = new ArrayList<>();
        int current = dest;
        while (current != -1) {
            path.add(arr[current]);
            current = prev[current];
        }
        Collections.reverse(path);
        for (char c : path) {
            System.out.print(c + " ");
        }
        System.out.println();
        System.out.println("Total cost: " + dist[dest]);
    }
}
}

```

### Output :-



```

PS C:\Users\konda\Desktop\Computer Networks> java .\Dijkstra.java
Enter No of Vertices :
5
Enter the name of Vertex 0:
0
Enter the name of Vertex 1:
1
Enter the name of Vertex 2:
2
Enter the name of Vertex 3:
3
Enter the name of Vertex 4:
4
Enter the number of edges:
5
Enter edge (e.g., a -> b): 0->1
Enter the weight for edge 0 -> 1: 4
Enter edge (e.g., a -> b): 1->4
Enter the weight for edge 1 -> 4: 6
Enter edge (e.g., a -> b): 4->3
Enter the weight for edge 4 -> 3: 10
Enter edge (e.g., a -> b): 2->3
Enter the weight for edge 2 -> 3: 2
Enter edge (e.g., a -> b): 0->2
Enter the weight for edge 0 -> 2: 8
Enter the source vertex (e.g., a): 0
Enter the destination vertex (e.g., b): 4
Shortest path from 0 to 4:
0 1 4
Total cost: 10
PS C:\Users\konda\Desktop\Computer Networks>

```

**11. Develop a Program to implement Distance vector routing algorithm by obtaining routing table at each node (Take an example subnet graph with weights indicating delay between nodes).**

```
import java.util.*;
public class DistanceVectorRouting{
    static final int INF=9999;
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter number of nodes: ");
        int n=sc.nextInt();
        int[][] graph=new int[n][n];
        System.out.println("Enter adjacency matrix (use -1 for no direct link:");
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                int val=sc.nextInt();
                graph[i][j]=(val!=-1)?INF:val;
            }
        }
        int[][] cost=new int[n][n];
        int[][] nextHop=new int[n][n];
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                cost[i][j]=graph[i][j];
                if(graph[i][j]!=INF&&i!=j) nextHop[i][j]=j;
                else nextHop[i][j]=-1;
            }
            cost[i][i]=0;
            nextHop[i][i]=i;
        }
        boolean updated;
        do{
            updated=false;
            for(int i=0;i<n;i++){
                for(int j=0;j<n;j++){
                    for(int k=0;k<n;k++){
                        if(cost[i][k]!=INF&&cost[k][j]!=INF&&cost[i][k]+cost[k][j]
<cost[i][j]){
                            cost[i][j]=cost[i][k]+cost[k][j];
                            nextHop[i][j]=nextHop[i][k];
                            updated=true;
                        }
                    }
                }
            }
        }
    }
}
```



```

        }
    }
}while(updated);
System.out.println("\nRouting tables for all nodes:");
for(int node=0;node<n;node++){
    printRoutingTable(node,cost,nextHop,n);
}
System.out.print("\nEnter source node: ");
int src=sc.nextInt();
if(src<0||src>=n){
    System.out.println("Invalid source node.");
    sc.close();
    return;
}
System.out.print("Enter destination node: ");
int dest=sc.nextInt();
if(dest<0||dest>=n){
    System.out.println("Invalid destination node.");
    sc.close();
    return;
}
if(cost[src][dest]==INF){
    System.out.println("No path exists from "+src+" to "+dest);
}else{
    System.out.print("Path from node "+src+" to node "+dest+": ");
    printPath(src,dest,nextHop);
    System.out.println();
}
sc.close();
}
static void printPath(int src,int dest,int[][] nextHop){
    System.out.print(src);
    while(src!=dest){
        src=nextHop[src][dest];
        System.out.print(" -> "+src);
    }
}
static void printRoutingTable(int node,int[][] cost,int[][] nextHop,int n){
    System.out.println("\nRouting table for node "+node+":");
    System.out.println("Destination\tCost\tNext Hop");
    for(int j=0;j<n;j++){
        if(cost[node][j]==INF) System.out.println(j+"\t\tINF\t-");
        else System.out.println(j+"\t\t"+cost[node][j]+" \t"+nextHop[node][j]);
    }
}
} }

```

**Output :-**

```

PS C:\Users\konda\Desktop\computer Networks> java .\DistanceVectorRouting.java
Enter number of nodes: 5
Enter adjacency matrix (use -1 for no direct link):
0 2 3 -1 -1
3 0 -1 5 1
2 -1 0 7 -1
-1 5 7 0 2
-1 1 -1 0 2

Routing tables for all nodes:

Routing table for node 0:
Destination Cost Next Hop
0 0 0
1 2 1
2 3 2
3 3 1
4 3 1

Routing table for node 1:
Destination Cost Next Hop
0 3 0
1 0 1
2 6 0
3 1 4
4 1 4

Routing table for node 2:
Destination Cost Next Hop
0 2 0
1 4 0
2 0 2
3 5 0
4 5 0

Routing table for node 3:
Destination Cost Next Hop
0 6 4
1 3 4
2 7 2
3 0 3
4 2 4

Routing table for node 4:
Destination Cost Next Hop
0 4 1
1 1 1
2 7 1
3 0 3
4 0 4

Enter source node: 0
Enter destination node: 4
Path from node 0 to node 4: 0 -> 1 -> 4
PS C:\Users\konda\Desktop\computer Networks>

```

## 12. Wireshark

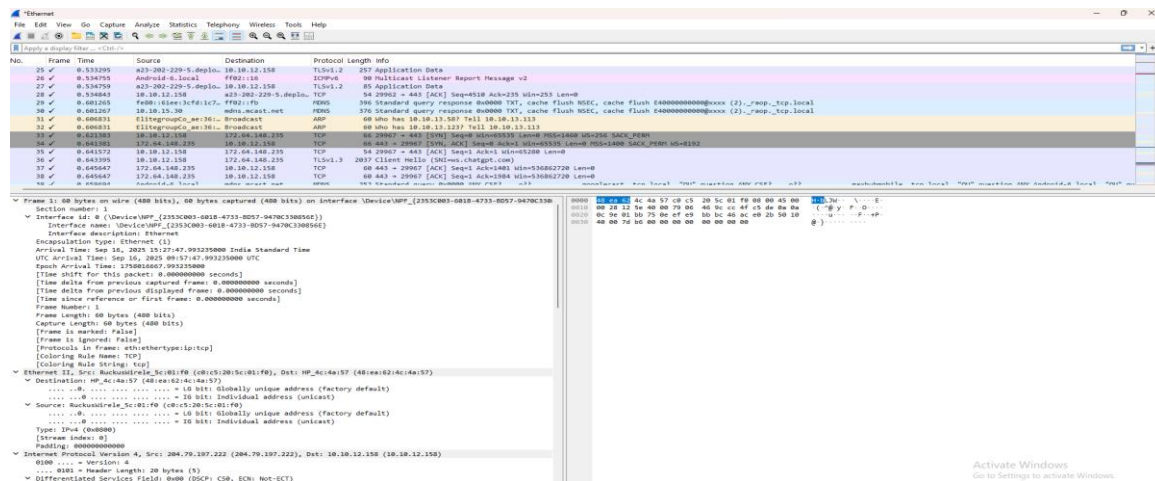
- i. Packet Capture Using Wire shark
- ii. Starting Wire shark
- iii. Viewing Captured Traffic
- iv. Analysis and Statistics & Filters.

### i). Packet Capture:

### ii). Starting Wire shark

- Launch Wireshark from the Start menu (Windows/Linux) or Applications folder (Mac).
- Confirm it is running and the interface selection screen is visible.

: Wireshark startup screen with interface list.



The screenshot shows the Wireshark I/O Graphs - Ethernet window. The main graph displays packet rate over time. The Y-axis is labeled 'Packets/sec' and ranges from 0 to 300. The X-axis is labeled 'Time (s)' and ranges from 0 to 200. The graph shows three data series: 'All Packets' (blue line), 'TCP Errors' (red line), and 'Filtered packets' (yellow line). The 'All Packets' series shows a significant spike to approximately 250 packets/sec around 100 seconds. The 'TCP Errors' series shows a small spike to approximately 20 packets/sec around 100 seconds. The 'Filtered packets' series shows a small spike to approximately 10 packets/sec around 100 seconds. The graph is titled 'Wireshark I/O Graphs - Ethernet'. The legend in the top right corner indicates '1 sec intervals' and lists the three data series. The bottom of the window shows the 'Packet List' pane with a table of captured packets, including columns for 'Enabled', 'Graph Name', 'Display Filter', 'Color', 'Line', 'Style', 'Packets', 'Y Axis', 'Y Field', 'SMA Period', and 'Y Axis Factor'. The table contains three rows: 'All Packets', 'TCP Errors', and 'Filtered packets'. The 'All Packets' row is selected. The bottom status bar shows 'Mouse: drag to zoom', 'Interval: 1 sec', and 'Log scale'.

Ethernet

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter: <Ctrl>

No	Frame	Time	Source	Destination	Protocol	Length	Info
55	✓	0.697462	172.64.148.235	10.10.12.158	TCP	68	643 → 29967 [FIN, ACK] Seq=8551 Ack=7490 Win=147456 Len=0
56	✓	0.698248	10.10.12.158	172.64.148.235	TCP	54	29967 → 443 [ACK] Seq=7490 Ack=6852 Win=65280 Len=0
57	✓	0.708158	10.10.12.158	172.64.148.235	TCP	54	29967 → 443 [FIN, ACK] Seq=7490 Ack=6852 Win=65280 Len=0
58	✓	0.710983	172.64.148.235	10.10.12.158	TCP	68	443 → 29967 [ACK] Seq=8551 Ack=7491 Win=147456 Len=0
59	✓	0.710983	172.64.148.235	10.10.12.158	TCP	68	443 → 29967 [ACK] Seq=8551 Ack=7491 Win=147456 Len=0
60	✓	0.741867	EF0000-182.local	ndns.mcast.net	PDNS	305	Standard query response 0x0000 PTR EF0000-182._dovsc._tcp.local SRV 0 0 7680 EF0000-182.local TXT

▼ Frame 58: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF\_{2353C083-6818-4733-8057-9478C330856E}

Section number: 1

Interface id: 0 (\Device\NPF\_{2353C083-6818-4733-8057-9478C330856E})

Interface name: \Device\NPF\_{2353C083-6818-4733-8057-9478C330856E}

Interface description: Ethernet

Encapsulation type: Ethernet (1)

Arrival Time: Sep 16, 2025 15:07:48.693527000 India Standard Time

UTC Arrival Time: Sep 16, 2025 09:57:48.693527000 UTC

Epoch Arrival Time: 1758016668.693527000

[Time shift for this packet: 0.000000000 seconds]

[Time delta from previous captured frame: 0.000134000 seconds]

[Time delta from previous displayed frame: 0.000134000 seconds]

[Time since reference or first frame: 0.708292000 seconds]

Frame Number: 58

Frame Length: 54 bytes (432 bits)

Capture Length: 54 bytes (432 bits)

[Frame is marked: False]

[Frame is ignored: False]

[Protocols in frame: ethertypeip:tcp]

[Coloring Rule Name: TCP RST]

[Coloring Rule String: tcp.flags.reset eq 1]

▼ Ethernet II, Src: HP\_Ac4a:57 (48:ea:62:4c:4a:57), Dst: RuckusWirele\_Sc81f0 (c8:c5:20:5c:01:f0)

Destination: RuckusWirele\_Sc81f0 (c8:c5:20:5c:01:f0)

.....0..... = 10 bit: Globally unique address (factory default)

.....0..... = 10 bit: Individual address (unicast)

Source: HP\_Ac4a:57 (48:ea:62:4c:4a:57)

.....0..... = 10 bit: Globally unique address (factory default)

.....0..... = 10 bit: Individual address (unicast)

Type: IPv4 (0x0000)

[Stream Index: 0]

▼ Internet Protocol Version 4, Src: 10.10.12.158 (10.10.12.158), Dst: 172.64.148.235 (172.64.148.235)

0100 .... = Version: 4

....0011 = Header Length: 20 bytes (5)

▼ Differentiated Services Field: 0x00 (DSCP: CS, ECN: Not-ECT)

0000 00.. = Differentiated Services Codepoint: Default (0)

.....00 = Explicit Congestion Notification: Not ECT-Capable Transport (0)

Total Length: 60

Identification: 0x0268 (25304)

▼ 010. .... = Flags: 0x2, Don't fragment

0..... = Reserved bit: Not set

..1.... = Don't fragment: Set

..0.... = More fragments: Not set

...0 0000 0000 0000 = Fragment Offset: 0

Time to Live: 128

0000 c8 c5 20 5c 01 f0 48 ea 62 4c 4a 57 06 00 45 00 ... \.H. bL2M: E-

0010 00 28 62 65 00 00 00 05 00 00 0a 0c 9c 9c 40 ... (0)@ .....@

0020 94 e0 75 0f 01 b0 51 e0 06 0c 95 58 09 76 50 14 ... -@ Q .....@

0030 00 00 57 ee 00 00 ... -M---

Activate Windows  
Go to Settings to activate Windows.

Ethernet

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter: <Ctrl>

No	Frame	Time	Source	Destination	Protocol	Length	Info
184	✓	0.871776	b-resolvers.level3...	DESKTOP-THPV9M1.LOC...	DNS	154	Standard query response 0x000d No such name PTR 80-12.10.10.in-addr.arpa SOA localhost OPT
185	✓	0.871776	b-resolvers.level3...	DESKTOP-THPV9M1.LOC...	DNS	152	Standard query response 0x000e No such name PTR 251.0.0.254.in-addr.arpa SOA svs.dns.icann.org OPT
186	✓	0.871776	b-resolvers.level3...	DESKTOP-THPV9M1.LOC...	DNS	153	Standard query response 0x000f No such name PTR 1.15.10.10.in-addr.arpa SOA localhost OPT
187	✓	0.871776	b-resolvers.level3...	DESKTOP-THPV9M1.LOC...	DNS	155	Standard query response 0x0010 No such name PTR 250.250.250.250.in-addr.arpa SOA vni-devs.icann.org OPT
188	✓	0.871776	b-resolvers.level3...	DESKTOP-THPV9M1.LOC...	DNS	154	Standard query response 0x0011 No such name PTR 35.15.10.10.in-addr.arpa SOA localhost OPT
189	✓	0.871776	b-resolvers.level3...	DESKTOP-THPV9M1.LOC...	DNS	158	Standard query response 0x0012 PTR 255.255.255.255.in-addr.arpa SOA localhost OPT
190	✓	0.871776	b-resolvers.level3...	DESKTOP-THPV9M1.LOC...	DNS	153	Standard query response 0x0013 PTR 0.0.0.0.in-addr.arpa SOA localhost OPT

▼ Frame 6: 530 bytes on wire (4240 bits), 530 bytes captured (4240 bits) on interface \Device\NPF\_{2353C083-6818-4733-8057-9478C330856E}

Section number: 1

Interface id: 0 (\Device\NPF\_{2353C083-6818-4733-8057-9478C330856E})

Encapsulation type: Ethernet (1)

Arrival Time: Aug 26, 2025 14:30:29.277531000 India Standard Time

UTC Arrival Time: Aug 26, 2025 09:08:29.277531000 UTC

Epoch Arrival Time: 1756180389.277531000

[Time shift for this packet: 0.000000000 seconds]

[Time delta from previous captured frame: 0.017040000 seconds]

[Time delta from previous displayed frame: 0.000000000 seconds]

[Time since reference or first frame: 0.865690000 seconds]

Frame Number: 6

Frame Length: 530 bytes (4240 bits)

Capture Length: 530 bytes (4240 bits)

[Frame is marked: False]

[Frame is ignored: False]

[Protocols in frame: ethertypeip:ipudp:data]

[Coloring Rule Name: UDP]

[Coloring Rule String: udp]

▼ Ethernet II, Src: KichTeching\_Sc9c158 (e4:65:64:5e:9c:158), Dst: IP4mcast\_7a1fa1fa (01:00:5e:7a:1f:a1fa)

Destination: IP4mcast\_7a1fa1fa (01:00:5e:7a:1f:a1fa)

Source: KichTeching\_Sc9c158 (e4:65:64:5e:9c:158)

Type: IP4v (0x0000)

[Stream Index: 5]

▼ Internet Protocol Version 4, Src: 10.10.15.1 (10.10.15.1), Dst: 239.250.250.250 (239.250.250.250)

0100 .... = Version: 4

....0101 = Header Length: 20 bytes (5)

▼ Differentiated Services Field: 0x00 (DSCP: CS, ECN: Not-ECT)

Total Length: 516

Identification: 0x07ab (20683)

▼ 010. .... = Flags: 0x2, Don't fragment

...0 0000 0000 0000 = Fragment Offset: 0

Time to Live: 1

Protocol: UDP (17)

Header Checksum: 0x745d [validation disabled]

[Header checksum status: Unverified]

Source Address: 10.10.15.1 (10.10.15.1)

Destination Address: 239.250.250.250 (239.250.250.250)

[Stream Index: 2]

User Datagram Protocol, Src Port: 40592, Dst Port: 37020

Data (488 bytes)

0000 01 00 5e 7a 1f a1 fa e4 65 64 5e 9c 15 80 00 00 ... .-e d'X: E-

0010 02 00 7f ab 40 00 01 11 f4 3d 0a 0a 0f 01 ef fa ... @ . . . . .

0020 fa fa 9e 00 00 0c 01 f0 80 c3 3f 78 6d 6c 28 ... .f.ml

0030 76 05 72 73 69 6f 6a 3d 22 31 2a 38 22 28 65 6e ... 6 . . . . .

0040 63 6f 64 69 6e 67 3d 22 75 74 66 2d 38 22 3f 3e ... codings= utf-8">

0050 00 0a 3c 48 65 6c 6c 6f 3e 0d 0a 3c 54 79 78 05 ... <html> < type

0060 73 3e 68 65 6c 6c 6f 3c 2f 54 79 78 65 73 3e 6d ... <hollow /types>

0070 0a 3c 44 65 76 69 63 65 54 79 78 65 3e 33 33 33 ... <device Type>233

0080 33 33 3c 2f 44 65 76 69 63 65 54 79 78 65 3e 6d ... 33/<Device type>

0090 0a 3c 44 65 76 69 63 65 44 65 73 63 72 69 78 74 ... <Device Descript

00a0 09 6f 6a 3e 78 78 78 78 3c 2f 44 65 76 69 63 65 ... Description Loo

00b0 44 65 73 63 72 69 78 74 09 6f 6e 3e 0d 6a 3c 44 ... evic&lt; 2cfe1a2

00c0 65 76 69 63 65 53 4e 3c 32 63 3a 66 65 3a 05 32 ... <saM18 B/Device

00d0 3a 35 61 3a 33 3a 33 38 3c 2f 44 65 76 69 63 65 ... <id>: <@ ttpurty>

00e0 65 53 4e 3a 0d 0a 3c 48 74 74 78 58 6f 72 74 3e ... 0c/nettp ort> <@

00f0 30 3c 2f 48 74 74 78 58 6f 72 74 3e 0d 0a 3c 44 ... <@urty> 53c/0a0

0100 57 58 6f 72 74 3e 38 35 3c 3c 2f 44 65 76 69 6f ... <@a 3c 2f 4d 41 43

0110 72 74 3e 0d 0a 3c 4d 41 43 3e 32 63 3a 66 65 3a ... <@a 3c 2f 4d 41 43

0120 65 52 3a 35 63 3a 33 3e 33 38 3c 2f 4d 41 43 ... <@a 3c 2f 4d 41 43

0130 3a 0d 0a 3c 49 58 76 3a 41 64 64 72 65 73 73 3e ... <@a 3c 2f 4d 41 43

0140 35 38 2a 31 38 2a 31 35 2a 31 3c 2f 49 50 76 34 ... 10.10.15.1/<IPv4

0150 41 64 64 72 65 73 73 3e 0d 0a 3c 49 50 76 34 53 ... Address> <@IPv4

0160 75 62 6e 65 74 4d 61 73 6b 3e 2f 49 50 76 34 ... Subnetthe sk> <@

0170 53 75 62 6e 65 74 4d 61 73 6b 3e 2f 49 50 76 34 ... vAdateme yy>/<IPv4

0180 76 34 47 61 74 65 77 61 79 3e 3c 2f 49 50 76 34 ... Gateway> <@aufr

0190 47 61 74 65 77 61 79 3e 0d 0a 3c 41 75 78 58 72 ... <@uipr> <@uipr

01a0 69 76 69 6c 65 67 65 3a 0d 0a 3c 4c 69 73 74 58 ... <@uipr> <@uipr

01b0 6f 6a 65 63 74 3e 0d 0a 3c 49 78 4c 69 73 74 58 ... <@uipr> <@uipr

01c0 74 3a 31 38 2a 31 38 2a 31 35 2a 31 3c 31 39 32 ... <@uipr> <@uipr

01d0 2a 31 38 2a 31 38 2a 31 3c 3c 2f 49 50 76 34 ... <@uipr> <@uipr

01e0 73 74 76 34 3e 0d 0a 3c 2f 48 65 6c 6c 6f 3e 6d ... <@uipr> <@uipr

01f0 00 00 ... ..

Activate Windows

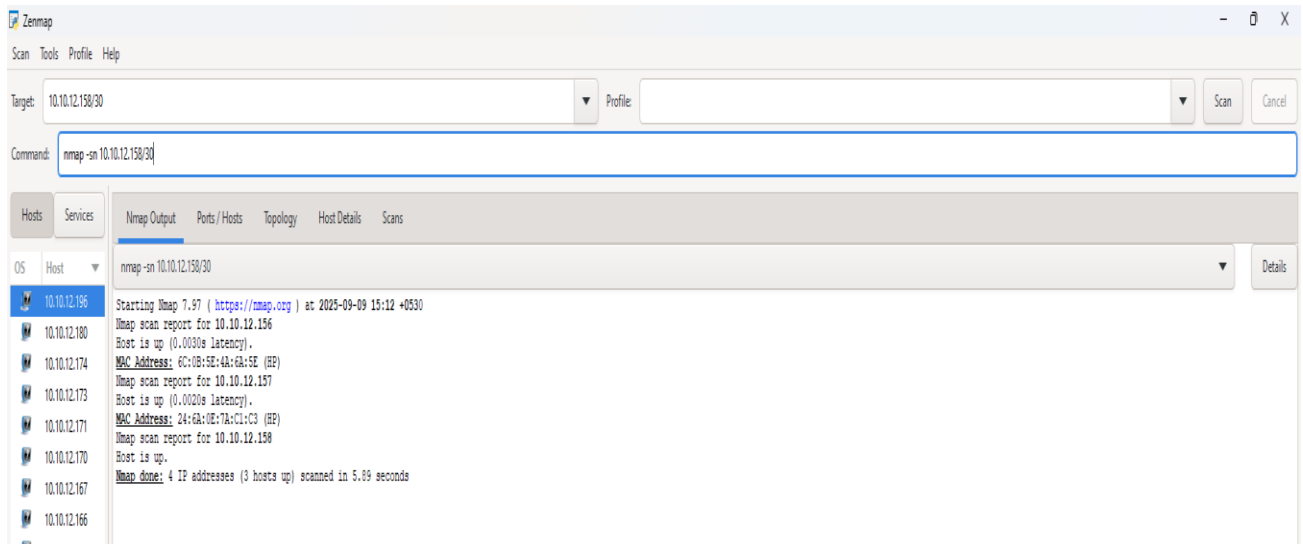
[illegible]



### 13. Discover active hosts, open ports and running services in a network using Nmap

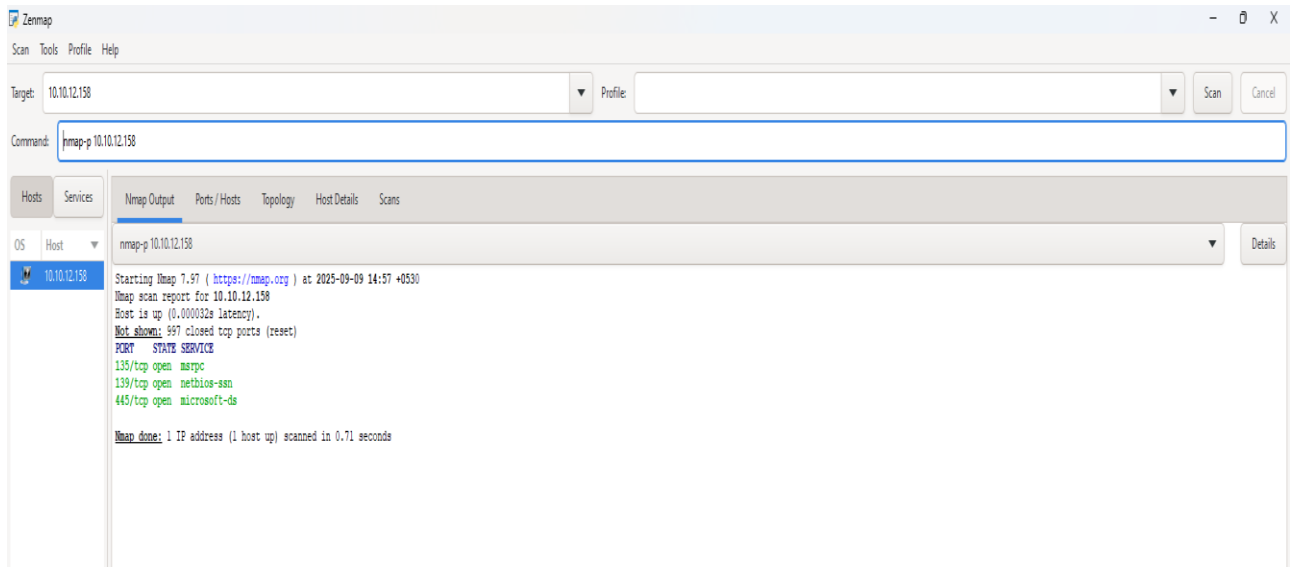
#### i) Discover active hosts using Nmap:

**Command** :- `nmap -sn <IP Address>`

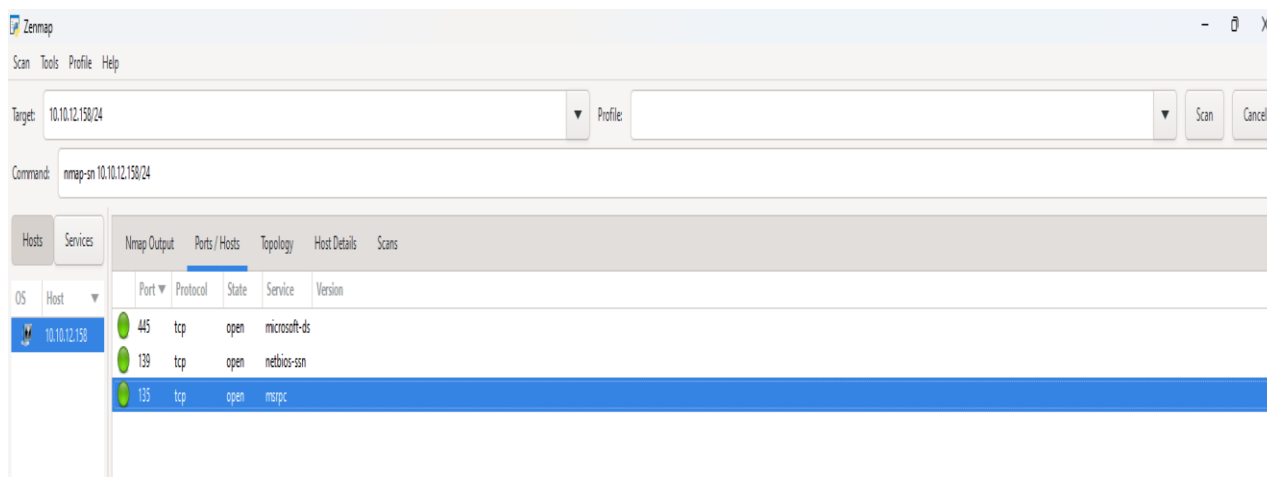


#### ii) To find no.of open ports in the system

**Command** :- `nmap -p 1-65535 <IP Address>`

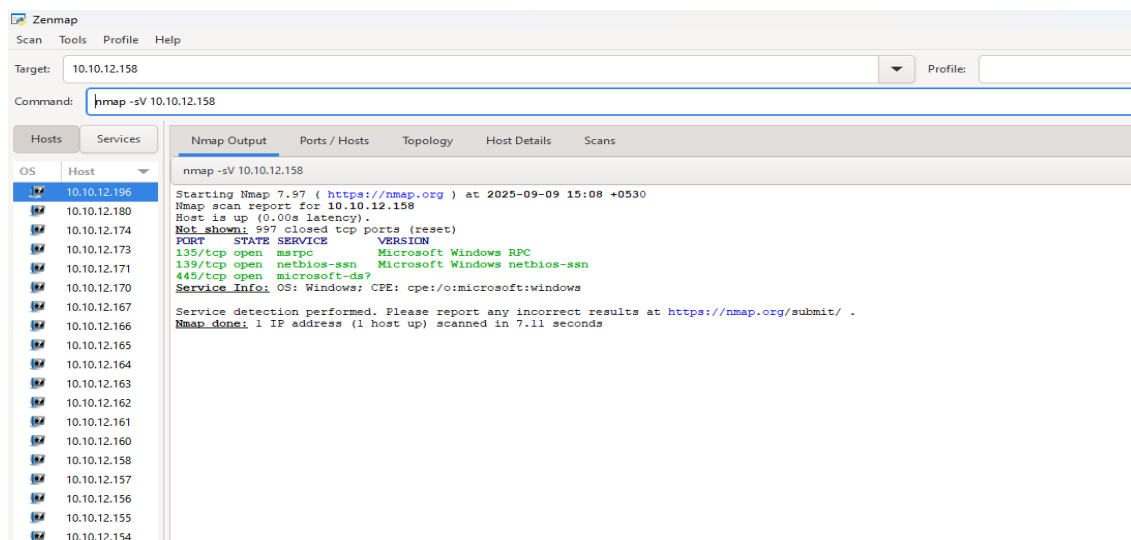






iii) To find Running services and versions using Nmap:

**Command** :- `nmap -sV <IP Address>`



## 14. Find Operating System in a host using Nmap

### Command to Use:

*nmap -O 10.10.12.163*

