GATE INSTITUTE OF TECHNOLOGY & MANAGEMENT SCIENCES

(Approved by AICTE , Affiliated to Sri Venkateswara University , TIRUPATI)

ICET CODE : GTMT



Master of Computer Applications

SEMESTER - 3rd

Hall Ticket No: 12253143

Name of the Student:

Subject: Computer Networks

Year: IInd Semester: 3rd Batch 2025 - 2027

GATE EDUCATIONAL INSTITUTIONS

DEPARTMENT OF COMPUTERS

YEAR II nd SEMESTER 3 rd

Reg No. 122531043

CERTIFICATE

Certified that this is the bonafide record of practical work done in the laboratory by the candidate _____ *during the academic year* 2026-2027 *subject* Computer Networks .

No. of Practical's Conducted 12 ***No. of Practical's Attended*** 12

LECTURER **HEAD OF THE DEPARTMENT**

Submitted for the Practical Examination held on _____

Examiners

1.

2.

INDEX

S.No.	Date	Name of the Experiment	Page No.	Marks Awarded	Remarks
1.	10/9/26	Creating a client - server application in java.	1-5		
2.	17/9/26	One Time pad using Java.	6-9		
3.	28/9/26	My PC IP Addren using Java.	10-12		
4.	6/10/26	Leaky basket algorithm to smooth out bursty traffic and control network congestion.	13-17		
5.	24/10/26	Stop-and-wait sliding window protocol for flow control.	18-22		
6.	4/11/26	Two-Dimensional redundancy parity.	23-27		
7.(a)	11/11/26	Bit stuffeing and bit destrutting. Using Java	28-32		

INDEX

S.No.	Date	Name of the Experiment	Page No.	Marks Awarded	Remarks
7.(b)	17/11/26	character stuffing using Java.	33-37		
8.	24/11/26	CRC polynomial is sender side and Receiver side.	38-42		
9.	28/11/26	Data Link layer framing method. (character string)	43-46		
10.	8/12/26	parity check bit (Even & odd)	47-53		
11.	22/12/26	RSA Algorithm for string Encryption.	54-60		
12.	30/12/26	playfair Cipher Substitution technique.	61-67		

Date : 10/9/26	GATE Educational Institutions	Page No. : 1
	Creating a client - server application Using Java.	Practical No. : 01

1. Aim :-

creating a client - server application in java involves writing two seperate program :
one of the server and one for the client.

Description :-

Server program

The server program listens on a specified port and waits for the client connections. once a client connects, it reads a message from the client and spends a response.

client program

the client program connects to the server on the specified port, sends a message, and waits for the server's response.

procedure :-

steps to Run programs

compile the server and client programs

Run the server program

Run the client program (in a seperate terminal or command prompt) :

when you run these programs, the server will start and stop for a client connection. the client will connect to the server, send a menage, and receive a response. Both messages will be printed to their respective consoles.

Date :

GATE Educational Institutions

Page No. : 8

Practical No. :

//1. Server.java

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

public class Server {
   public static void main(String[] args) {
      final int PORT = 1234;

      try {
         // Create server socket
         ServerSocket serverSocket = new ServerSocket(PORT);
         System.out.println("Server waiting for client on port " + PORT);

         // Accept client connection
         Socket socket = serverSocket.accept();
         System.out.println("Client connected");

         // Create input stream to receive messages from client
         BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));

         // Create output stream to send messages to client
         PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

         // Read messages from client and send back a response
         String message;
         while ((message = in.readLine()) != null) {
            System.out.println("Client: " + message);
            out.println("Server received: " + message);
         }

         // Close streams and socket
         in.close();
         out.close();
         socket.close();
         serverSocket.close();
      } catch (IOException e) {
         e.printStackTrace();
      }
   }
}
```

Date :

GATE Educational Institutions

Page No. : 3

Practical No. :

//Client.java

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

public class Client {
   public static void main(String[] args) {
      final String SERVER_ADDRESS = "localhost";
      final int PORT = 1234;

      try {
         // Create socket to connect to server
         Socket socket = new Socket(SERVER_ADDRESS, PORT);

         // Create input stream to receive messages from server
         BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));

         // Create output stream to send messages to server
         PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

         // Create input stream to read user input
         BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in));

         String message;
         while (true) {
            System.out.print("Enter message to send to server: ");
            message = userInput.readLine();
            out.println(message);

            // Exit loop if user types "exit"
            if (message.equalsIgnoreCase("exit")) {
               break;
            }

            // Read response from server
            String response = in.readLine();
            System.out.println("Server: " + response);
         }

         // Close streams and socket
         in.close();
         out.close();
```

```
            userInput.close();
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Output:

Server Site:

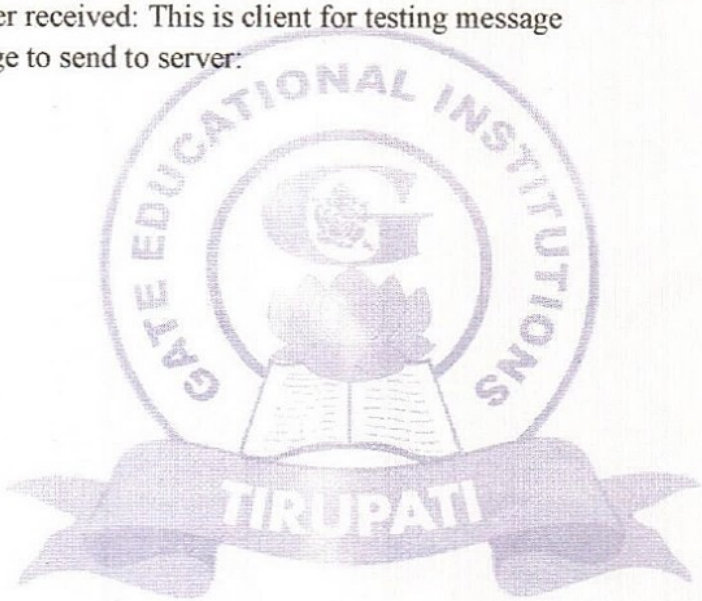
Server waiting for client on port 1234

Client Site:

Enter message to send to server**: This is client for testing message**

Server: Server received: This is client for testing message

Enter message to send to server:



conclusion :-

This is a simple example of a client-server application. for more complex applications, you might want to handle multiple clients, use more sophisticated communication protocols or employ threading to handle concurrent connections.

GATE EDUCATIONAL INSTITUTIONS
TIRUPATI

One Time pad using Java | Practical No. : 02

2. Aim :-

Implement the data link layer framing methods such as One Time pad using Java program.

Description :-

one-time pads are practical in situations where two parties in a secure environment must be able to depart from one another and communication from two seperate secure environments with perfect securey. The one-time pad can be used in super Encryption. the primary use of padding with classical ciphers is to prevent the cryptanalyst from using that predictability to find known plaintext that aids in breaking the encryption. Random length padding also prevents from knowing the exact length of the plaintext message.

plaintext : THIS IS SECRET
OTP-Key : XVHE UW NOPGDZ

---

ciphertext : QCPW CO FSRXHS
In groups : QCPWC OFSRXHS

Date :

GATE Educational Institutions

Page No. : ८

Practical No. :

// 2. Program to write and execute a program OnetimePad using Java.

```
import java.io.*;
import java.math.*;
import java.lang.*;
import java.util.*;
class Onetimepad
{
public static void main(String arg[])throws IOException
{
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
Random rg=new Random();
String temp=" ";
System.out.print("enter plain text:");
temp=br.readLine();
temp=temp.toUpperCase();
int length=temp.length();
char plaintext[]=new char[length];
char key[]=new char[length];
char ciphertext[]=new char[length];
char deciphertext[]=new char[length];
for(int i=0;i<length;i++)
{
plaintext[i]=temp.charAt(i);
key[i]=((char)(65+rg.nextInt(26)));
ciphertext[i]=(char)((int)plaintext[i]^(int)key[i]);
deciphertext[i]=((char)((int)ciphertext[i]^(int)key[i]));
}
System.out.print("entered plaintext:");
for(int i=0;i<length;i++)
System.out.print(" "+plaintext[i]);
System.out.println(" ");
System.out.print("enter key:");
for(int i=0;i<length;i++)
System.out.print(" "+key[i]);
System.out.println(" ");
System.out.print("ciphertext:");
for(int i=0;i<length;i++)
System.out.print("  "+ciphertext[i]);
System.out.println(" ");
System.out.print("deciphertext:");
for(int i=0;i<length;i++)
System.out.print(" "+deciphertext[i]);  }}
```

Date :

GATE Educational Institutions

Page No. : 8

Practical No. :

Output:

```
C:\dccn>javac Onetimepad.java

C:\dccn>java Onetimepad

enter plain text:kmmips

entered plaintext: K M M I P S

enter key: N L D G W C

ciphertext: ♣ ☺ ♫ ►

deciphertext: K M M I P S
```



conclusion :-

finally the following One Time pad program executed succesfully using Java and output generated without any errors.

GATE EDUCATIONAL INSTITUTIONS
TIRUPATI

Date : 28/9/26	GATE Educational Institutions	Page No. : 10
	My PC IP Address using Java	Practical No. : 03

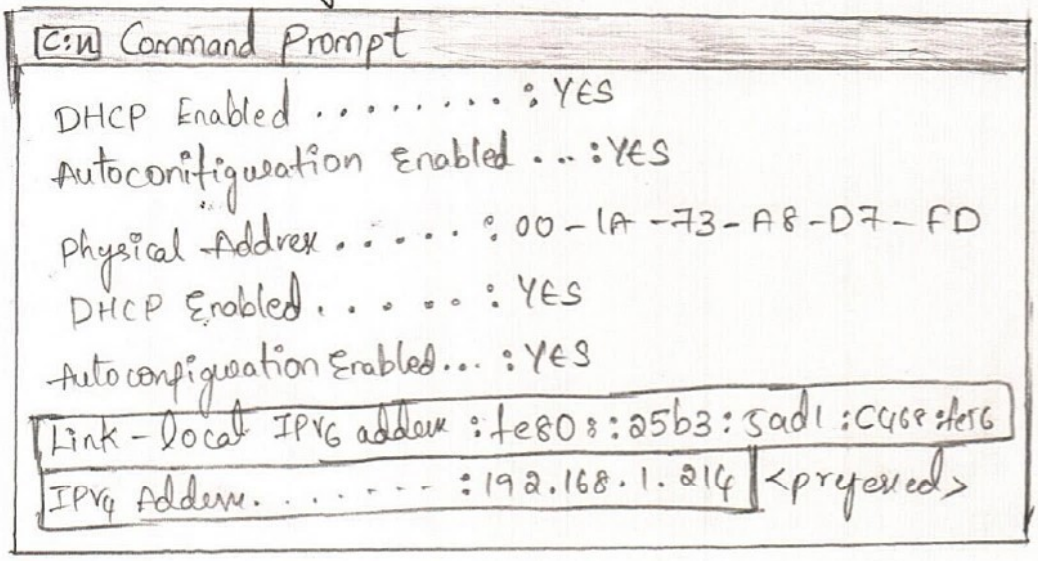
3. Aim :-

Implement the data link layer framing methods such as find My pc IP Address using Java program.

Description :-

An IP (Internet protocol) address is an identifier assigned to each computer and another device (eg:- router, mobile, etc) connected to a TCP/IP network that is used to locate and identify the node in communication with other nods on the network. IP Addresses are usually written and displayed in human-readable notation such as 192.168.1.35 in IPV4 (32-bit IP address).

An IP address serves two principal functions: host (or) network interface identification and local addressing. Its role has been characterized as follows: "A name indicates what we seek. An address indicates where it is. A route indicates how to get there".



Date :

GATE Educational Institutions

Page No. : 11

Practical No. :

//3.Program to write and Execute a program Find My PC IP Address using Java.

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

public class ipadd {
    public static void main(String[] args) {
        try {
            // Get the local host information
            InetAddress localHost = InetAddress.getLocalHost();

            // Get the hostname
            String hostname = localHost.getHostName();

            // Get the IP address
            String ipAddress = localHost.getHostAddress();

            // Print the hostname and IP address
            System.out.println("Hostname: " + hostname);
            System.out.println("IP Address: " + ipAddress);
        } catch (UnknownHostException e) {
            System.err.println("Unable to determine the hostname and IP address.");
            e.printStackTrace();
        }
    }
}
```

Output:

Hostname: DESKTOP-61JE93H
IP Address: 192.168.1.4

conclusion :-

finally the following find my pc IP address program executed succesfully using Java and output generated without any error's.

GATE EDUCATIONAL INSTITUTIONS
TIRUPATI

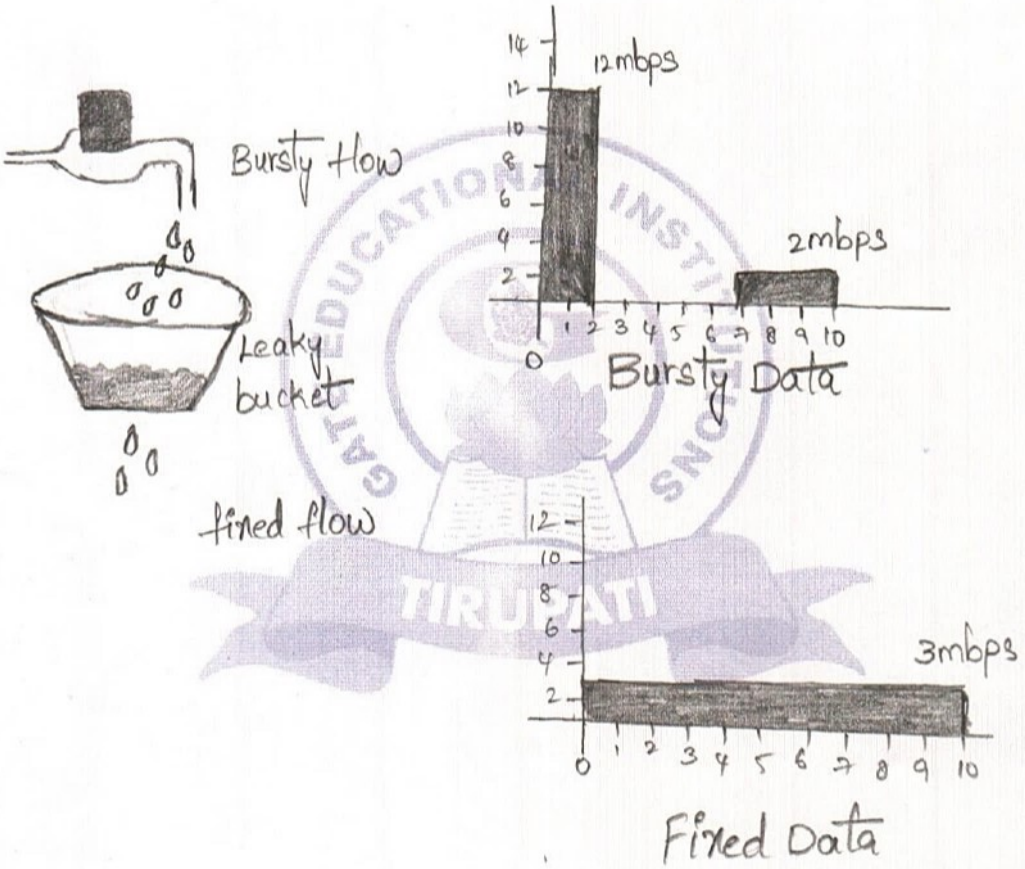
Leaky bucket algorithm to smooth out bursty traffic and control network congestion.

4. Aim :-
To write a Java program to implement the leaky bucket algorithm to smooth out bursty traffic and control network congestion.

Description :-
The leaky bucket algorithm shapes bursty traffic into a steady, predictable output stream. The bucket represents a local buffer memory. Incoming data packets can arrive at random, erratic intervals (bursts). If the data flow exceeds the maximum buffer storage capacity, excess packets drop immediately. The bucket leaks data packets out at a fixed, constant transmission rate.

Algorithm :-
* Define the bucket storage capacity (bucket_size) and the static transmission rate (output_rate)
* Input the number of execution steps or time intervals to simulate.
* At each time interval, generate a random number of incoming packets (bursty input)
* check if the incoming packet size fits into the remaining storage of the bucket.
* If it fits, accept the packets into the bucket. Update the total storage counter.
* If it exceeds capacity, accept packets up to the maximum limit and drops the remaining excess traffic.

* Deduct the fixed output-rate from the bucket data to simulate transmission.
* If the bucket holds fewer packets than the output rate, empty the bucket completely.
* Repeat the cycle for all remaining time slices.



Date :

GATE Educational Institutions

Page No. : 15

Practical No. :

//4. A Java program to implement the Leaky Bucket algorithm to smooth out bursty traffic and control network congestion.

Source Code:

```
import java.util.Random;
import java.util.Scanner;

public class LeakyBucket {
   public static void main(String[] args) throws InterruptedException {
      Scanner sc = new Scanner(scScannerSystem.in);
      Random rand = new Random();

      System.out.print("Enter Bucket Capacity (BufferSize): ");
      int bucketSize = sc.nextInt();

      System.out.print("Enter Constant Output Leak Rate: ");
      int outputRate = sc.nextInt();

      System.out.print("Enter total time steps to simulate: ");
      int timeSteps = sc.nextInt();

      int currentBufferContent = 0;

      System.out.println("\nTime\tIncoming\tAccepted\tDropped\t\tBuffer Status");
      System.out.println("-------------------------------------------------------------------");

      for (int t = 1; t <= timeSteps; t++) {
         // Generate erratic burst traffic between 0 and twice the output rate
         int incomingPacket = rand.nextInt(outputRate * 2 + 5);
         int acceptedPacket = 0;
         int droppedPacket = 0;

         if (incomingPacket + currentBufferContent <= bucketSize) {
            acceptedPacket = incomingPacket;
            currentBufferContent += incomingPacket;
         } else {
            acceptedPacket = bucketSize - currentBufferContent;
            droppedPacket = incomingPacket - acceptedPacket;
            currentBufferContent = bucketSize;
         }
```

```
System.out.print(t + "\t" + incomingPacket + "\t\t" + acceptedPacket + "\t\t" +
droppedPacket + "\t\t" + currentBufferContent + " (Before Leak)");

            // Apply steady leakage flow
            if (currentBufferContent >= outputRate) {
                currentBufferContent -= outputRate;
            } else {
                currentBufferContent = 0; // Bucket completely drains out
            }

            System.out.println(" -> " + currentBufferContent + " (After Leak)");
            Thread.sleep(600); // Small delay to observe terminal print outs
        }
        sc.close();
    }
}
```

OUT PUT:

```
Enter Bucket Capacity (BufferSize): 15
Enter Constant Output Leak Rate: 4
Enter total time steps to simulate: 5

Time  Incoming     Accepted     Dropped      Buffer Status
---------------------------------------------------------------
1     3            3            0            3 (Before Leak) -> 0 (After
Leak)
2     11           11           0            11 (Before Leak) -> 7 (After
Leak)
3     12           8            4            15 (Before Leak) -> 11 (After
Leak)
4     2            2            0            13 (Before Leak) -> 9 (After
Leak)
5     9            6            3            15 (Before Leak) -> 11 (After
Leak)
```

conclusion :-

The Java program was succesfully implemented to simulate Leaky Bucket Algorithm for controlling network cogestion and smoothing bursty traffic. The program accepts income packets into the bucket according to its capacity and transmits them at a fixed rate. When the bucket becomes full, excess packets are discarded. Thus, the algorithm helps maintain a steady and controlled data flow in a network.

GATE EDUCATIONAL INSTITUTIONS TIRUPATI

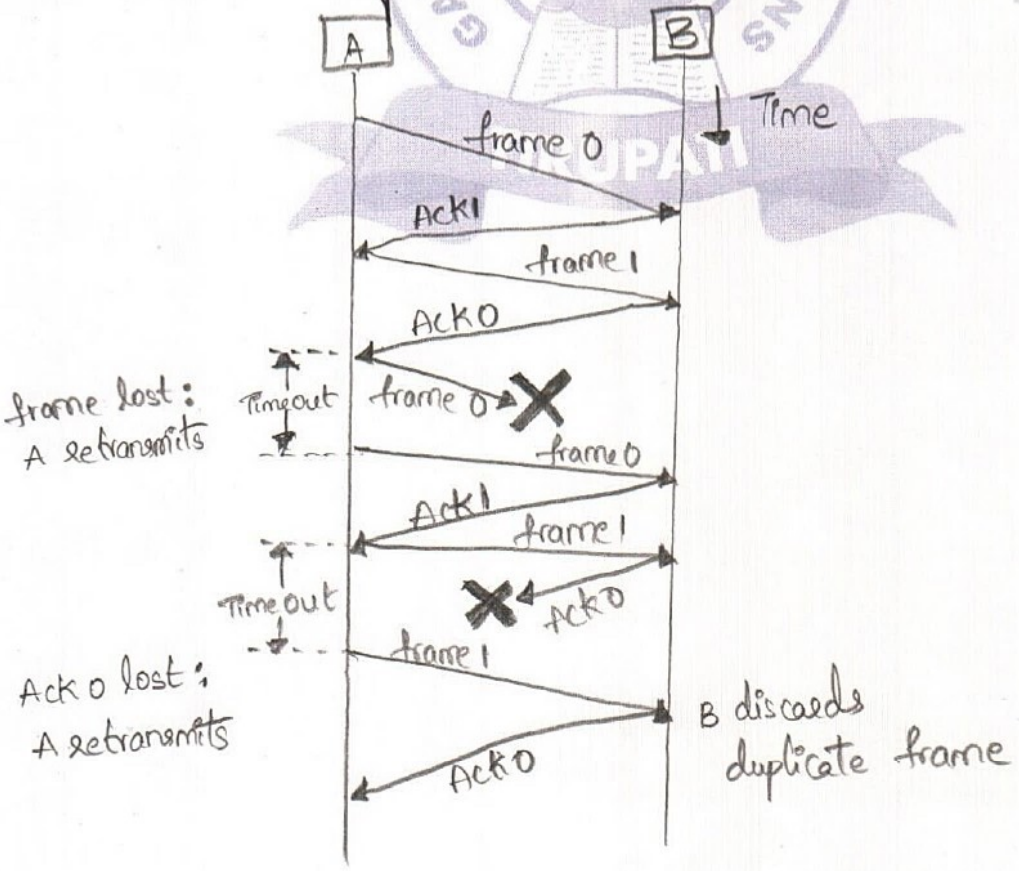
stop - and - wait sliding window protocol for flow control. | Practical No. : 05

5. Aim :-

To write a Java program to simulate the stop - and - wait sliding window protocol for flow control.

Description :-

stop - and - wait is a structural layer flow control protocol. It ensures a high - speed sender does not overwhelm a slower destination node. The sender outputs a single numered data limit (frame) and transitions into an idle state. It says frozen until a positive acknowledgement packet (Ack) arrives. If a packet gets dropped by bad routing or ambient noise, an internal clock trigger fires a timeout, forcing a retransmission.



Date :

GATE Educational Institutions

Page No. : 19

Practical No. :

//5. A Java program to simulate the Stop-and-Wait Sliding Window protocol for flow control.

Source Code:

```
import java.io.*;
import java.util.Random;
public class StopAndWait {
   public static void main(String[] args) throws Exception {
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      Random rand = new Random();
      System.out.print("Enter the total number of frames to transmit: ");
      int totalFrames = Integer.parseInt(br.readLine());
      int currentFrame = 1;
          while (currentFrame <= totalFrames) {
         System.out.println("\n[Sender] Sending Frame " + currentFrame + "...");
         Thread.sleep(800);
          int outcome = rand.nextInt(3); // Outcomes: 0 = Clear, 1 = Frame Drop, 2 = ACK
Drop
           if (outcome == 1) {
             System.out.println("[Network] Warning: Frame " + currentFrame + " dropped!");
             System.out.println("[Sender] Timeout! Retransmitting frame...");
          }
         else if (outcome == 2) {
             System.out.println("[Receiver] Frame " + currentFrame + " processed.");
             System.out.println("[Receiver] Dispatched ACK " + (currentFrame + 1));
             System.out.println("[Network] Warning: ACK " + (currentFrame + 1) + "
dropped!");
             System.out.println("[Sender] Timeout! Retransmitting frame...");
          }
```

```
        else {
            System.out.println("[Receiver] Frame " + currentFrame + " processed");
            System.out.println("[Receiver] Dispatched ACK " + (currentFrame + 1));
            System.out.println("[Sender] ACK " + (currentFrame + 1) + " verified.");
            currentFrame++;
        }
    }
    System.out.println("\nAll transmissions completed!");
  }
}
```



Date :

GATE Educational Institutions

Page No. : 21

Practical No. :

Output:

Enter the total number of frames to transmit: 2

[Sender] Sending Frame 1...

[Receiver] Frame 1 processed.

[Receiver] Dispatched ACK 2

[Sender] ACK 2 verified.

[Sender] Sending Frame 2...

[Network] Warning: Frame 2 dropped!

[Sender] Timeout! Retransmitting frame...

[Sender] Sending Frame 2...

[Receiver] Frame 2 processed.

[Receiver] Dispatched ACK 3

[Sender] ACK 3 verified.

All transmissions completed!

GATE EDUCATIONAL INSTITUTIONS TIRUPATI

Conclusion :—

The Java program was succesfully implemented to simulate the stop-and-wait sliding window protocol for flow control. The protocol ensures that the sender transmits one frame at a time and waits for an acknowledgement from the receiver before sending the next frame. If an acknowledgement is not received within the specified time, the frame is retransmitted. Thus, the protocol provides reliable data transmission and prevents the receiver from being overwhelmed.

GATE EDUCATIONAL INSTITUTIONS TIRUPATI

Two-Dimensional redundancy parity. | Practical No.: 06

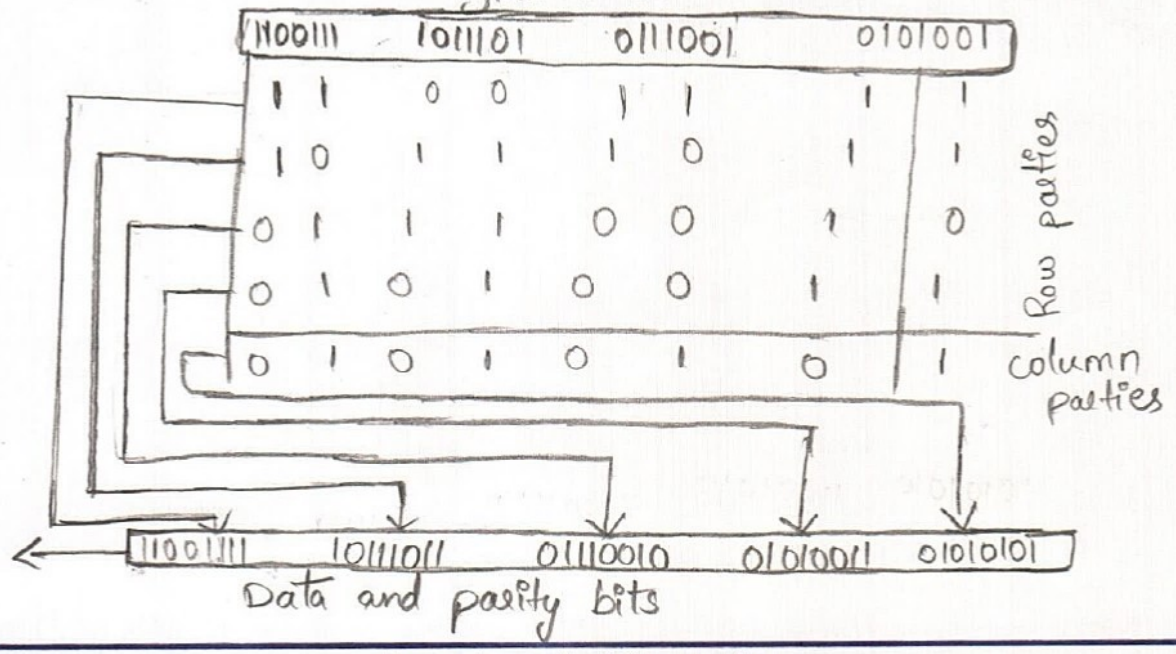
8. Aim :-

To write a Java program to implement the Two-Dimensional redundancy parity check system to compute block check Validation bits.

Description :-

standard 1D parity checks only count ones inside a single line slice, failing if an even number of bit shifts occur. A 2D structural approach organizes binary strings into blocks of specify rows and columns. parity tracks equations are run horizontally across every individual row array row layer and vertically down every column structure column section. This two-axis checksum allows the receiver to detect burst failures and pinpoint the exact row-column intersection of a single-bit flip error for correction. Two Dimensional parity check (LRC)

Longitudinal Redundancy check



Date :

GATE Educational Institutions

Page No. : 29

Practical No. :

// 6. A Java program to implement the Two-Dimensional Redundancy Parity Check system to compute block check validation bits.

Source Code:

```
import java.util.Scanner;
public class TwoDimensionalParity {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter total structural Rows to allocate: ");
      int rows = sc.nextInt();
      System.out.print("Enter total data bit Columns per Row: ");
      int cols = sc.nextInt();
      // Matrix dimensions include space for parity columns and row bits
      int[][] dataGrid = new int[rows + 1][cols + 1];
      System.out.println("Enter binary bit matrices tokens (Enter 0 or 1 line by line):");
      for (int i = 0; i < rows; i++) {
         int rowOnesCount = 0;
         for (int j = 0; j < cols; j++) {
            dataGrid[i][j] = sc.nextInt();
            if (dataGrid[i][j] == 1) {
               rowOnesCount++;
            }
         }
         // Assign calculated horizontal row bit check (Even parity logic)
         dataGrid[i][cols] = (rowOnesCount % 2 == 0) ? 0 : 1;
      }
      // Calculate vertical column parity values across block fields
      for (int j = 0; j <= cols; j++) {
```

```
        int colOnesCount = 0;
        for (int i = 0; i < rows; i++) {
            if (dataGrid[i][j] == 1) {
                colOnesCount++;
            }
        }
        dataGrid[rows][j] = (colOnesCount % 2 == 0) ? 0 : 1;
    }
    // Print final generated transmission array block output format
    System.out.println("\n======= GENERATED 2D EVEN PARITY DATA MATRIX
=======");
    for (int i = 0; i <= rows; i++) {
        if (i == rows) {
            System.out.println("----------------------------------------");
            System.out.print("V-Parity: ");
        }
        for (int j = 0; j <= cols; j++) {
            if (j == cols && i != rows) {
                System.out.print("| H-Parity: ");
            }
            System.out.print(dataGrid[i][j] + " ");
        }
        System.out.println();
    }
System.out.println("========================================");
    sc.close();
  }
}
```

Date :	GATE Educational Institutions	Page No. : 26
Output:		Practical No. :

Enter total structural Rows to allocate: 3

Enter total data bit Columns per Row: 4

Enter binary bit matrices tokens (Enter 0 or 1 line by line):

1 1 0 0

0 1 1 1

1 0 1 0

═══ GENERATED 2D EVEN PARITY DATA MATRIX ═══

1 1 0 0 | H-Parity: 0

0 1 1 1 | H-Parity: 1

1 0 1 0 | H-Parity: 0

V-Parity: 0 0 0 1 1

═══

GATE EDUCATIONAL INSTITUTIONS TIRUPATI

conclusion :-

The Java program was succesfully implemented to peeform a Two-dimensional redundancy parity check (2D parity check). The program calculates parity bits for each row and column of the binary data block and uses them to validate the received data. The method helps in detecting and locating errors in the transmitted data, particularly single bit errors

GATE EDUCATIONAL INSTITUTIONS TIRUPATI

Bit stuffing and bit destuffing using Java.

7.a) Aim :-

To write a Java program to peyorm bit stuffering and bit destuffing on a given binary data.

Desciption :-

Bit stuffing is a technique used in computer networks to insert an extract 0 bit after every sequence of five consecutive 1 bits in the data. This help in identifying the beginning and ending of frames during data transmission.

In this program, the uses enters binary data using the Scanner class. The program performs bit stuffing by inserting 0 after five consective 1s. It then peyorms bit dustuffing by removing the inserted 0 and displays the stuffed and destuffed data.

Algorithm :-

1. Start the program.
2. Import the Scanner class for taking input from the User.
3. Read the binary data from the User.
4. Initalize a counter to count consective 1 bits.
5. Read each unit of the input data one by one.
6. If the bit is 1, increment the counter.
7. If the bit is 0, reset the counter to zero.
8. Append each bit to the stuffed data.
9. When five consecutive 1s are found, insert an

additional 0 after them.
10. Display the stuffed data.
11. Read the stuffed data bit by bit for destuffing.
12. When five consective 1s followed by 0 are found, remove the extra 0.
13. Display the destuffed data
14. Stop the program.



Date :

GATE Educational Institutions

Page No. : 30

Practical No. :

// 7. a) ~~Java Program For Bit Stuffing Using Java~~

```
import java.util.Scanner;
public class Bitstf {

   // Function to perform bit stuffing
   public static String bitStuffing(String data) {
      int count = 0;
      StringBuilder stuffedData = new StringBuilder();

      for (int i = 0; i<data.length(); i++) {
         if (data.charAt(i) == '1') {
            count++;
         } else {
            count = 0;
         }
stuffedData.append(data.charAt(i));
         // If there are five consecutive 1's, add a 0 after them
         if (count == 5 &&i != data.length() - 1) {
stuffedData.append('0');
            count = 0; // Reset count after stuffing
         }
      }
      return stuffedData.toString();
   }
 // Function to perform bit destuffing
   public static String bitDestuffing(String stuffedData) {
      int count = 0;
      StringBuilder destuffedData = new StringBuilder();

      for (int i = 0; i<stuffedData.length(); i++) {
destuffedData.append(stuffedData.charAt(i));
         if (stuffedData.charAt(i) == '1') {
            count++;
         } else {
            count = 0;
         }
         // If there are five consecutive 1's followed by 0, skip the next bit
         if (count == 5 &&i<stuffedData.length() - 1 &&stuffedData.charAt(i + 1) == '0') {
i++;
            count = 0; // Reset count after destuffing
         }
      }
      return destuffedData.toString();
```

```
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
System.out.println("Enter data to be stuffed (binary format): ");
        String data = scanner.nextLine();

        // Perform bit stuffing
        String stuffedData = bitStuffing(data);
        System.out.println("Stuffed data: " + stuffedData);
        String destuffedData = bitDestuffing(stuffedData);
System.out.println("Destuffed data: " + destuffedData);
Scanner.close();
    }
}
```

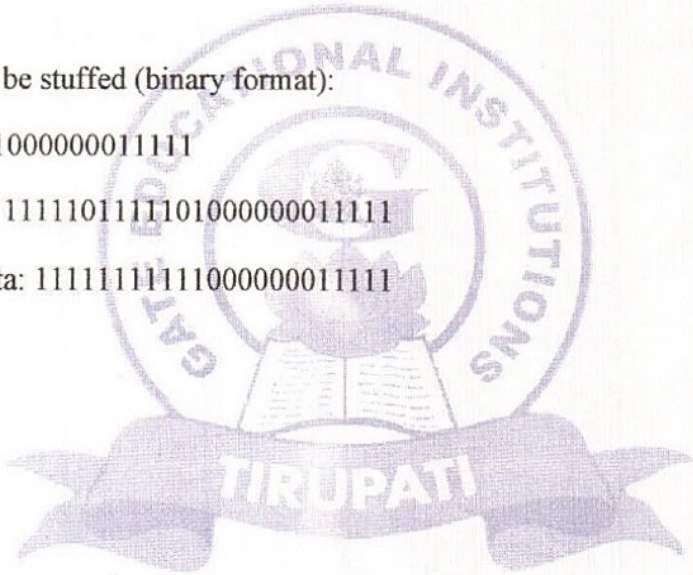
Input/Output:

Enter data to be stuffed (binary format):

11111111111000000011111

Stuffed data: 111110111111010000000011111

Destuffed data: 11111111111000000011111



conclusion :-

the Java program was Succesfully executed to peform bit Stuffing and bit destuffing on the given binary data.

GATE EDUCATIONAL INSTITUTIONS
TIRUPATI

character stuffing using Java.

7.b) Aim :-

To write a Java program to peeform chaeacter stuffing on a given sting of data.

Desciption :-

character stuffing is a technique uned in computer Neteworks to distinguish the actual data the control characters uned for framing. In this program, special chaeacters are added to the data before transmission.

The program accepts a string from the uner the Scanner class. It adds the required control chaeacters at the beginning and end of the data. If a special character sequence is found in the input, an additional escape chaeacter is inseited. Finally, the program displays the original data and the transmitted data.

Algorithm :-

1. start the program.
2. Import the lequired Java packages.
3. Create a Scanner object to lead input from the uners.
4. Read the string from the uner.
5. find the length of the input string.
6. Display the original data.
7. Initialize the string with the starting control chaeacters.
8. Read each chaeacter of the input string one by one.

9. check whether the required special character sequence is present.
10. If the sequence is found, insert the escape sequence.
11. Append the current character to the transmitted data.
12. Add the ending control characters to the transmitted data.
13. Display the transmitted data.
14. Stop the program.

GATE EDUCATIONAL INSTITUTIONS TIRUPATI

Date :

GATE Educational Institutions

Page No. : 35

Practical No. :

//7.b) Java Program For Character Stuffing Using Java

```
import java.io.*;
import java.lang.*;
import java.util.*;

class chst
{
public static void main(String args[])
        {
                Scanner k=new Scanner(System.in);
                System.out.println("Enter the string\t");
                String s=k.nextLine();
                String str1;
                String str2="";
                int i,j,m;
                m=s.length();
                System.out.println("Original Data "+s);
                str1= "dleghththtx";
                for(i=0;i<=m-1;i++)
                {
                if((s.charAt(i)=='d')&&(s.charAt(i+2)=='e'))
                {
                str1=str1+ "dle";
                }
                str1=str1+s.substring(i,i+1);
                }
                str1=str1+"dlestx";
                int p=str1.length();
                System.out.println("trasmitted data " +str1);
                }
}
```

Date :

GATE Educational Institutions

Page No. : 36

Practical No. :

Input/ Output:

```
C:\dccn>javac chst.java

C:\dccn>java chst

Enter the string

GTMT

Original Data GTMT

trasmitted data dleghththtxGTMTdlestx
```



conclusion :-

The Java program was succesfully executed to perform character stuffing on the given data and display the transmitted data.

GATE EDUCATIONAL INSTITUTIONS
TIRUPATI

CRC polynomial is sender side and Receives side.-

8. Aim :-

To design and design develop a java program to implement cyclic redundancy check (CRC) using CRC polynomials the sender side and receives side for detecting errors transmitted data.

Desciption :-

cyclic Redundancy check (CRC) is an error detection technique used in computes Networks to detect errors in data transmission. In this program, the user enters a generator polynomial and the original data.

At the sender side, zeros are appended to the data according to the length of the generator polynomial. The program peeform binary division using the generator and obtains the CRC remainder. This remainder is added to the original data to form the transmitted code word.

At the receives side, the receives code word is divided by the same generator polynomial. If the remainder is zero, the received data contains no errors. If the remainder is non-zero, the received data contains errors.

Algorithm :-

1. start the program.
2. Read the generator polynomial from the user.
3. Read the data from the user.
4. store the original data as the code.
5. Append zero to the data until its length becomes equal to the data length plus the generator length minus one.
6. perform binary division using the generator polynomial.
7. obtain the CRC remainder.
8. Append the CRC remainder to the original data to form the transmitted code coord.
9. Display the transmitted code word.
10. Read the received code word from the user.
11. Divide the received code word by the same generator polynomial.
12. check the remainder obtained after division.
13. If the remainder is zero, display "The received code word contains no errors."
14. otherwise, display "The received code word contains errors".
15. stop the program.

Date :

GATE Educational Institutions

Page No. : 40

Practical No. :

//8. Design And Develop A Java Program To Implement The CRC Polynomials In Sender Side And Receiver Side.

```
//CRC
import java.io.*;
class crc
{
 public static void main(String args[]) throws IOException
 {
 BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
 System.out.println("Enter Generator:");
 String gen = br.readLine();
 System.out.println("Enter Data:");
 String data = br.readLine();
 String code = data;
 while(code.length() < (data.length() + gen.length() - 1))
  code = code + "0";
 code = data + div(code,gen);
 System.out.println("The transmitted Code Word is: " + code);
 System.out.println("Please enter the received Code Word: ");
 String rec = br.readLine();
 if(Integer.parseInt(div(rec,gen)) == 0)
  System.out.println("The received code word contains no errors.");
 else
  System.out.println("The received code word contains errors.");
}
 static String div(String num1,String num2)
{
 int pointer = num2.length();
 String result = num1.substring(0, pointer);
 String remainder = "";
 for(int i = 0; i < num2.length(); i++)
 {
  if(result.charAt(i) == num2.charAt(i))
   remainder += "0";
  else
   remainder += "1";
 }
 while(pointer < num1.length())
 {
  if(remainder.charAt(0) == '0')
  {
   remainder = remainder.substring(1, remainder.length());
   remainder = remainder + String.valueOf(num1.charAt(pointer));
```

```
    pointer++;
   }
   result = remainder;
   remainder = "";
   for(int i = 0; i < num2.length(); i++)
   {
    if(result.charAt(i) == num2.charAt(i))
     remainder += "0";
    else
     remainder += "1";
   }
  }
  return remainder.substring(1,remainder.length());
 }
}
```

Output1:

Enter Generator:

1011

Enter Data:

1001010

The transmitted Code Word is: 1001010111

Please enter the received Code Word:

1001010111

The received code word contains no errors.

Output2:

Enter Generator:

1011

Enter Data:

1001010

The transmitted Code Word is: 1001010111

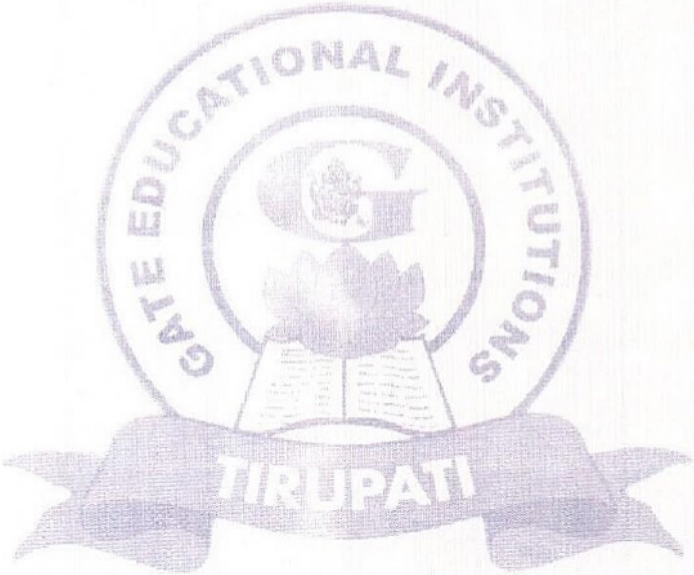
Please enter the received Code Word:

1001010011

The received code word contains errors.

conclusion :-

The Java program was succesfully executed to implement CRC error detection at the sender and receiver sides. The program correctly identifies whether the received code word contains errors or no errors.



Data Link Layer framing method (character string)

9. Aim :-

To design and develop a Java program to implement the Data link layer framing method using character string by reading a string and displaying each character along with its position.

Description :-

The data link layer is responsible for framing the data before it is transmitted over a network. In this program, a character string is taken as input from the user using the scanner class.

The program reads the string character by character using the charAt() method. It checks the each character and displays the character and its position in the string. Blank spaces are ignored. Thus, the program demonstrates the handling of characters in the string as part of a simple character-based framing method.

Algorithm :-

1. Start the program.
2. Import the Scanner class.
3. Create a scanner object to read input from the user.
4. Read a string from the user by using nextLine().
5. pass the string to the printstring characters() method.
6. find the length of the string using the charAt() method.

7. Read each character using the charAt() method.
8. check whether the character is a blank space.
9. If the character is not a black space, display its position and character.
10. Repeat the proceu until all characters are procexed.
11. program stop.
stop the program.



Date :

GATE Educational Institutions

Page No. : 95

Practical No. :

//9. Design And Develop A Java Program For Implement Of The Data Link Layer Framing Method (Character String).

```
import java.util.Scanner;
public class PrintStringChars5 {
        private static Scanner sc;
        public static void main(String[] args) {
                String str;
                sc= new Scanner(System.in);
                System.out.print("\nPlease Enter any String to Print =  ");
                str = sc.nextLine();
                PrintStringCharacters(str);
        }
        public static void PrintStringCharacters(String str) {

                for(int i = 0; i < str.length(); i++)
                {
                        if(str.charAt(i) != ' ') {
                                System.out.println("The Character at Position " + i + " = " +
str.charAt(i));
                        }
                }
        }
}
```

Input/output:

Please Enter any String to Print = GATEMCA

The Character at Position 0 = G

The Character at Position 1 = A

The Character at Position 2 = T

The Character at Position 3 = E

The Character at Position 4 = M

The Character at Position 5 = C

The Character at Position 6 = A

conclusion :-

The Java program was succesfully executed to implement the Data link Layer character string framing method and display method and display each character along with the position.

parity check bit (Even & odd)

10. Aim :-

To design and develop a Java program to implement the parity check bit using even parity and odd parity for detecting errors in transmitted binary data.

Description :-

parity check is an error detection technique used in computer networks to detect errors during data transmission. In this method, an additional bit called the parity bit is added to the binary data.

In Even parity, the parity bit is added so that the total number of 1s in the transmitted data becomes even. In odd parity, the parity bit is added so that the total number of 1s becomes odd.

In this program, the sender enters binary data and selects either even or odd parity. The program add the appropriate parity bit to the data. At the receiver side, the received data is checked according to the selected parity. If the parity condition is satisfied, the displays that program there is no error; otherwise, it displays that can an error has been detected.

Algorithm :-

1. start the program.
2. create an object of the oddEven parity class.
3. At the sender side, read the binary data from the user.
4. check whether the entered data contains only 0 and 1.
5. Ask the user to select Even parity or odd parity.
6. count the number of 1s in the binary data.
7. for Even parity :
   * If the number of 1s is even, append 0.
   * If the number of 1s is odd, append 1.
8. for odd parity :
   * Display the number of 1s is odd, append 0.
   * If the number of 1s is even, append 1.
9. Display the data with the parity bit.
10. At the received side, read the received binary data.
11. count the number of 1s in the received data.
12. for Even parity, check whether the number of 1s is even.
13. for odd parity, check whether the number of 1s is odd.
14. If the required parity condition is satisfied, display "Data received succesfully with no error".
15. Other wise, display "Error has been detected in received data".
16. stop the program.

//10. Design And Develop A Java Program For Implementation Of The Parity Check Bit (Even And Odd) .

Practical No. : 1

```
package oddevenparity;
import java.io.*;
import java.util.*;
public class OddEvenParity {
String dataR=new String();
String dataS=new String();
int i,option,len;
int count1,count0;
void getDataSender()throws IOException{
System.out.println("----------At Sender----------");
System.out.println("Enter the Binary data:");
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
dataS=br.readLine();
len=dataS.length();
for(i=0;i<dataS.length();i++){
if(dataS.charAt(i)!='1' && dataS.charAt(i)!='0'){
System.out.println("Enter only binary values");
return;
}
}
System.out.println("Even parity or Odd Parity?");
System.out.println("Enter 1 for even or 2 for odd");
Scanner s=new Scanner(System.in);
option=s.nextInt();
if(option==1){
System.out.println("Sender has agreed for Even parity");
even();
}
else{
System.out.println("Sender has agreed for odd parity");
odd();
}
System.out.println();
}
void even()throws IOException{
count1=count0=0;
for(i=0;i<dataS.length();i++){
if(dataS.charAt(i)=='1'){
count1++;
}
if(dataS.charAt(i)=='0'){
```

```
count0++;
}
}
if(count1%2==0){
dataS=dataS+'0';
System.out.println("Data after even parity:");
for(i=0;i<dataS.length();i++){
System.out.print(dataS.charAt(i));
}
}
else{
dataS=dataS+'1';
System.out.println("Data after even parity:");
for(i=0;i<dataS.length();i++){
System.out.print(dataS.charAt(i));
}
}
}
void odd()throws IOException{
count1=count0=0;
for(i=0;i<dataS.length();i++){
if(dataS.charAt(i)=='1'){
count1++;
}
if(dataS.charAt(i)=='0'){
count0++;
}
}
if(count1%2!=0){
dataS=dataS+'0';
System.out.println("Data after odd parity:");
for(i=0;i<dataS.length();i++){
System.out.print(dataS.charAt(i));
}
}
else{
dataS=dataS+'1';
System.out.println("Data after odd parity:");
for(i=0;i<dataS.length();i++){
System.out.print(dataS.charAt(i));
}
}
}
void getDataReciver()throws IOException{
```

```
System.out.println("\n\n----------At Receiver----------");
System.out.println("Enter the Binary data:");
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
dataR=br.readLine();
len+=1;
if(len!=dataR.length()){
System.out.println("Enter the Binary data of length: "+len);
return;
}
for(i=0;i<dataR.length();i++){
if(dataR.charAt(i)!='1' && dataR.charAt(i)!='0'){
System.out.println("Enter only binary values");
return;
}
}
count1=count0=0;
for(i=0;i<dataR.length();i++){
if(dataR.charAt(i)=='1'){
count1++;
}
if(dataR.charAt(i)=='0'){
count0++;
}
}
if(option==1){
if(count1%2==0){
System.out.println("Data received successfully with no error");

}
else{
System.out.println("Error has been detected in received data");
}
}
else{
if(count1%2!=0){
System.out.println("Data received successfully with no error");
//System.out.println("Error has been detected in received data");
}
else{
System.out.println("Error has been detected in received data");
}
}
System.out.println();
}
```

```
public static void main(String[] args) throws IOException {
OddEvenParity p=new OddEvenParity();
p.getDataSender(),
p.getDataReciver();
}
}
```

Output:

Output1:

----------At Sender----------
Enter the Binary data:
01101110101
Even parity or Odd Parity?
Enter 1 for even or 2 for odd
2
Sender has agreed for odd parity
Data after odd parity:
011011101010

----------At Receiver----------
Enter the Binary data:
011011101010
Data received successfully with no error

Output2:

----------At Sender----------
Enter the Binary data:
101100111011
Even parity or Odd Parity?
Enter 1 for even or 2 for odd
1
Sender has agreed for Even parity
Data after even parity:
1011001110110

----------At Receiver----------
Enter the Binary data:
1011001110111
Error has been detected in received data

conclusion :-

The Java program was succesfully executed to implement Even parity and odd parity checking and to detect errors in the received binary data.

GATE EDUCATIONAL INSTITUTIONS TIRUPATI

RSA Algorithm for string Encryption

11. Aim :-

To write a Java program to implement the RSA algorithm for string encryption and decryption using public and private keys.

Description :-

RSA is a public-key cryptographic algorithm used for encrypting and decrypting data. It uses two different keys : a public key for encryption and a private key for Decryption.

In this program, two prime numbers p and q are entered by the user. The program calculates n and Euler's totient value Φ. It then generates the public key e and private key d. The input string is converted into ASCII values, and each value is encrypted using the public key. The encrypted values are then decrypted using the private key and converted back into the original string.

Algorithm :-

1. start the program.
2. Import the required Java packages.
3. Read two prime numbers p and q from the user.
4. calculate $n = p \times q$
5. calculate $\Phi = (p-1) \times (q-1)$
6. find a value e such that

$1 < e < \Phi$ and $GCD(e, \Phi) = 1$

7. Calculate the private key d such that (dxe) mod Q = 1
8. Display the public key e and private key d.
9. Read the message as a string.
10. convert each character of the string into its ASCII value.
11. Encrypt each ASCII value using the public key (e,n)
12. Display the encrypted message.
13. Decrypt the encrypted values using the private key (d,n)
14. convert the decrypted ASCII values back into characters.
15. Display the original message after decryption.
16. Stop the program.

GATE EDUCATIONAL INSTITUTIONS TIRUPATI

//11.Java Program to Implement the RSA Algorithm for String Encryption

```
import java.math.*;
import java.util.*;
import java.math.BigInteger;
import java.util.Scanner;
public class RSA {

  public static void main(String args[]) {
    int p, q, n, phi, e, d = 0, i;
    Scanner scanner = new Scanner(System.in);

    // Input two prime numbers
    System.out.println("Enter two prime numbers:");
    p = scanner.nextInt();
    q = scanner.nextInt();

    // Calculating n and phi
    n = p * q;
    phi = (p - 1) * (q - 1);

    // Generating public key
    for (e = 2; e < phi; e++) {
      // e must be less than phi and GCD of (e, phi) must be 1
      if (gcd(e, phi) == 1) {
        break;
      }
    }
    System.out.println("Public key (e): " + e);

    // Generating private key
    for (i = 0; ; i++) {
      int x = 1 + (i * phi);
      if (x % e == 0) {
        d = x / e;
        break;
      }
    }
    System.out.println("Private key (d): " + d);

    // Input message as a string
    System.out.println("Enter a message to encrypt:");
    scanner.nextLine(); // Consume newline left from previous nextInt()
    String plainTextStr = scanner.nextLine();
```

```
    // Convert string to ASCII numeric representation
    int[] asciiValues = stringToAscii(plainTextStr);

    // Encrypting the data
    BigInteger[] encryptedMsg = new BigInteger[asciiValues.length];
    for (int j = 0; j < asciiValues.length; j++) {
        BigInteger msg = BigInteger.valueOf(asciiValues[j]);
        encryptedMsg[j] = msg.modPow(BigInteger.valueOf(e), BigInteger.valueOf(n));
    }
    System.out.println("Encrypted message:");

    for (BigInteger msg : encryptedMsg) {
        System.out.print(msg + "");
    }
    System.out.println();

    // Decrypting the data
    int[] decryptedValues = new int[encryptedMsg.length];
    for (int j = 0; j < encryptedMsg.length; j++) {
        BigInteger decryptedMsg = encryptedMsg[j].modPow(BigInteger.valueOf(d),
BigInteger.valueOf(n));
        decryptedValues[j] = decryptedMsg.intValue();
    }

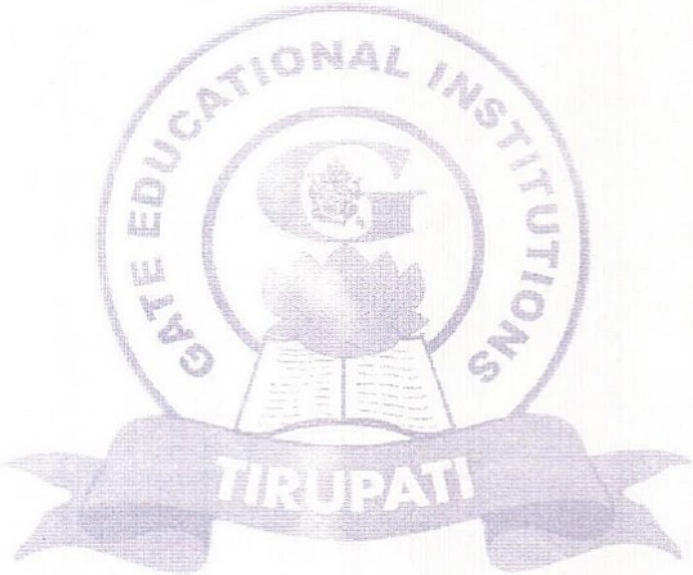
    // Convert ASCII values back to string
    String decryptedStr = asciiToString(decryptedValues);
    System.out.println("Original Message After Decryption: " + decryptedStr);
  }

  // Method to compute GCD
  static int gcd(int e, int phi) {
    if (e == 0)
        return phi;
    else
        return gcd(phi % e, e);
  }

  // Convert a string to an array of ASCII values
  static int[] stringToAscii(String str) {
    int[] asciiValues = new int[str.length()];
    for (int i = 0; i < str.length(); i++) {
        asciiValues[i] = (int) str.charAt(i);
    }
    return asciiValues;
```

```
    }

    // Convert an array of ASCII values back to a string
    static String asciiToString(int[] asciiValues) {
        StringBuilder sb = new StringBuilder();
        for (int ascii : asciiValues) {
            sb.append((char) ascii);
        }
        return sb.toString();
    }
}
```



Date :

GATE Educational Institutions

Page No. : 59

Practical No. :

Input/Output:

Enter two prime numbers:

7

13

Public key (e): 5

Private key (d): 29

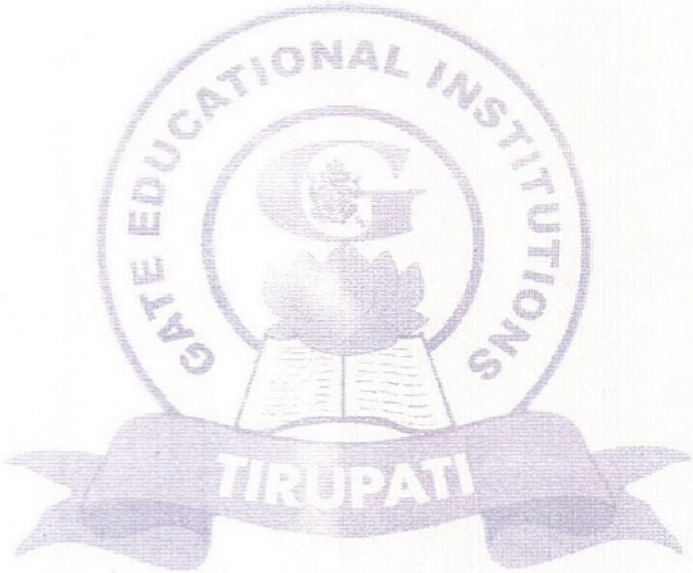
Enter a message to encrypt:

GATEINSTITUTIONS

Encrypted message:

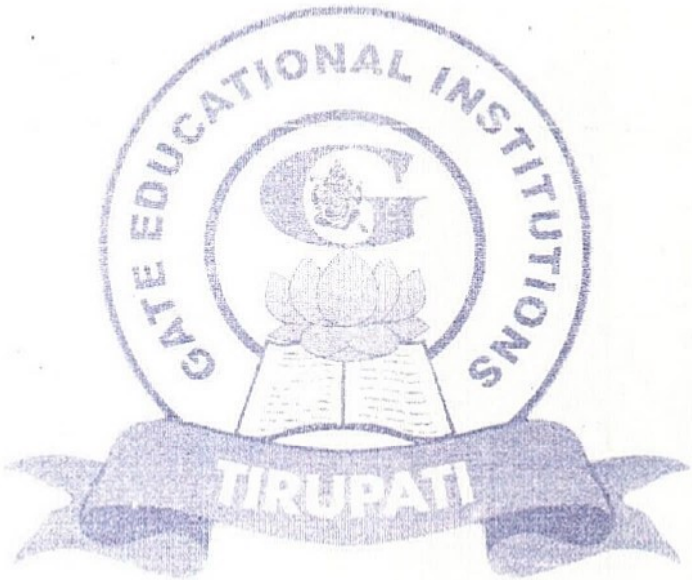
153928624778832847285028475378 83

Original Message After Decryption: **GATEINSTITUTIONS**



Conclusion :-

The Java program was succesfully executed to implement the RSA algorithm for string encryption and decryption, and the original message was succesfully recovered after decryption.



playfair Cipher substitution technique

12. Aim :-

To write and execute a Java program to implement playfair cipher substitution Technique for encrypting a given plaintext using a secret key.

Description :-

The playfair cipher is a classical substitution cipher that encrypts plaintext in pairs of characters instead of encrypting individual characters. It uses a 5×5 matrix generated from a given key. The letter of J is usually combinated with I, resulting in 25 characters in matrix.

In this program, the user enters a key and plaintext. The program creates a 5×5 encryption matrix using the key and the remaining alphabet characters. The plaintext is divided into pairs of characters. If two consecutive characters are the same, an x is inserted between them. If the plaintext has an odd number of characters, an x is added to the end. Each pair is encrypted according to the playfair rules: characters in the same row are replaced by the characters formating to the right, characters in the same columns below them, and characters forming a rectangle are replaced by the characters at the opposite corners. Finally, the encrypted text is displayed as the cipher text.

Algorithm :-

1. Start the program.
2. Import the required Java packages.
3. Read the key from the user.
4. Convert the key to uppercase and replace J with I.
5. Read the plaintext from the user.
6. Convert the plaintext to uppercase and replace J with I.
7. Format the plaintext into pairs of characters.
8. Insert X between repeated characters in a pair.
9. If the plaintext has an odd number of characters, append X.
10. Create a 5x5 matrix using the characters of the key.
11. Fill the remaining positions with unused letter of alphabet.
12. Display the encryption matrix.
13. Take the plaintext two characters at a time.
14. Find the positions of both characters in the matrix.
15. If both characters are in the same row, replace with each character to its right.
16. If both characters are in the same column, replace each with the character below it.
17. Otherwise, treat the two characters as opposite corners of a rectangle and replace them with the characters at the other corners.
18. Repeat the process for all character pairs.
19. Display the resulting cipher text.
20. Stop the program.

Date :

GATE Educational Institutions

Page No. : 63

Practical No. :

//12. A java program to execute and playfair Cipher Substitution technique using java.

```
import java.util.*;

public class PlayfairCipher {

    private static final String ALPHABET = "ABCDEFGHIKLMNOPQRSTUVWXYZ"; // 'J'
is omitted

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        // Input key for Playfair Cipher

        System.out.println("Enter the key for Playfair Cipher:");

        String key = scanner.nextLine().toUpperCase().replace("J", "I");

        // Input plaintext

        System.out.println("Enter the plaintext:");

        String plaintext = scanner.nextLine().toUpperCase().replace("J", "I");

        // Format plaintext

        String formattedPlaintext = formatPlaintext(plaintext);

        char[][] matrix = createMatrix(key);

        System.out.println("Encryption Matrix:");

        printMatrix(matrix);

        // Encrypt the plaintext

        String ciphertext = encrypt(formattedPlaintext, matrix);

        System.out.println("Ciphertext: " + ciphertext);

        scanner.close();

    }  // Format plaintext into digraphs

    private static String formatPlaintext(String text) {

        StringBuilder formatted = new StringBuilder();

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

            char ch = text.charAt(i);
```

```
        if (ALPHABET.indexOf(ch) != -1) {
            formatted.append(ch);
            if (i < text.length() - 1 && text.charAt(i) == text.charAt(i + 1)) {
                formatted.append('X'); // Insert 'X' between repeated letters
            }
        }
    }
    if (formatted.length() % 2 != 0) {
        formatted.append('X'); // Add 'X' if odd length
    }  return formatted.toString();
}  // Create the 5x5 matrix from the key
private static char[][] createMatrix(String key) {
    Set<Character> seen = new HashSet<>();
    char[][] matrix = new char[5][5];
    int index = 0;
    // Add characters from the key
    for (char ch : key.toCharArray()) {
        if (seen.add(ch) && ALPHABET.indexOf(ch) != -1) {
            matrix[index / 5][index % 5] = ch;
            index++;
        }
    }
    // Add remaining letters of the alphabet
    for (char ch : ALPHABET.toCharArray()) {
        if (seen.add(ch)) {
            matrix[index / 5][index % 5] = ch;
            index++;
```

```
        }
    }
        return ciphertext.toString();
    }
    // Find the position of a character in the matrix
    private static int[] findPosition(char ch, char[][] matrix) {
        for (int i = 0; i < 5; i++) {
            for (int j = 0; j < 5; j++) {
                if (matrix[i][j] == ch) {
                    return new int[]{i, j};
                }
            }
        }
        return new int[]{-1, -1}; // Not found
    }
}
```

Input/output:

Enter the key for Playfair Cipher: GATEMCA

Enter the plaintext: TODAY IS SATURDAY

Encryption Matrix:

[G, A, T, E, M]

[C, B, D, F, H]

[I, K, L, N, O]

[P, Q, R, S, U]

[V, W, X, Y, Z]

Ciphertext: MLBTVNRYQEMRXLEW

Conclusion :—

The Java program was succesfully executed to implement the playfair Cipher substitution Technique and generate the encrypted Ciphertext for the given plaintext and key.

