```
/////////        server         ///////////////

package socket;

import java.io.*;
import java.net.*;
public class server{
    public static void main(String[] args) throws
Exception{

        ServerSocket s = new ServerSocket(9999);
        Socket ss = s.accept();
        System.out.println("Coonected");
        DataInputStream dout = new
DataInputStream(ss.getInputStream());
        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
        while(true){
             String yoo = dout.readUTF();
             System.out.println("client :"+yoo);
                if(yoo.equalsIgnoreCase("exit"))
                    break;
        }
        ss.close();
    }
}


/////////          client           //////////////

package socket;

import java.io.*;
import java.net.*;
public class client {
    public static void main(String[] args) throws
Exception{
       //tcp end to end conn
     Socket s = new Socket("localhost",9999);
     DataOutputStream dout = new
DataOutputStream(s.getOutputStream()); //convert data
into streams
     BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
     while(true){
         String so = br.readLine();
         dout.writeUTF(so);
         if(so.equalsIgnoreCase("exit"))
             break;
     }
     s.close();
    }
}
```

```
////////////          server            ///////////////

import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.util.Scanner;

public class server{
    public static void main(String[] args) throws IOException {
        DatagramSocket serverSocket = new DatagramSocket(4160);
        byte[] buf = new byte[256];
            DatagramPacket receivePacket = new DatagramPacket(buf,buf.length);
            serverSocket.receive(receivePacket);
            String resp = new String(receivePacket.getData());
            System.out.println("Response data: " +resp);
    }
}


/////////////           client            /////////////////

import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;

public class client {
    public static void main(String[] args) throws IOException {
        DatagramSocket cSocket = new DatagramSocket();
        Scanner sc=new Scanner(System.in);
        InetAddress Address = InetAddress.getByName("localhost");
        while(true) {
            String str = sc.next();
            byte[] buf = str.getBytes();
            DatagramPacket p = new DatagramPacket(buf, buf.length,Address,
4160);
            cSocket.send(p);
        }
    }
}
```

```
#include<stdio.h>
struct node
{
      unsigned dist[20];
      unsigned from[20];
}rt[10];
int main()
{
      int dmat[20][20];
      int n,i,j,k,count=0;
      printf("\nEnter the number of nodes : ");
      scanf("%d",&n);
      printf("\nEnter the cost matrix :\n");
      for(i=0;i<n;i++)
            for(j=0;j<n;j++)
            {
                  scanf("%d",&dmat[i][j]);
                  dmat[i][i]=0;
                  rt[i].dist[j]=dmat[i][j];
                  rt[i].from[j]=j;
            }
            do
            {
                  count=0;
                  for(i=0;i<n;i++)
                  for(j=0;j<n;j++)
                  for(k=0;k<n;k++)
                        if(rt[i].dist[j]>dmat[i][k]+rt[k].dist[j])
                        {
                              rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j];
                              rt[i].from[j]=k;
                              count++;
                        }
            }while(count!=0);
            for(i=0;i<n;i++)
            {
                  printf("\n\nWhen source is router %d \n",i+1);
                  for(j=0;j<n;j++)
                  {
                        printf("\t\n destination %d via %d shortest path = 
%d",j+1,rt[i].from[j]+1,rt[i].dist[j]);
                  }
            }
      printf("\n\n");
}
```

```c
#include<stdio.h>
#include<string.h>

#define N strlen(gen_poly)

char data[28];
char check_value[28];
char gen_poly[10];
int data_length,i,j;

void XOR(){
    for(j = 1;j < N; j++)
    check_value[j] = (( check_value[j] == gen_poly[j])?'0':'1');
}

void receiver(){
    printf("Enter the received data: ");
    scanf("%s", data);
    printf("\n-----------------------------\n");
    printf("Data received: %s", data);

    crc();

    for(i=0;(i<N-1) && (check_value[i]!='1');i++);
        if(i<N-1)
            printf("\nError detected\n\n");
        else
            printf("\nNo error detected\n\n");
}

void crc(){
    for(i=0;i<N;i++)
        check_value[i]=data[i];
    do{
        if(check_value[0]=='1')
            XOR();

        for(j=0;j<N-1;j++)
            check_value[j]=check_value[j+1];

        check_value[j]=data[i++];
    }while(i<=data_length+N-1);
}

int main()
{
    printf("\nEnter data to be transmitted: ");
    scanf("%s",data);
    printf("\n Enter the Generating polynomial: ");
    scanf("%s",gen_poly);

    data_length=strlen(data);

    for(i=data_length;i<data_length+N-1;i++)
```

```
        data[i]='0';

    printf("\n----------------------------------------");
    printf("\n Data padded with n-1 zeros : %s",data);
    printf("\n----------------------------------------");

    crc();

    printf("\nCRC or Check value is : %s",check_value);

    for(i=data_length;i<data_length+N-1;i++)
        data[i]=check_value[i-data_length];
    printf("\n----------------------------------------");
    printf("\n Final data to be sent : %s",data);
    printf("\n----------------------------------------\n");

    receiver();

        return 0;
}
```

```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>

long int p, q, n, t, flag, e[100], d[100], temp[100], j, m[100], en[100],
i;
char msg[100];
int prime(long int);
void ce();
long int cd(long int);
void encrypt();
void decrypt();
void main()
{
    printf("\nENTER FIRST PRIME NUMBER\n");
    scanf("%d", &p);
    flag = prime(p);
    if (flag == 0)
    {
        printf("\nWRONG INPUT\n");
        getch();
        exit(1);
    }
    printf("\nENTER ANOTHER PRIME NUMBER\n");
    scanf("%d", &q);
    flag = prime(q);
    if (flag == 0 || p == q)
    {
        printf("\nWRONG INPUT\n");
        getch();
        exit(1);
    }
    printf("\nENTER MESSAGE\n");
    fflush(stdin);
    scanf("%s", msg);
    for (i = 0; msg[i] != NULL; i++)
        m[i] = msg[i];
    n = p * q;
    t = (p - 1) * (q - 1);
    ce();
    printf("\nPOSSIBLE VALUES OF e AND d ARE\n");
    for (i = 0; i < j - 1; i++)
        printf("\n%ld\t%ld", e[i], d[i]);
    encrypt();
    decrypt();
}
int prime(long int pr)
{
    int i;
    j = sqrt(pr);
    for (i = 2; i <= j; i++)
    {
        if (pr % i == 0)
            return 0;
    }
    return 1;
}
void ce()
{
    int k;
    k = 0;
    for (i = 2; i < t; i++)
    {
        if (t % i == 0)
            continue;
        flag = prime(i);
        if (flag == 1 && i != p && i != q)
        {
            e[k] = i;
            flag = cd(e[k]);
            if (flag > 0)
            {
                d[k] = flag;
                k++;
            }
            if (k == 99)
                break;
        }
    }
}
long int cd(long int x)
{
    long int k = 1;
    while (1)
    {
        k = k + t;
        if (k % x == 0)
            return (k / x);
    }
}
void encrypt()
{
    long int pt, ct, key = e[0], k, len;
    i = 0;
    len = strlen(msg);
    while (i != len)
    {
        pt = m[i];
        pt = pt - 96;
        k = 1;
        for (j = 0; j < key; j++)
        {
            k = k * pt;
            k = k % n;
        }
        temp[i] = k;
        ct = k + 96;
        en[i] = ct;
        i++;
    }
    en[i] = -1;
    printf("\nTHE ENCRYPTED MESSAGE IS\n");
    for (i = 0; en[i] != -1; i++)
        printf("%c", en[i]);
}
void decrypt()
{
    long int pt, ct, key = d[0], k;
    i = 0;
    while (en[i] != -1)
    {
        ct = temp[i];
        k = 1;
        for (j = 0; j < key; j++)
        {
            k = k * ct;
            k = k % n;
        }
        pt = k + 96;
        m[i] = pt;
        i++;
    }
    m[i] = -1;
    printf("\nTHE DECRYPTED MESSAGE IS\n");
    for (i = 0; m[i] != -1; i++)
        printf("%c", m[i]);
}
```

```
#include <stdio.h>
#include <stdlib.h>
struct packet
{
    int time;
    int size;
} p[50];

int main()
{
    int i, n, m, k = 0;
    int bsize, bfilled, outrate;
    printf("Enter the number of packets:");
    scanf("%d", &n);
    printf("Enter packets in the order of their arrival time\n");
    for (i = 0; i < n; i++)
    {
        printf("Enter the time and size:");
        scanf("%d%d", &p[i].time, &p[i].size);
    }
    printf("Enter the bucket size:");
    scanf("%d", &bsize);
    printf("Enter the output rate:");
    scanf("%d", &outrate);

    m = p[n-1].time; //m is the time of last packet..
    i = 1;             //frame no
    k = 0;             //which pkt is it referes to
    bfilled = 0;
    while (i <= m || bfilled != 0)
    {
        printf("\n\nAt time %d", i);

        if (p[k].time == i) //checks if packets are coming in order or
not
        {
            if (bsize >= bfilled + p[k].size)
            {
                bfilled = bfilled + p[k].size;
                printf("\n%dbyte packet is inserted", p[k].size);
                k = k + 1;
            }
            else
            {
                printf("\n%dbyte packet is discarded", p[k].size);
                k = k + 1;
            }

        }
//release the packt

    if (bfilled == 0)
    {
        printf("\nNo packets to transmitte");
    }
    else if (bfilled >= outrate)
    {
        bfilled = bfilled - outrate;
        printf("\n%dbytes transfered", outrate);
    }

    else
    {
        printf("\n%dbytes transfered", bfilled);
        bfilled = 0;
    }
    printf("\nPackets in the bucket%dbyte", bfilled); //remaing space
    i++;
    }
    return 0;
}
```