

**ABESIT**

**COLLEGE CODE – 290**

**LAB FILE**

|  |  |
| --- | --- |
| **NAME** | Sandeep Kumar Shukla |
| **BRANCH** | CSE |
| **UNIVERSITY ROLL NO.** | 1729010140 |
| **SESSION** | 2019-20 |
| **NAME OF LAB** | Computer Networks Lab (RCS 651) |

**ABES INSTITUTE OF TECHNOLOGY, GHAZIABAD**

**Department of Computer Science and Engineering**

|  |  |
| --- | --- |
| **Department Vision** | To provide excellence by imparting knowledge to the learners enabling them to become skilled professionals to be recognized as a responsible citizen. |
| **Department Mission** | 1. Provide quality education in the field of computer science and engineering through experienced and qualified faculty members. 2. Motivate learners for higher studies and research oriented activities by utilizing resources of Centers of Excellence. 3. Inculcate societal values, professional ethics, team work, and leadership qualities by having exposure at National and International level activities. |
| **Institute Vision** | To be leading institution in technical education providing education and training enabling human resource to serve nation and world at par with global standards in education |
| **Institute Mission** | 1)Developing state of art infrastructure which also includes establishment of centre of excellence in pursuit of academic and technical excellence  2)Valuing work force inculcating belongingness and professional integrity  3)To develop human resource to solve local, regional and global problems to make technology relevant to those who mean it most |

**Program Educational Objectives (PEOs)**

|  |  |
| --- | --- |
| **PEO1** | Graduates of the program are expected to be employed in IT industry or  indulge in higher studies and research. |
| **PEO2** | Graduates of the program are expected to exhibit curiosity to learn new technologies and work with ethical values and team work. |
| **PEO3** | Graduates of the program are expected to design and develop innovative solutions related to real world problems of the society. |

**Program Outcomes (PSOs)**

|  |  |
| --- | --- |
| **PSO 1** | Solve complex problems using data structures and other advanced suitable algorithms. |
| **PSO 2** | Interpret fundamental concepts of computer systems and understand its hardware and software aspect. |
| **PSO 3** | Analyze the constraints of the existing data base management systems and get experience on large-scale analytical methods in the evolving technologies. |
| **PSO 4** | Develop intelligent systems and implement solutions to cater the business specific requirements. |

**Program Specific Outcomes (POs)**

|  |  |
| --- | --- |
| **PO1** | Engineering knowledge |
| **PO2** | Problem analysis |
| **PO3** | Design/development of solutions |
| **PO4** | Conduct investigations of complex problems |
| **PO5** | Modern tool usage |
| **PO6** | The engineer and society |
| **PO7** | Environment and sustainability |
| **PO8** | Ethics |
| **PO9** | Individual and team work |
| **PO10** | Communication |
| **PO11** | Project management and finance |
| **PO12** | Life-long learning |

**LIST OF EXPERIMENTS:**

|  |  |
| --- | --- |
| **No.** | **Name of Experiment** |
| 1 | To Create Cross-Over and straight through cable. |
| 2 | Study & Implementaion of Network IP and Sub netting. |
| 3 | Connection of Computer in LAN & Configuration of router, hub, switch etc using simulators. |
| 4 | Running and using services/commands like ping, trace route, telnet,ftp etc. |
| 5 | Configure a network topology using packet tracer software |
| 6 | Configure a Network using Distance Vector and Link State Routing protocol |
| 7 | To write a program to Calculate Checksum. |
| 8 | To write a Program to compute Hamming Distance between Two Data word. |
| 9 | To write a program for error detecting code using CRC-CCITT(16-bits). |
| 10 | To write a program to Implement Hamming Code |
| 11 | To write a program for simple RSA algorithm to encrypt and decrypt the data. |
| 12 | To write a Program to Compute Link Utilization on Stop and Wait Protocol |
| 13 | To write a program for bit stuffing used in HDLC |
| 14 | To write a program for Routing algorithm to find Shortest Path. |
| 15 | To write a program to find Class of IP address and Network Address for Given IP address. |

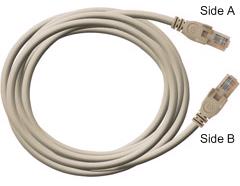
**Experiment No. 1**

**AIM: To Create Cross-Over and straight through cable.**

Common Ethernet network cable are straight and crossover cable. This Ethernet network cable is made of 4 pair high performance cable that consists twisted pair conductors that used for data transmission. Both end of cable is called RJ45 connector.

The cable can be categorized as **Cat 5, Cat 5e, Cat 6 UTP cable**. Cat 5 UTP cable can support 10/100 Mbps Ethernet network, whereas Cat 5e and Cat 6 UTP cable can support Ethernet network running at 10/100/1000 Mbps. You might heard about Cat 3 UTP cable, it's not popular anymore since it can only support 10 Mbps Ethernet network.

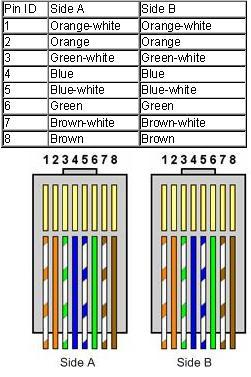
Straight and crossover cable can be Cat3, Cat 5, Cat 5e or Cat 6 UTP cable, the only difference is each type will have different wire arrangement in the cable for serving different purposes.



### **Straight Cable**

You usually use straight cable to connect different type of devices. This type of cable will be used most of the time and can be used to:

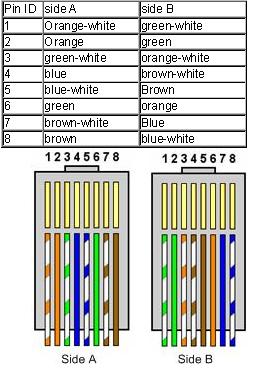
1) Connect a computer to a switch/hub's normal port.   
2) Connect a computer to a cable/DSL modem's LAN port.   
3) Connect a router's WAN port to a cable/DSL modem's LAN port.   
4) Connect a router's LAN port to a switch/hub's uplink port. (Normally used for expanding network)   
5) Connect 2 switches/hubs with one of the switch/hub using an uplink port and the other one using normal port.

**Both sides (side A and side B) of cable have wire arrangement with same color**. Check out different types of straight cablehttps://www.assoc-amazon.com/e/ir?t=homnetwirneta-20&l=ur2&o=1 that are available in the market here.

### **Crossover Cable**

Sometimes you will use crossover cable, it's usually used to connect same type of devices. A crossover cable can be used to:

1)Connect2computersdirectly.   
2) Connect a router's LAN port to a switch/hub's normal port. (normally used for expanding network)   
3) Connect 2 switches/hubs by using normal port in both switches/hubs.

In you need to check how crossover cable looks like, **both side (side A and side B) of cable have wire arrangement with following different color**. Have a look on these [crossover cables](http://www.amazon.com/gp/search?ie=UTF8&keywords=crossover%20cable&tag=homnetwirneta-20&index=blended&linkCode=ur2&camp=1789&creative=9325)https://www.assoc-amazon.com/e/ir?t=homnetwirneta-20&l=ur2&o=1 if you plan to buy one. You can also find more network cable choices and information from [Comtrad Cables](http://www.comtradcables.com/).

**Experiment No. 2**

**AIM: NETWORK IP AND SUB NETTING.**

Following is required to be study under this practical.

Classification of IP address

Class Address Range Supports

**Class A:** 1.0.0.1 to 126.255.255.254 Supports 16 million hosts on each of 127 networks.

**Class B:** 128.1.0.1 to 191.255.255.254 Supports 65,000 hosts on each of 16,000 networks.

**Class C:** 192.0.1.1 to 223.255.254.254 Supports 254 hosts on each of 2 million networks.

**Class D:** 224.0.0.0to 239.255.255.255 Reserved for multicast groups.

**Class E:** 240.0.0.0 to 254.255.255.254 Reserved

**SUB NETTING**

Subnetting is a process of dividing large network into the smaller networks based on layer 3 IP address. Every computer on network has an IP address that represent its location on network. Two version of IP addresses are available IPv4 and IPv6. In this article we will perform subnetting on IPv4.

IPv4

IP addresses are displayed in dotted decimal notation, and appear as four numbers separated by dots. Each number of an IP address is made from eight individual bits known as octet. Each octet can create number value from 0 to 255. An IP address would be 32 bits long in binary divided into the two components, network component and host component. Network component is used to identify the network that the packet is intend for, and host component is used to identify the individual host on network.

IP addresses are broken into the two components

**Network component: -** Defines network segment of device.

**Host component: -** Defines the specific device on a particular network segment.

IP Classes in decimal notation

Class A addresses range from 1-126 Class B addresses range from 128-191 Class C addresses range from 192-223 Class D addresses range from 224-239 Class E addresses range from 240-254

* + - * + 0 [Zero] is reserved and represents all IP addresses.
        + 127 is a reserved address and is used for testing, like a loop back on an interface.
        + 255 is a reserved address and is used for broadcasting purposes.

**Subnet mask**

Subnet mask is a 32 bits long address used to distinguish between network address and host address in IP address. Here we need subnet mask to get details about network address and host address.

* In decimal notation subnet mask value 1 to 255 represent network address and value 0 [Zero] represent host address.
* In binary notation subnet mask ON bit [ 1] represent network address while OFF bit[0] represent host address.

In decimal notation

IP address 192.168.1.10

Subnet mask 255.255.255.0

Network address is 192.168.1 and host address is 10.

In binary notation

IP address 11000000.10101000.00000001.00001010

Subnet mask 11111111.11111111.11111111.00000000

Network address is 11000000.10101000.00000001 and host address is 00001010

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| IP Class | Default Subnet | Network bits | Host bits | Total hosts | Valid hosts |
| A | 255.0.0.0 | First 8 bits | Last 24 bits | 16, 777, 216 | 16, 777, 214 |
| B | 255.255.0.0 | First 16 bits | Last 16 bits | 65,536 | 65,534 |
| C | 255.255.255.0 | First 24 bits | Last 8 bits | 256 | 254 |

**Network ID**

First address of subnet is called network ID. This address is used to identify one segment or broadcast domain from all the other segments in the network.

**Block Size**

Block size is the size of subnet including network address, hosts addresses and broadcast address.

**Broadcast ID**

There are two types of broadcast, direct broadcast and full broadcast.

Direct broadcast or local broadcast is the last address of subnet and can be hear by all hosts in subnet.

Full broadcast is the last address of IP classes and can be hear by all IP hosts in network. Full broadcast address is 255.255.255.255

The main difference between direct broadcast and full broadcast is that routers will not propagate local broadcasts between segments, but they will propagate directed broadcasts.

**Host Addresses**

All address between the network address and the directed broadcast address is called host address for the subnet. You can assign host addresses to any IP devices such as PCs, servers, routers, and switches.

**Experiment No. 3**

**AIM: CONNECTION OF COMPUTER IN LAN**

#### **Procedure**: On the host computer

On the host computer, follow these steps to share the Internet connection:

1. Log on to the host computer as Administrator or as Owner.
2. Click Start, and then click Control Panel.
3. Click Network and Internet Connections.
4. Click Network Connections.
5. Right-click the connection that you use to connect to the Internet. For example, if you connect to the Internet by using a modem, right-click the connection that you want under Dial-up / other network available.
6. Click Properties.
7. Click the Advanced tab.
8. Under Internet Connection Sharing, select the Allow other network users to connect through this computer's Internet connection check box.
9. If you are sharing a dial-up Internet connection, select the Establish a dial-up connection whenever a computer on my network attempts to access the Internet check box if you want to permit your computer to automatically connect to the Internet.
10. Click OK. You receive the following message:

When Internet Connection Sharing is enabled, your LAN adapter will be set to use IP address 192.168.0.1. Your computer may lose connectivity with other computers on your network. If these other computers have static IP addresses, it is a good idea to set them to obtain their IP addresses automatically. Are you sure you want to enable Internet Connection Sharing?

1. Click Yes.

The connection to the Internet is shared to other computers on the local area network (LAN). The network adapter that is connected to the LAN is configured with a static IP address of 192.168.0.1 and a subnet mask of 255.255.255.0

On the client computer

To connect to the Internet by using the shared connection, you must confirm the LAN adapter IP configuration, and then configure the client computer. To confirm the LAN adapter IP configuration, follow these steps:

1. Log on to the client computer as Administrator or as Owner.
2. Click Start, and then click Control Panel.
3. Click Network and Internet Connections.
4. Click Network Connections.
5. Right-click Local Area Connection and then click Properties.
6. Click the General tab, click Internet Protocol (TCP/IP) in the connection uses the following items list, and then click Properties.
7. In the Internet Protocol (TCP/IP) Properties dialog box, click Obtain an IP address automatically (if it is not already selected), and then click OK.

8. IP Address 192.168.31.202

9. Subnet mask 255.255.255.0

1. Default gateway 192.168.31.1
2. In the Local Area Connection Properties dialog box, click OK.

**Experiment No. 4**

**AIM: Running and using services/commands like ping, trace route, telnet,ftp etc**

**PING**: ping(8) sends an ICMP *ECHO\_REQUEST* packet to the specified host. If the host responds, you get an ICMP packet back. Sound strange? Well, you can “ping” an IP address to see if a machine is alive. If there is no response, you know something is wrong. Here is an example conversation between two Linux users:

*User* Loki's down again.

*User B*: Are you sure?

*User A*: Yeah, I tried pinging it, but there's no response.

It's instances like these that make ping a very useful day-to-day command. It provides a very quick way to see if a machine is up and connected to the network. The basic syntax is:

% **ping** [**www.google.com**](http://www.google.com)

**TRACEROUTE:** Slackware's traceroute command is a very useful network diagnostic tool. traceroute displays each host that a packet travels through as it tries to reach its destination. You can see how many “hops” from the Slackware web site you are with this command:

% **traceroute** [**www.google.com**](http://www.google.com)

**NSLOOKUP:** nslookup is a tried and true program that has weathered the ages. nslookup has been deprecated and may be removed from future releases. There is not even a man page for this program.

% nslookup 64.57.102.34

Note: nslookup is deprecated and may be removed from future releases. Consider using the `dig' or `host' programs instead. Run nslookup with the `-sil[ent]' option to prevent this message from appearing.

Server: 192.168.1.254

Address: 192.168 1.254#53

Non-authoritative answer:

[www.slackware.com](http://www.slackware.com/) canonical name = slackware.com. Name: slackware.com

Address: 64.57.102.34

## **TELNET:**

Telnet allows you to log in to a computer, just as if you were sitting at the terminal. Once your username and password are verified, you are given a shell prompt. From here, you can do anything requiring a text console. Compose email, read newsgroups, move files around, and so on. If you are running X and you telnet to another machine, you can run X programs on the remote computer and display them on yours.

To login to a remote machine, use this syntax:

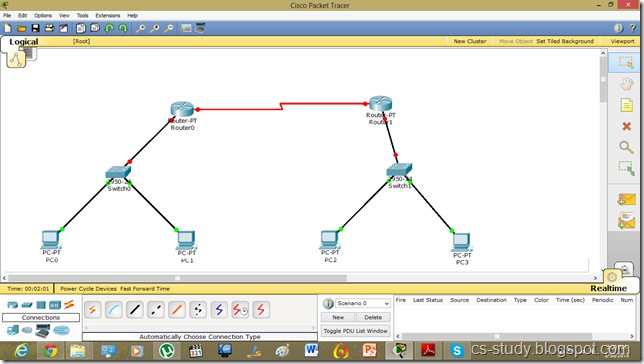
% **telnet <*hostname*>**

**Experiment No. 5**

## **AIM: CONFIGURE A NETWORK TOPOLOGY USING PACKET TRACER**

**SOFTWARE**

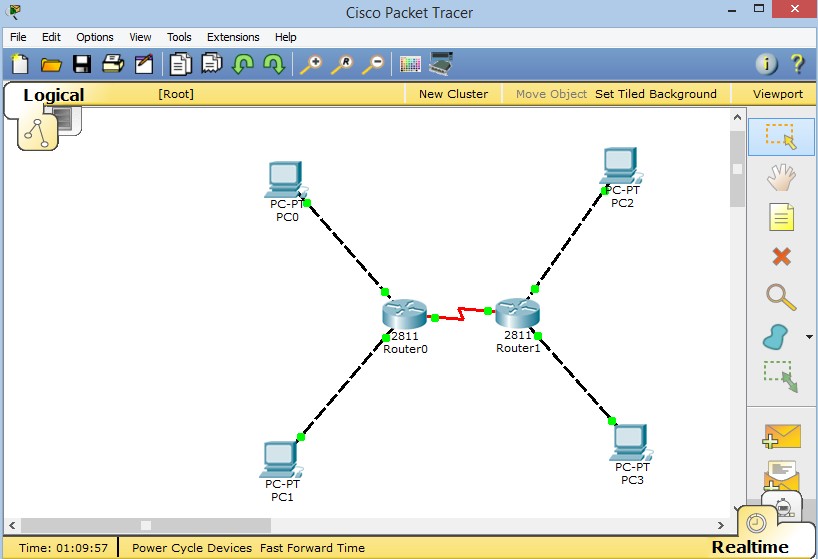
To implement this practical following network topology is required to be configured using the commands learned in previous practical. After configuring the given network a packet should be ping from any one machine to another



**Experiment No. 6**

AIM: **CONFIGURE A NETWORK USING DISTANCE VECTOR ROUTING PROTOCOL**

* 1. Configure all Routers
  2. Implement RIP protocols in Router to configure Network.



**Experiment No. 7**

**AIM: To write a program to Calculate Checksum**

**Theory:**

A checksum is a error detection method in Data Communication. It is used for errors which may have been introduced during transmission or storage. It is usually applied to an installation file after it is received from the download server.

Checksum method can only detect errors but is unable to correct the error.

In this method a checksum is calculated based on the given binary strings which is sent with the data as redundant bits. This data + checksum is received at receiver end and checksum is calculated again, if checksum is 0 it means no error in data received, else there exists some error in the received data.

**PROGRAM:**

#include<stdio.h>

#include<conio.h>

#include<math.h>

int sender(int b[10],int k) {

int checksum,sum=0,i;

printf("\n\*\*\*\*SENDER\*\*\*\*\n");

for(i=0;i<k;i++)

sum+=b[i];

printf("SUM IS: %d",sum);

checksum=~sum;

printf("\nSENDER's CHECKSUM IS:%d",checksum);

return checksum;

}

int receiver(int c[10],intk,intscheck)

{

intchecksum,sum=0,i;

printf("\n\n\*\*\*\*RECEIVER\*\*\*\*\n");

for(i=0;i<k;i++)

sum+=c[i];

printf(" RECEIVER SUM IS:%d",sum);

sum=sum+scheck;

checksum=~sum;

printf("\nRECEIVER's CHECKSUM IS:%d",checksum);

return checksum;

}

void main()

{

inta[10],i,m,scheck,rcheck;

printf("\nENTER SIZE OF THE STRING:");

scanf("%d",&m);

printf("\nENTER THE ELEMENTS OF THE ARRAY:");

for(i=0;i<m;i++)

scanf("%d",&a[i]);

scheck=sender(a,m);

rcheck=receiver(a,m,scheck);

if(rcheck==0)

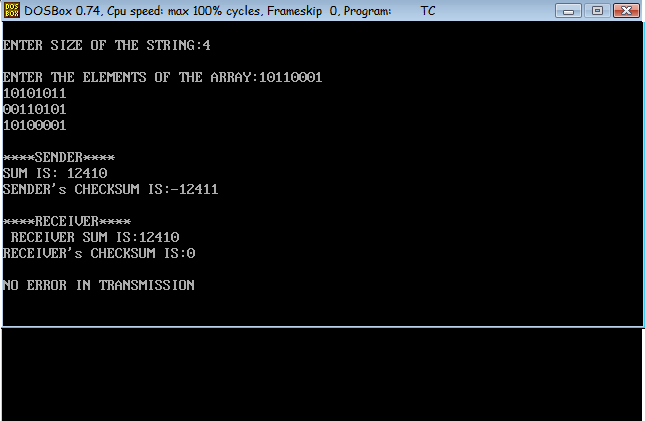
printf("\n\nNO ERROR IN TRANSMISSION\n\n");

else

printf("\n\nERROR DETECTED");

getch();

}

**OUTPUT:**

**Experiment No. 8**

**AIM: Write a Program to compute Hamming Distance between Two Data word.**

**Theory:**

Given two strings of equal length, find the [Hamming Distance](https://en.wikipedia.org/wiki/Hamming_distance) between these string.  
Where the Hamming distance between two strings of equal length is the number of positions at which the corresponding character are different.

**Program:**

#include<stdio.h>

void main()

{

int a[10]; int b[10]; int c=0; int i,n;

printf("enter the length of data code\n");

scanf("%d",&n);

printf("enter the sender side data code\n");

for(i=0;i<n;i++)

{

scanf("%d",&a[i]);

}

printf("enter the reciver side datacode\n");

for(i=0;i<n;i++)

{

scanf("%d",&b[i]);

}

for(i=0;i<n;i++)

{

if(a[i]!=b[i])

{

c++;

}

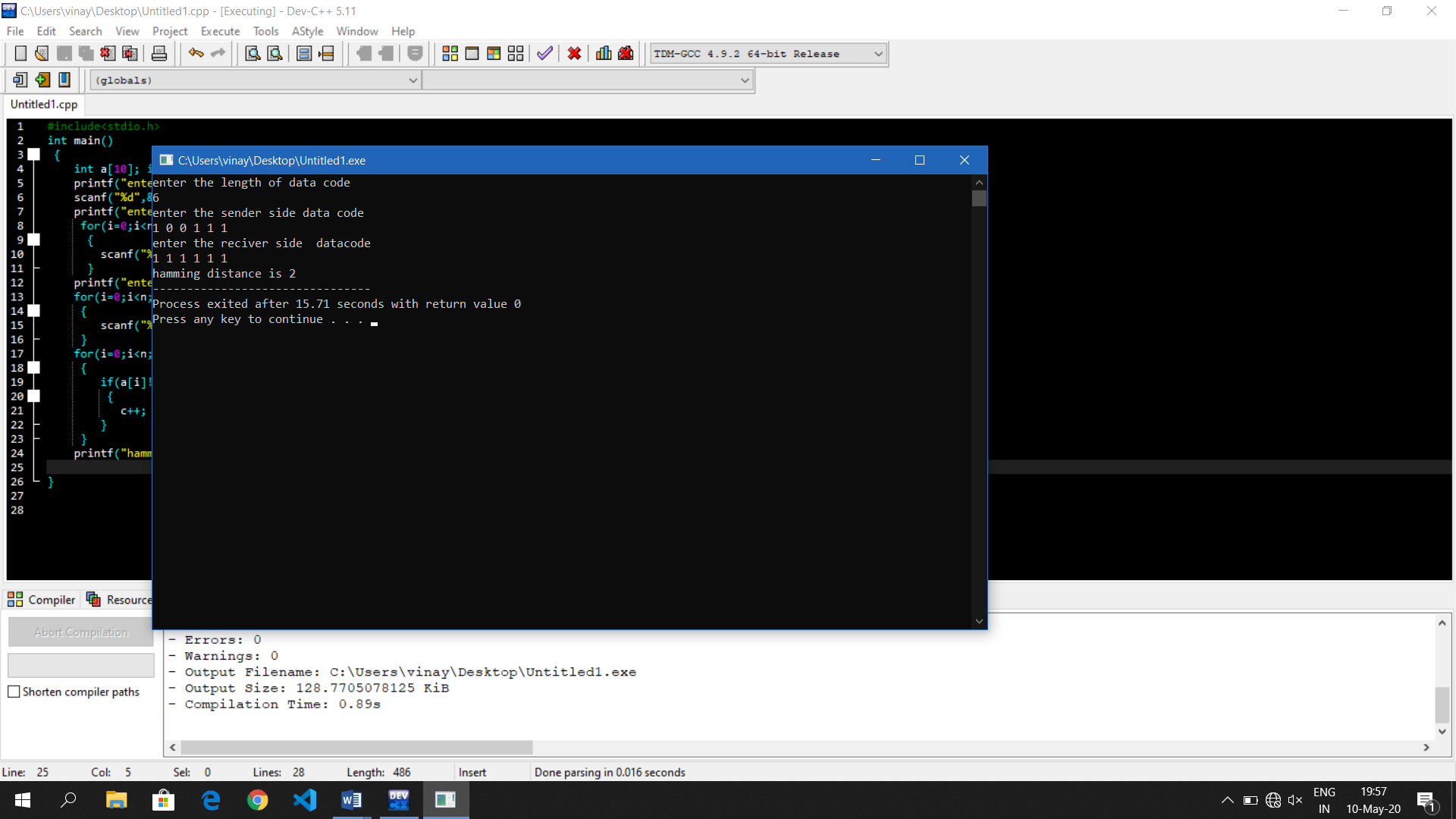
}

printf("hamming distance is %d",c);

getch();

}

**OUTPUT:**

****

## **Experiment No. 9**

**AIM: Write a program for error detecting code using CRC-CCITT (16-bits).**

It does error checking via polynomial division. In general, a bit string bn-1bn-2bn-3…b2b1b0 As bn-1Xn-1+ bn-2 Xn-2+ bn-3 Xn-3+ …b2 X2+ b1 X1+ b0

For example- 10010101110 can be expressed as X10 + X7 + X5 + X3 + X2 + X1

**PROGRAM:**

#include<stdio.h>

char m[50],g[50],r[50],q[50],temp[50];

void caltrans(int);

void crc(int);

void calram();

void shiftl();

int main()

{

int n,i=0;

char ch,flag=0;

printf("Enter the frame bits:");

while((ch=getc(stdin))!='\n'){

m[i++]=ch;

n=i;

}

for(i=0;i<16;i++){

m[n++]='0';

m[n]='\0';

}

printf("Message after appending 16 zeros:%s",m);

for(i=0;i<=16;i++) {

g[i]='0';

g[0]=g[4]=g[11]=g[16]='1';

g[17]='\0';

}

printf("\ngenerator:%s\n",g);

crc(n);

printf("\n\nquotient:%s",q);

caltrans(n);

printf("\ntransmitted frame:%s",m);

printf("\nEnter transmitted frame:");

scanf("\n%s",m);

printf("CRC checking\n");

crc(n);

printf("\n\nlast remainder:%s",r);

for(i=0;i<16;i++){

if(r[i]!='0'){

flag=1;

}

else{

continue;

}

}

if(flag==1)

printf("Error during transmission");

else

printf("\n\nReceived frme is correct");

}

void crc(int n){

int i,j;

for(i=0;i<n;i++){

temp[i]=m[i];

}

for(i=0;i<16;i++){

r[i]=m[i];

}

printf("\nintermediate remainder\n");

for(i=0;i<n-16;i++)

{

if(r[0]=='1')

{

q[i]='1';

calram();

}

else

{

q[i]='0';

shiftl();

}

r[16]=m[17+i];

r[17]='\0';

printf("\nremainder %d:%s",i+1,r);

for(j=0;j<=17;j++){

temp[j]=r[j];

}

}

q[n-16]='\0';

}

void calram(){

int i,j;

for(i=1;i<=16;i++){

r[i-1]=((int)temp[i]-48)^((int)g[i]-48)+48;

}

}

void shiftl() {

int i;

for(i=1;i<=16;i++){

r[i-1]=r[i];

}

}

void caltrans(int n){

int i,k=0;

for(i=n-16;i<n;i++)

{

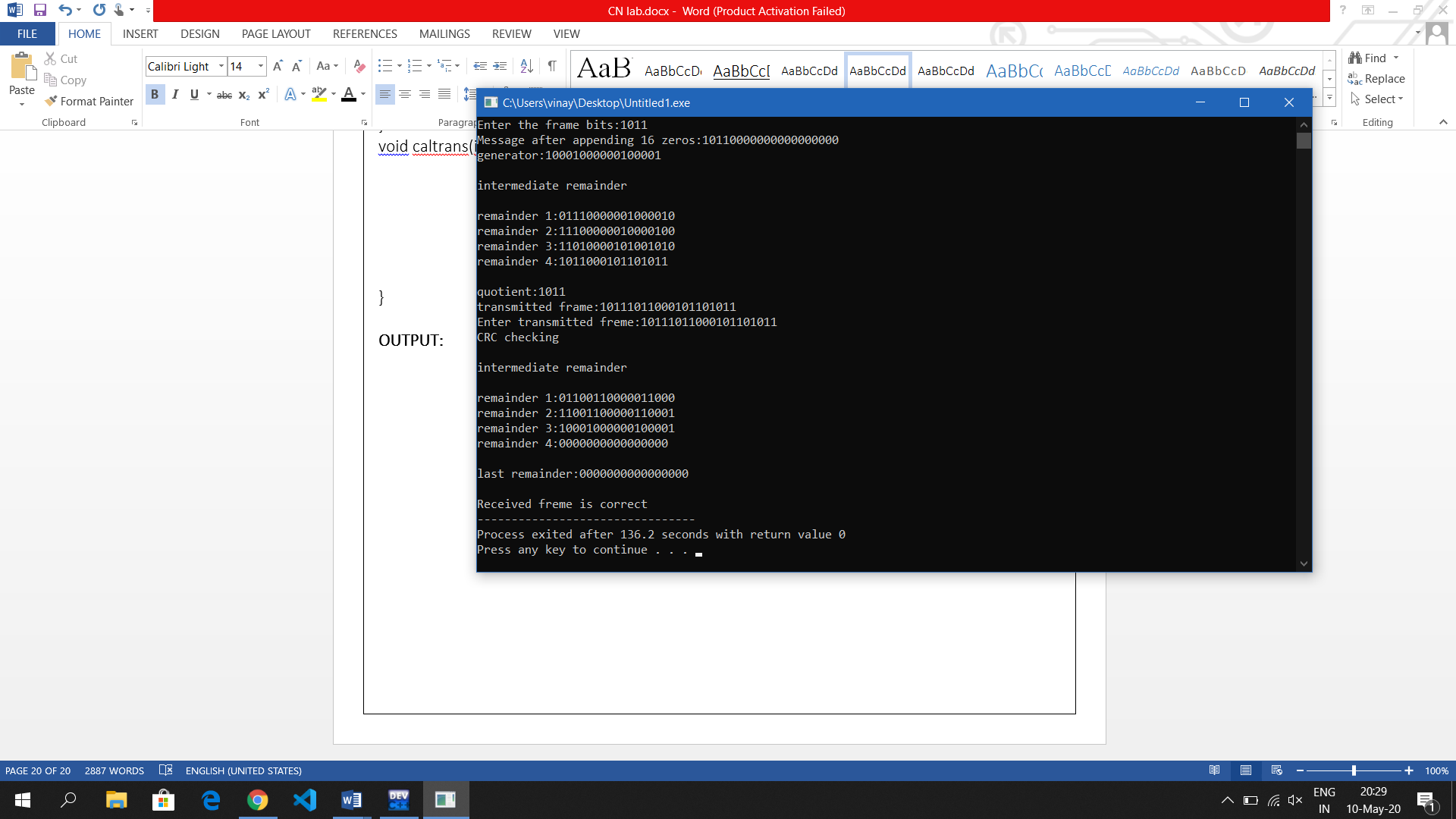
m[i]=((int)m[i]-48)^((int)r[k++]-48)+48;

m[i]='\0';

}

}

**OUTPUT:**

****

## **Experiment No. 10**

## **AIM: To write a program to Implement Hamming Code.**

Hamming code is a popular error detection and error correction method in data communication. Hamming code can only detect 2 bit error and correct a single bit error which means it is unable to correct burst errors if may occur while transmission of data.

#include<stdio.h>

int main()

{

int data[10];

int dataatrec[10],c,c1,c2,c3,i;

printf("Enter 4 bits of data one by one\n");

scanf("%d",&data[0]);

scanf("%d",&data[1]);

scanf("%d",&data[2]);

scanf("%d",&data[4]);

//Calculation of even parity

data[6]=data[0]^data[2]^data[4];

data[5]=data[0]^data[1]^data[4];

data[3]=data[0]^data[1]^data[2];

printf("\nEncoded data is\n");

for(i=0;i<7;i++) {

printf("%d",data[i]);

}

printf("\n\nEnter received data bits one by one\n");

for(i=0;i<7;i++){

scanf("%d",&dataatrec[i]);

}

c1=dataatrec[6]^dataatrec[4]^dataatrec[2]^dataatrec[0];

c2=dataatrec[5]^dataatrec[4]^dataatrec[1]^dataatrec[0];

c3=dataatrec[3]^dataatrec[2]^dataatrec[1]^dataatrec[0];

c=c3\*4+c2\*2+c1;

if(c==0){

printf("\nNo error while transmission of data\n");

}

else{

printf("\nError on position %d",c);

}

printf("\nData sent : ");

for(i=0;i<7;i++)

{

printf("%d",data[i]);

}

printf("\nData received : ");

for(i=0;i<7;i++){

printf("%d",dataatrec[i]);

}

printf("\nCorrect message is\n");

//if errorneous bit is 0 we complement it else vice versa

if(dataatrec[7-c]==0){

dataatrec[7-c]=1;

}

else{

dataatrec[7-c]=0;

}

for(i=0;i<7;i++){

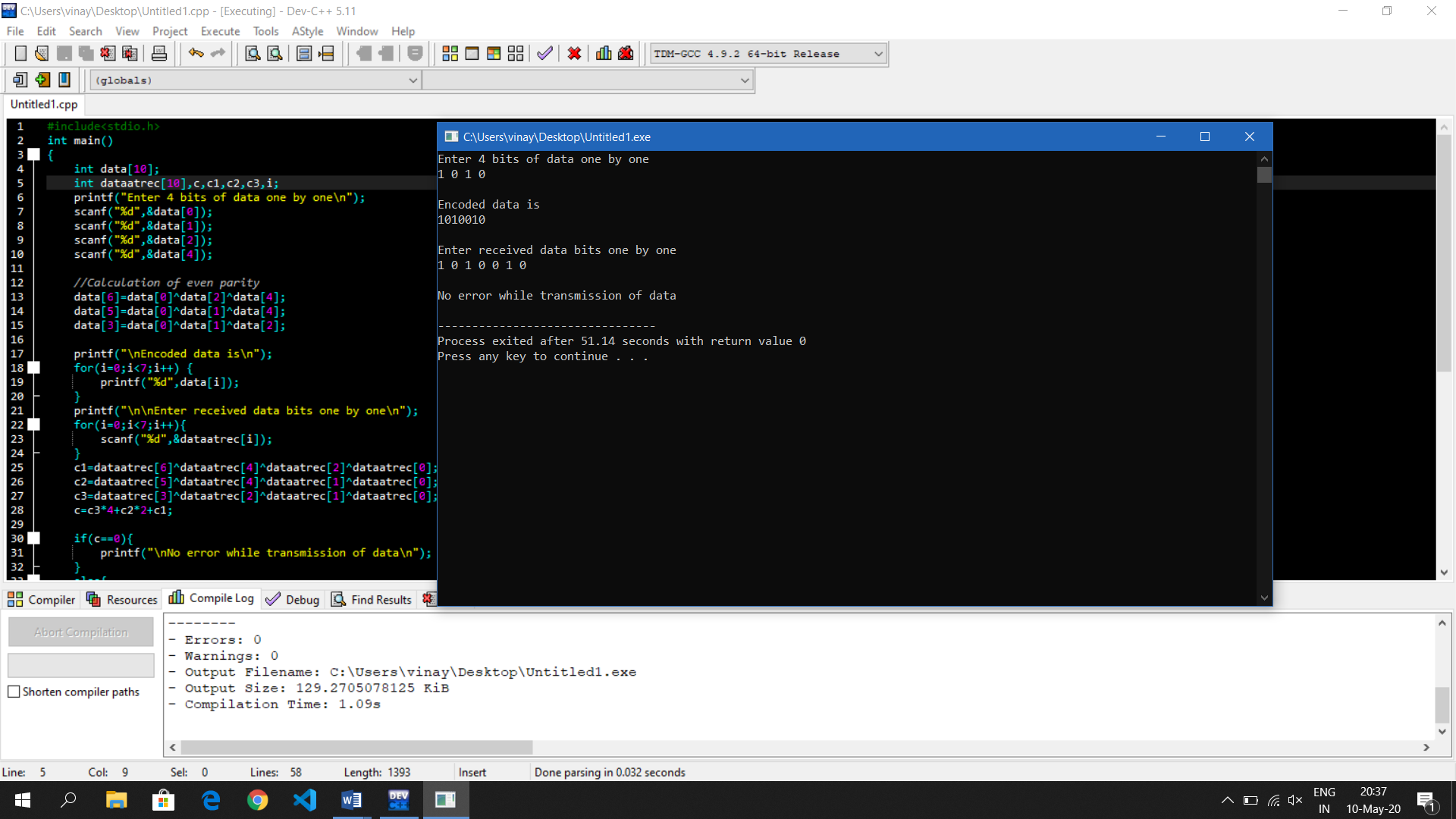
printf("%d",dataatrec[i]);

}

}

}

**OUTPUT:**

****

## **Experiment No. 11**

**AIM: Write a program for simple RSA algorithm to encrypt and decrypt the data.**

Cryptography has a long and colorful history. The message to be encrypted, known as the plaintext, are transformed by a function that is parameterized by a key. The output of the encryption process, known as the ciphertext, is then transmitted, often by messenger or radio. The enemy, or intruder, hears and accurately copies down the complete ciphertext. However, unlike the intended recipient, he does not know the decryption key and so cannot decrypt the ciphertext easily. The art of breaking ciphers is called **cryptanalysis** the art of devising ciphers

(cryptography) and breaking them

(cryptanalysis) is collectively known as cryptology.

There are several ways of classifying cryptographic algorithms. They are generally categorized based on the number of keys that are employed for encryption and decryption, and further defined by their application and use. The three types of algorithms are as follows:

1. Secret Key Cryptography (SKC): Uses a single key for both encryption and decryption. It is also known as symmetric cryptography.
2. Public Key Cryptography (PKC): Uses one key for encryption and another for decryption. It is also known as asymmetric cryptography.
3. Hash Functions: Uses a mathematical transformation to irreversibly "encrypt" Information.

Public-key cryptography has been said to be the most significant new development in cryptography. Modern PKC was first described publicly by Stanford University professor Martin Hellman and graduate student Whitfield Diffie in 1976. Their paper described a two-key crypto system in which two parties could engage in a secure communication over a non-secure communications channel without having to share a secret key.

Generic PKC employs two keys that are mathematically related although knowledge of one key does not allow someone to easily determine the other key. One key is used to encrypt the plaintext and the other key is used to decrypt the ciphertext. The important point here is that it does not matter which key is applied first, but that both keys are required for the process to work. Because pair of keys is required, this approach is also called asymmetric cryptography.

In PKC, one of the keys is designated the public key and may be advertised as widely as the owner wants. The other key is designated the private key and is never revealed to another party. It is straight forward to send messages under this scheme.

**PROGRAM:**

#include <stdio.h>

#include <string.h>

#include <conio.h> #include <math.h>

int mult(unsigned int x, unsigned int y, unsigned int n)

{

unsigned long int k=1;

int j;

for (j=1; j<=y; j++) {

k = (k \* x) % n;

}

return (unsigned int) k;

}

int main ()

{

char msg[100];

unsigned int pt[100], ct[100], n, d, e, p, q, i; printf("Enter message : "); gets(msg);

//strcpy(pt, msg);

for(i=0;i<strlen(msg);i++){

pt[i]=msg[i];

}

n = 253;

d = 17;

e = 13;

printf("\nCT = ");

for(i=0; i<strlen(msg); i++){

ct[i] = mult(pt[i], e,n);

}

for(i=0; i<strlen(msg); i++){

printf("%d ", ct[i]);

}

printf("\nPT = ");

for(i=0; i<strlen(msg); i++){

printf("%c", pt[i]);

}

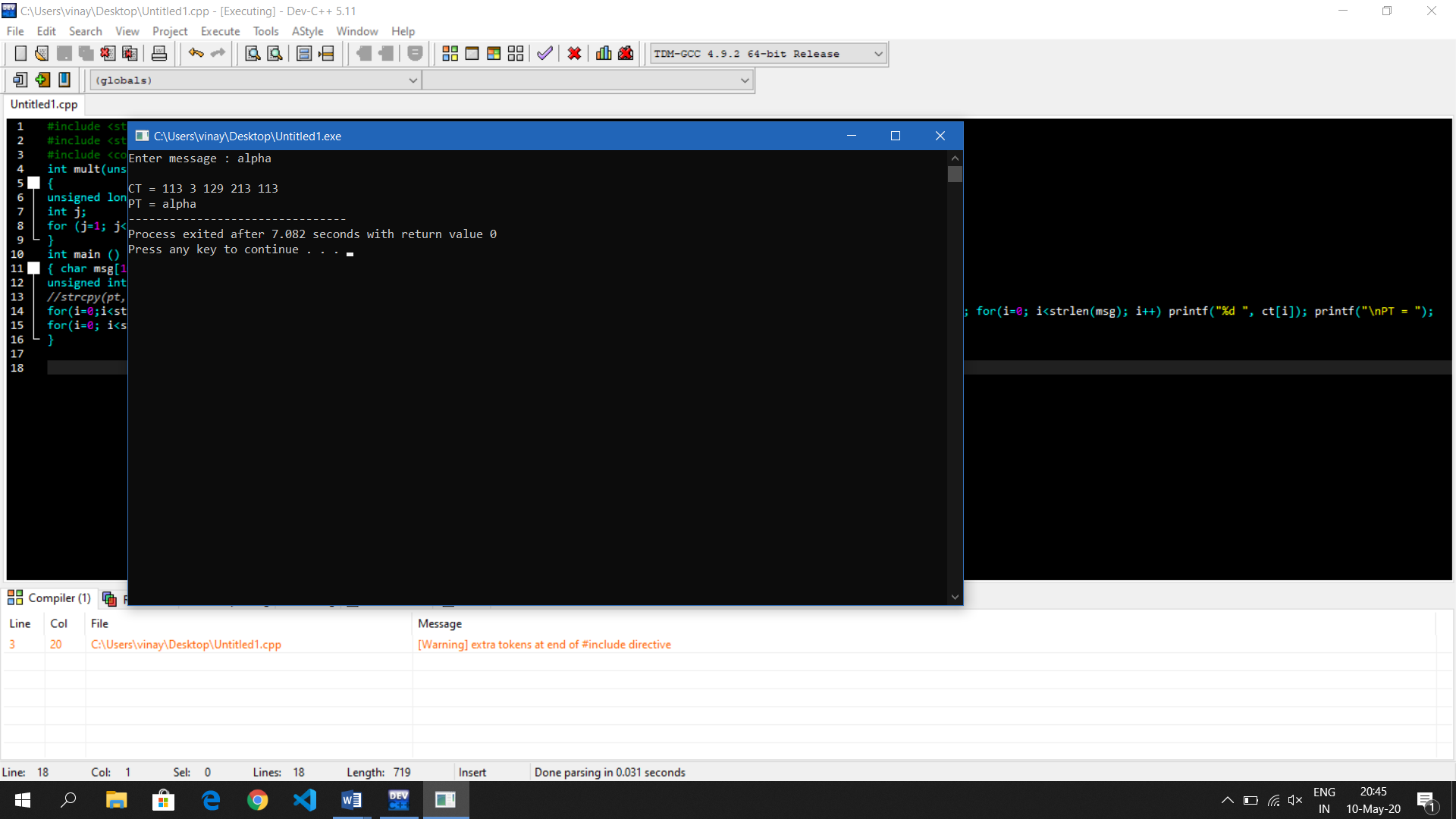
for(i=0; i<strlen(msg); i++){

pt[i] = mult(ct[i], d,n) ;

}

}

**OUTPUT:**

****

## **Experiment No. 12**

**AIM: WAP to Calculate the Link Utilization for Stop and Wait ARQ Protocol**

**Link utilization** is simply the average traffic over a particular **link** expressed. as a percentage of the total **link** capacity. **Link** efficiency is a less. commonly used term that is **defined** as the ratio of the time taken to. transmit a frame (or frames) of data to the total time it takes to transmit.

**PROGRAM:**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<math.h>

#define v 2.0e8

int main(void)

{

char instring[80];

double d,r,pr,p,a,u,u1;

long int l;

printf("....................");

printf("\n Program to Compute Utiliation of Stop and Wait Protocol\n");

printf("\n.....................\n");

printf("\n Length of the link in m:");

scanf("%s",instring);

d=atof(instring);

printf("\n Data rate of the link in bits per sec");

scanf("%s",instring);

r=atof(instring);

printf("\n Size of frame in byte:");

scanf("%s",instring);

l=8\*atoi(instring);

printf("\n pr[biterror](0.0to1.0):");

scanf("%s",instring);

pr=atof(instring);

p=1-pow((1-pr),(double)l);

a=((double)r\*d)/((double)v\*l);

u=(1-p)/(1+(2\*a));

u1=(1)/(1+(2\*a));

printf("\n...................\n");

printf("\n.....Protocol is Stop and Wait....\n");

printf("......a=%f.........\n",a);

printf("......pr[frameloss]=%f....\n",p);

printf(".....Utilisation Factor with Error=%f....\n",100.0\*u);

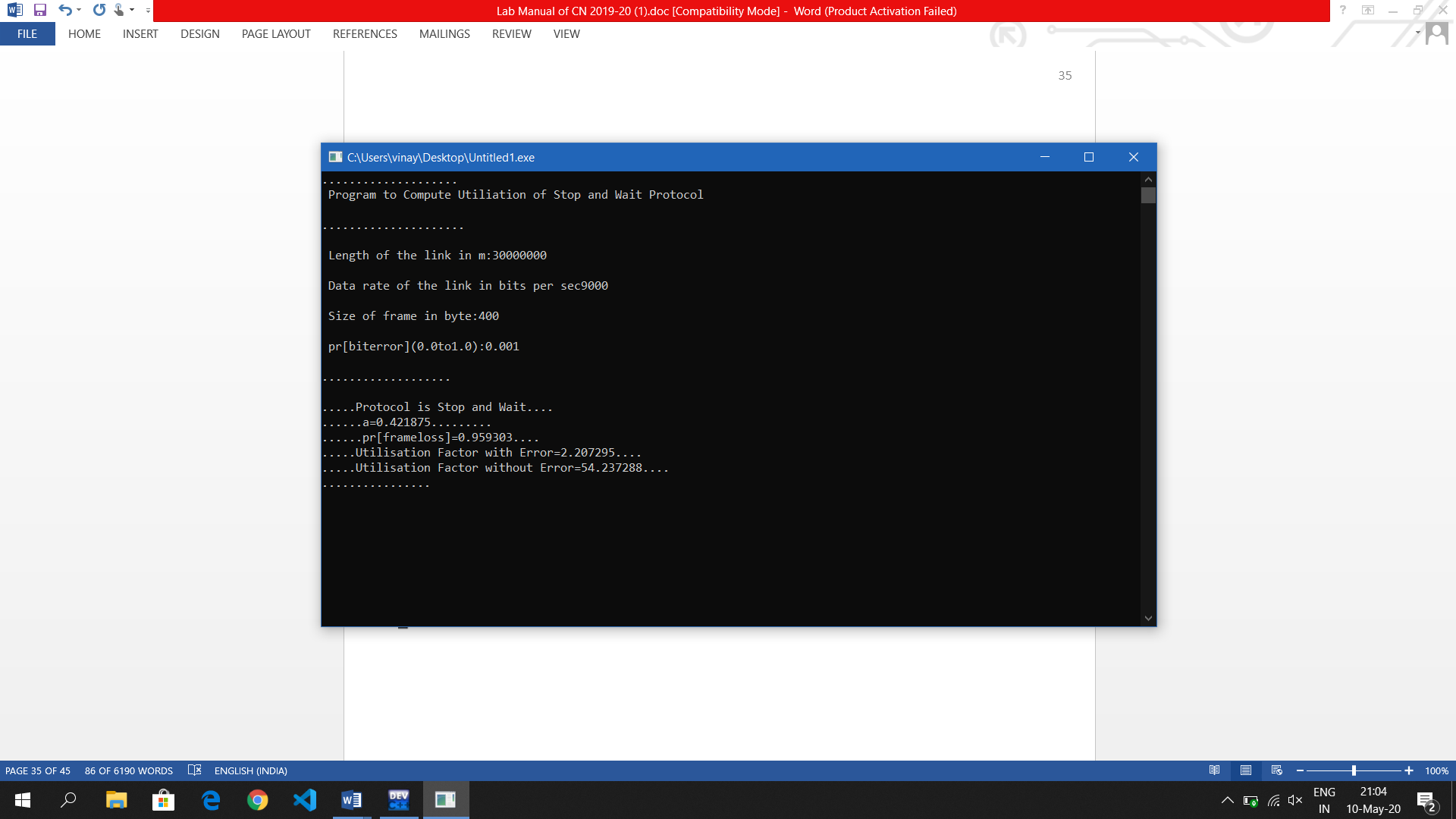
printf(".....Utilisation Factor without Error=%f....\n",100.0\*u1);

printf("................\n");

getch();

}

**OUTPUT:**

****

**Experiment No. 13**

**AIM: Write a c program for bit stuffing used in HDLC**

Bit stuffing is the insertion of one or more bits into a transmission unit as a way to provide signaling information to a receiver. The receiver knows how to detect and remove or disregard the stuffed bits.

In another example of bit stuffing, a standard HDLC packet begins and ends with 01111110. To make sure this sequence doesn't appear again before the end of the packet, a 0 is inserted after every five consecutive 1s.

Bit stuffing is defined by some to include bit padding, which is the addition of bits to a tranmission to make the transmission unit conform to a standard size, but is distinct from bit robbing, a type of in-band signaling.

**PROGRAM:**

#include<stdio.h>

#include<conio.h>

#include<string.h>

void main()

{ int ip[20],op[30],i=j=0,count=0; printf(“enter the i/p bit”); get(ip);

for(i=0;i<strlen(ip);i++) op[0]=ip[0]; op[j++]=ip[i];

if(ip[i]==’1’) count++; else count=0; if(count==5) {

op[j++]=’0’;

count=0;

}

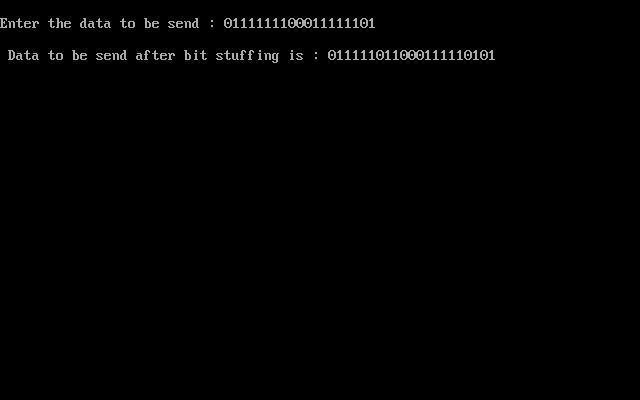
}

op[j++]=’\0’;

printf(“bits stuffed is %s”,op); getch();

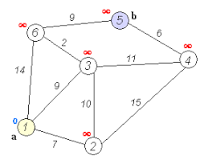
}

**OUTPUT:**



**Experiment No. 14**

**AIM: To write a program for Routing algorithm to find Shortest Path.**

**[](https://www.google.com/search?q=dijkstra's+algorithm+definition&tbm=isch&source=iu&ictx=1&fir=8B5aSrXgivKURM:,ixmtGM5qkdJ4uM,_&vet=1&usg=AI4_-kQlYj_5J2ZrgBO9P8dTRBUz1KLZcw&sa=X&ved=2ahUKEwj5gefk9ubhAhWz8XMBHfqwCtQQ9QEwAHoECA4QBg#imgrc=8B5aSrXgivKURM:)Dijkstra's algorithm** (or **Dijkstra's** Shortest Path First**algorithm**, SPF **algorithm**) is an **algorithm** for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. ... For a given source node in the graph, the **algorithm** finds the shortest path between that node and every other.

Dijkstra's algorithm to find the shortest path between *a* and *b*. It picks the unvisited vertex with the lowest distance, calculates the distance through it to each unvisited neighbor, and updates the neighbor's distance if smaller. Mark visited (set to red) when done with neighbors

**PROGRAM:**

#include<stdio.h>

#include<conio.h>

#define INFINITY 9999

#define MAX 10

void dijkstra(int G[MAX][MAX],int n,intstartnode);

int main()

{

int G[MAX][MAX],i,j,n,u;

printf("Enter no. of vertices:");

scanf("%d",&n);

printf("\nEnter the adjacency matrix:\n");

for(i=0;i<n;i++)

for(j=0;j<n;j++)

scanf("%d",&G[i][j]);

printf("\nEnter the starting node:");

scanf("%d",&u);

dijkstra(G,n,u);

return 0;

}

void dijkstra(int G[MAX][MAX],int n,intstartnode)

{

int cost[MAX][MAX],distance[MAX],pred[MAX];

int visited[MAX],count,mindistance,nextnode,i,j;

//pred[] stores the predecessor of each node

//count gives the number of nodes seen so far

//create the cost matrix

for(i=0;i<n;i++)

for(j=0;j<n;j++)

if(G[i][j]==0)

cost[i][j]=INFINITY;

else

cost[i][j]=G[i][j];

//initialize pred[],distance[] and visited[]

for(i=0;i<n;i++)

{

distance[i]=cost[startnode][i];

pred[i]=startnode;

visited[i]=0;

}

distance[startnode]=0;

visited[startnode]=1;

count=1;

while(count<n-1)

{

mindistance=INFINITY;

//nextnode gives the node at minimum distance

for(i=0;i<n;i++)

if(distance[i]<mindistance&&!visited[i])

{

mindistance=distance[i];

nextnode=i;

}

//check if a better path exists through nextnode

visited[nextnode]=1;

for(i=0;i<n;i++)

if(!visited[i])

if(mindistance+cost[nextnode][i]<distance[i])

{

distance[i]=mindistance+cost[nextnode][i];

pred[i]=nextnode;

}

count++;

}

//print the path and distance of each node

for(i=0;i<n;i++)

if(i!=startnode)

{

printf("\nDistance of node%d=%d",i,distance[i]);

printf("\nPath=%d",i);

j=i;

do

{

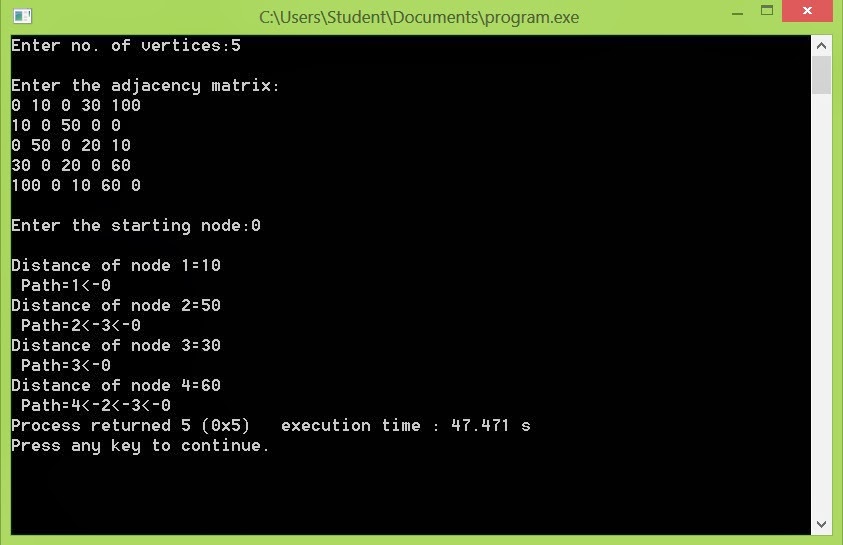
j=pred[j];

printf("<-%d",j);

}while(j!=startnode);

}

}

**OUTPUT:**

**Experiment No. 15**

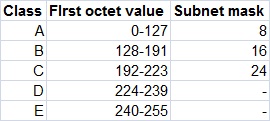
**AIM: Write A Program to find Class of IP Address and Network Address for Given**

**IP Address**

TCP/IP defines five classes of IP addresses: class A, B, C, D, and E. Each class has a range of valid IP addresses. The value of the first octet determines the class. IP addresses from the first three classes (A, B and C) can be used for host addresses. The other two classes are used for other purposes (class D for multicast and class E for experimental purposes).

The system of IP address classes was developed for the purpose of Internet IP addresses assignment. The classes created were based on the network size. For example, for the small number of networks with a very large number of hosts, the Class A was created. The Class C was created for the numerous networks with the small number of hosts.

Classes of IP addresses are:

[](https://study-ccna.com/wp-content/images/classes_of_ip_addresses.jpg)

For the IP addresses from Class A, the first 8 bits (the first decimal number) represent the network part, while the remaining 24 bits represent the host part. For Class B, the first 16 bits (the first two numbers) represent the network part, while the remaining 16 bits represent the host part. For Class C, the first 24 bits represent the network part, while the remaining 8 bits represent the host part.

**PROGRAM:**

#include<stdio.h>

int main()

{

int a,b,c,d;

printf("enter the first octet");

scanf("%d",&a); printf("\n");

printf("enter the second octet");

scanf("%d",&b); printf("\n");

printf("enter the third octet");

scanf("%d",&c); printf("\n");

printf("enter the fourth octet");

scanf("%d",&d); printf("\n"); if((a<=127))

{

printf("enterdip address belong to class A\n");

printf("network address of given ip address is %d.0.0.0",a);

}

else if((a>127)&&(a<=191))

{

printf("entered ip address belong to class B\n");

printf("network address of given ip address is %d.%d.0.0",a,b);

}

else if((a>191)&&(a<=223))

{

printf("entered ipadress belong to class C\n");

printf("network address of given ip address is %d.%d.%d.0",a,b,c);

}

else if((a>223)&&(a<=241))

{

printf("entered ip address belong to class D\n");

printf("network address of given ip address is useed for reserved");

}

else if((a>241)&&(a<=255))

{

printf("entered ipadress belong to E\n");

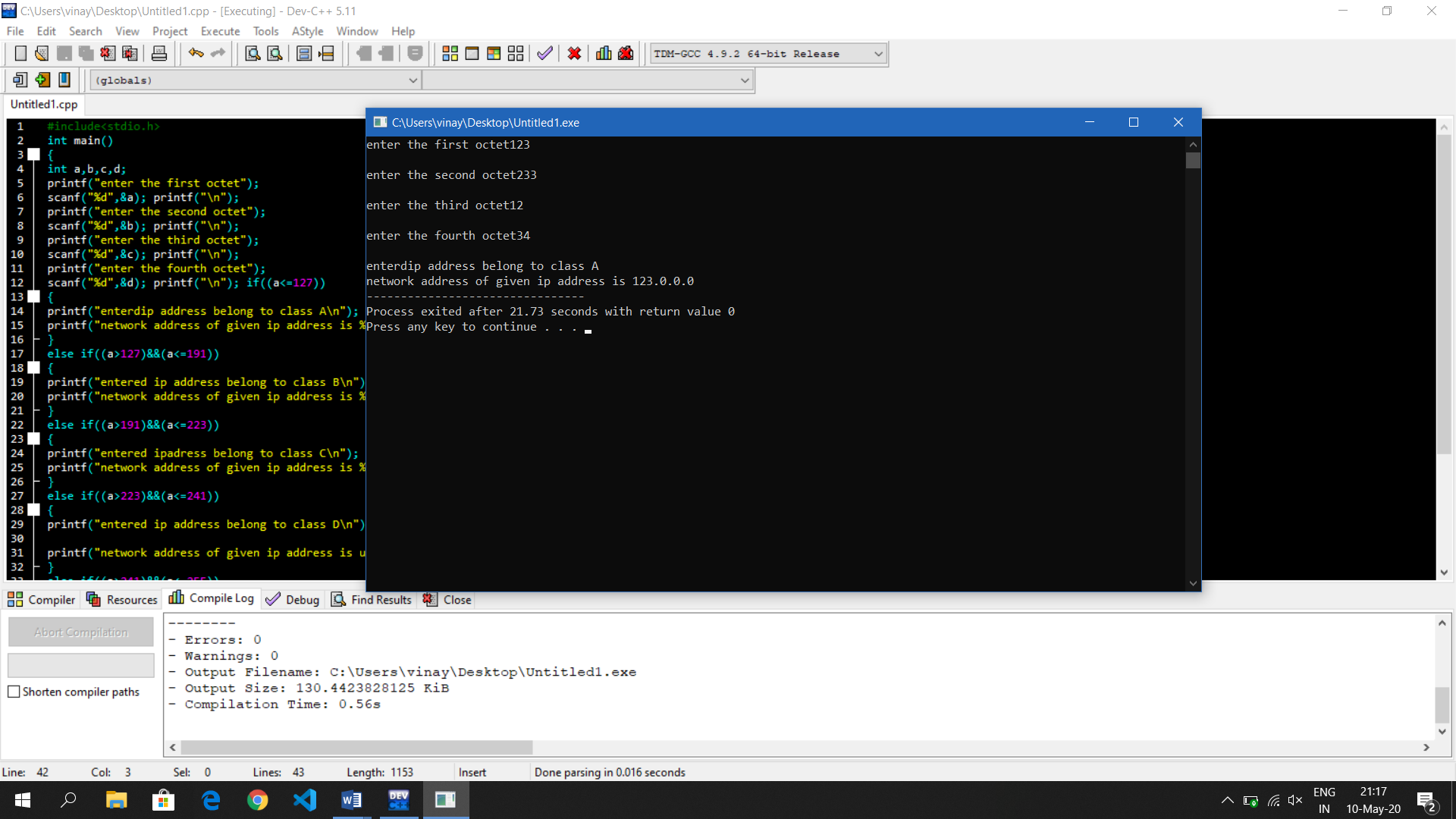
printf("network address of given ip address is used for multicasting");

} else {

printf("please enter the valid ip address\n");

}

}

**OUTPUT:**