

**DATA COMMUNICATION AND COMPUTER NETWORKS [CSE302]**

**AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY**

(Computer Science and Engineering)

**SEMESTER-6**

**SUBMITTED TO- SUBMITTED BY-**

MR. KUNAL GUPTA

**I N D E X**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Categ-ory | Code | Exp  No. | Name of Experiment | Date of allotment of exp. | Date of evaluation | Max. Marks | Marks obtained | Signatu-re of Faculty |
| Mandatory | LR (10) | 1 | Installation of Linux | 07/01/19 | 14/01/19 | 1 |  |  |
| 2 | To implement basic Linux Commands | 14/01/19 | 21/01/19 | 1 |  |  |
| 3 | (a)To establish straight configuration for LAN.  (b) To establish rollover configuration for LAN.  (c) To establish crossover configuration for LAN. | 21/01/19 | 28/01/19 | 1 |  |  |
| 4 | WAP to generate Hamming code and check if the received codeword is correct. | 28/01/19 | 04/02/19 | 1 |  |  |
| 5 | WAP which manipulates pure and slotted ALOHA based on user inputs. | 04/02/19 | 11/02/19 | 1 |  |  |
| 6 | WAP to translate between dotted decimal form to 32 bit binary | 11/02/19 | 18/02/19 | 1 |  |  |
| 7 | To enter the IP address in binary or decimal form, and find its class | 18/02/19 | 25/02/19 | 1 |  |  |
| 8 | WAP to implement stop and wait ARQ. | 25/02/19 | 11/03/19 | 1 |  |  |
| 9 | WAP to find shortest path using Dijkstra’s algorithm. | 11/03/19 | 18/03/19 | 1 |  |  |
|  |  | 10 | WAP to find shortest path using Bellman Ford’s algorithm. | 18/03/19 | 25/03/19 | 1 |  |  |
|  |  |  | Design based | 25/03/19 | 01/04/19 |  |  |  |

**EXPERIMENT NO 1**

**Objective: Installation of Linux**.

**Procedure:** The steps to install Linux or Ubuntu on a computer are explained below.

**Steps:**

1. **Download the Ubuntu ISO file:** An ISO file is a CD image file that will need to [be burned](http://www.wikihow.com/Burn-ISO-Files-to-DVD) before you can use it. There are two options available from the Ubuntu website:

* 14.04 LTS has continuous updates and provides technical support. It is scheduled to be supported until April 2019. This option will give the most compatibility with any existing hardware.
* Ubuntu builds (not yet released) 14.10, 15.04 and 15.10 will come with limited support. They will have the newest features, though they may not work with all hardware. These releases are geared more towards experienced Linux users.
* For a Windows 9 PC or a PC with UEFI firmware, download the 64-bit version of Ubuntu. Older machines should download the 32-bit version.

1. **Burn the ISO file:** Open up your burning program of choice. There are free and paid programs available that can burn an ISO to a CD or DVD.

* Windows 7, 8, and Mac OS X [can all burn ISO files to a disc](http://www.wikihow.com/Burn-ISO-Files-to-DVD) without having to download a separate program.

1. **Boot from the disc:** Once you have finished burning the disc, restart your computer and choose to boot from the disc. You may have to change your boot preferences by hitting the Setup key while your computer is restarting. This is typically F12, F2, or Del.
2. **Try Ubuntu before installing:** Once you boot from the disc, you will be given the option to try Ubuntu without installing it. The operating system will run from the disc, and you will have a chance to explore the layout of the operating system.
3. **Install Ubuntu:** Your computer will need at least 4.5 GB of free space. You will want more than this if you want to install programs and create files. If you are installing on a laptop, make sure that it is connected to a power source, as installing can drain the battery faster than normal.

* Check the “Download updates automatically” box, as well as the “Install this third-party software” box. The third-party software will allow you to play MP3 files as well as watch Flash video (such as YouTube).

1. **Set up the wireless connection:** If your computer is not connected to the internet via Ethernet, you can configure your wireless connection in the next step.
2. **Choose what to do with your existing operating system:** If you have Windows installed on your system, you will be given a couple options on how you’d like to install Ubuntu. You can either install it alongside your previous Windows installation, or you can replace your Windows installation with Ubuntu.

* If you install it alongside your old version of Windows, you will be given the option to choose your operating system each time you reboot your computer. Your Windows files and programs will remain untouched.
* If you replace your installation of Windows with Ubuntu all of your Windows files, documents, and programs will be deleted.

1. **Set your partition size:** If you are installing Ubuntu alongside Windows, you can use the slider to adjust how much space you would like to designate for Ubuntu. Remember that Ubuntu will take up about 4.5 GB when it is installed, so be sure to leave some extra space for programs and files.

* Once you are satisfied with your settings, click Install Now.

1. **Choose your location.** If you are connected to the internet, this should be done automatically. Verify that the time zone displayed is correct, and then click the Continue button.
2. **Set your keyboard layout.** You can choose from a list of options, or click the Detect Keyboard Layout button to have Ubuntu automatically pick the correct option.
3. **Enter your login information.** Enter your name, the name of the computer (which will be displayed on the network), choose a username, and come up with a password. You can choose to have Ubuntu automatically log you in, or require your username and password when it starts.
4. **Wait for the installation process to complete.** Once you choose your login info, the installation will begin. During setup, various tips for [using Ubuntu](http://www.wikihow.com/Get-Familiar-With-Ubuntu-Commands) will be displayed on the screen. Once it is finished, you will be prompted to restart the computer and Ubuntu will load.



**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT NO 2**

**Objective: To implement basic Linux commands**

**Equipment/Software Used:**

|  |  |  |
| --- | --- | --- |
| S.no. | Hardware | Software |
| 1. | I7 processor | Os(windows 8) |
| 2. | 8 gb ram | Microsoft word |
| 3. | Keyboard | Cmd |
| 4. | Mouse |  |
| 5. | Monitor |  |
| 6. | Printer |  |

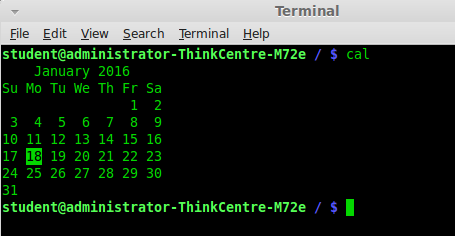
**COMMANDS :-**

* **Command name:** cal

**Syntax 1:** $cal

**Description:**To show the calendar of the current month.

**Output:** The output is as follows:

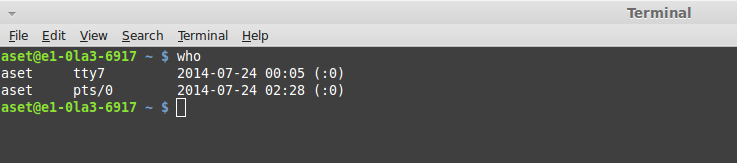


* **Command name:** who

**Syntax:** $who

**Description:**To display the details of the access of the system at every point of time.

**Output:** The output is as follows:



* **Command name:** whoami

**Syntax:** $whoami

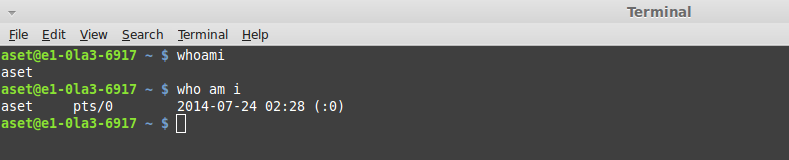
**Description:**To display the login name of the current user.

**Command name:** who am i

**Syntax:** $who am i

**Description:**Displays the user’s login name, date and the current time.

**Output:** The output is as follows:

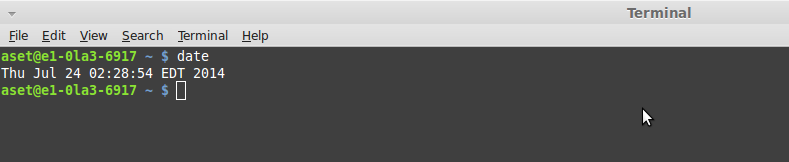


* **Command name:** date

**Syntax:** $date

**Description:**Displays current date on screen.

**Output:** The output is as follows:

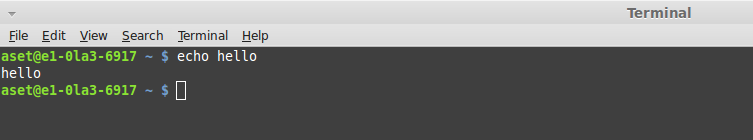


* **Command name:** echo

**Syntax:** $echo Hello

**Description:**Prints “Hello” on the screen.

**Output:** The output is as follows:



* **Command name:** cat

**Syntax 1:** $cat >ss

**Description:**Creates a file named “ss”.

**Syntax 2:** $cat ss

**Description:**Displays the content in the file “ss”.

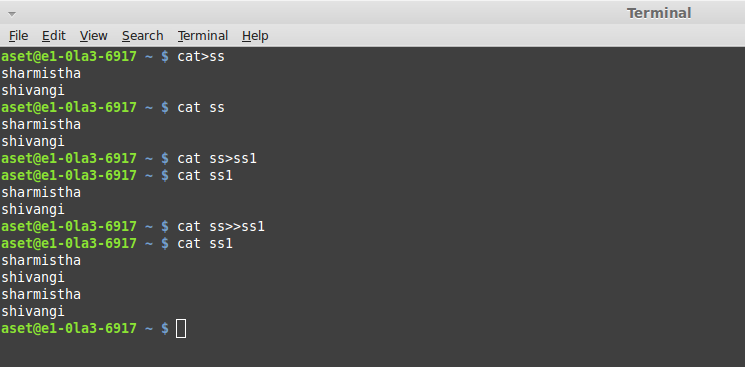
**Syntax 3:** $cat ss> ss1

**Description:**Copies the content of file “ss” to file ”ss1”.If file “ss1” is not present in the system, then cal command will first create file “ss1” and then copy the contents into it.

**Syntax 4:** $cat ss>> ss1

**Description:**To append or merge the content of both the files in file “ss1”.

**Output:** The output is as follows:

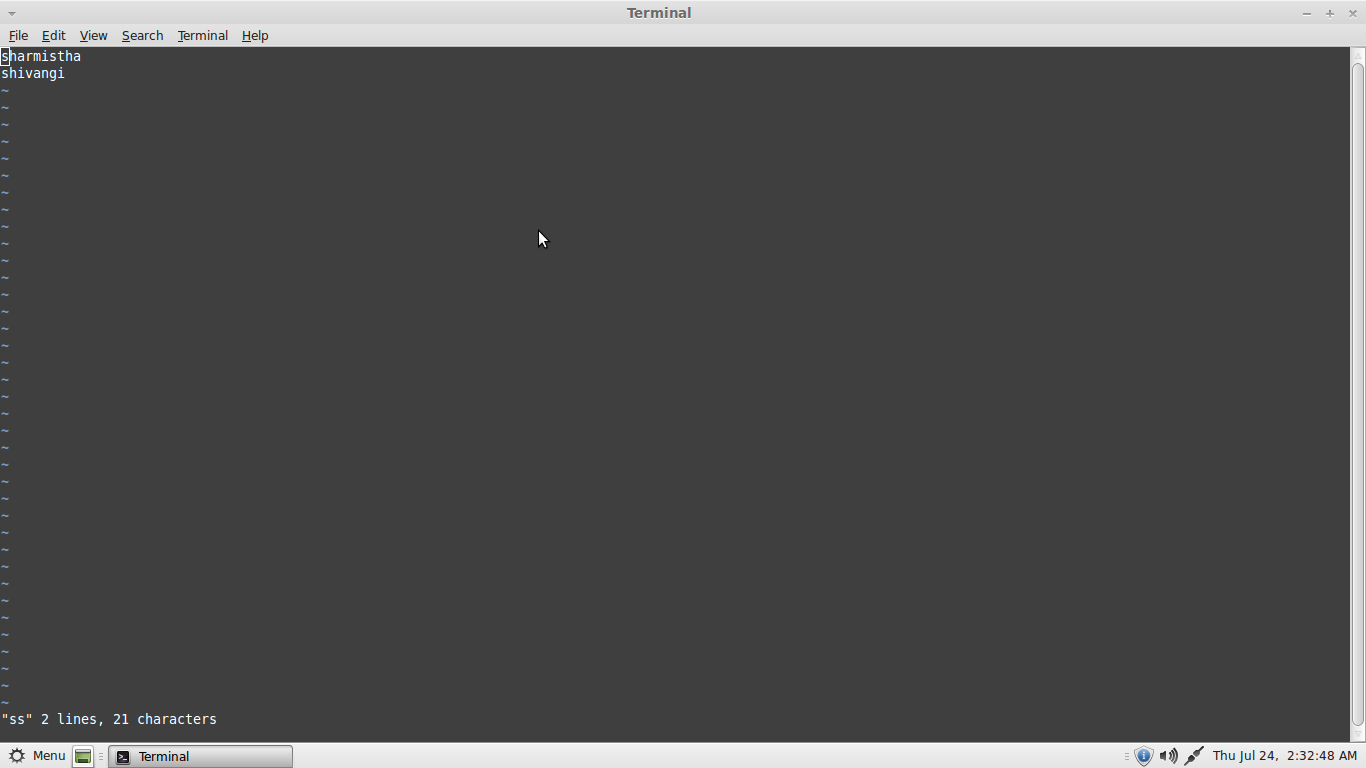


* **Command name:** vi

**Syntax:** $viss

**Description:**Does not allow the user to write anything into file “abc” i.e. switches to command-line mode.

**Output:** The output is as follows:

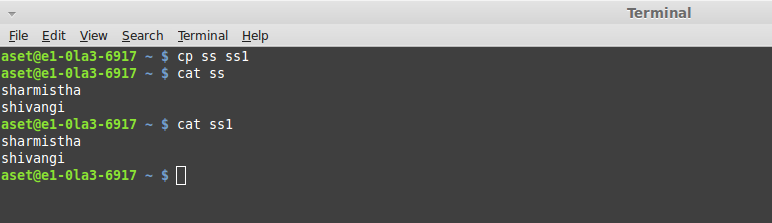


* **Command name:** cp

**Syntax:** $cpss ss1

**Description:**Directly copies the content of file “ss” to file “ss1”.

**Output:** The output is as follows:



* **Command name:** rm

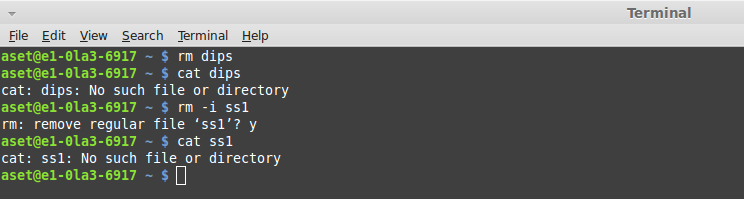
**Syntax 1:** $rm dips

**Description:** To delete the file “dips”.

**Syntax 2:** $rm –i ss1

**Description:** To make command “rm” user-interactive, i.e. asks whether to permanently delete the file “ss1” or not.

**Output:** The output is as follows:

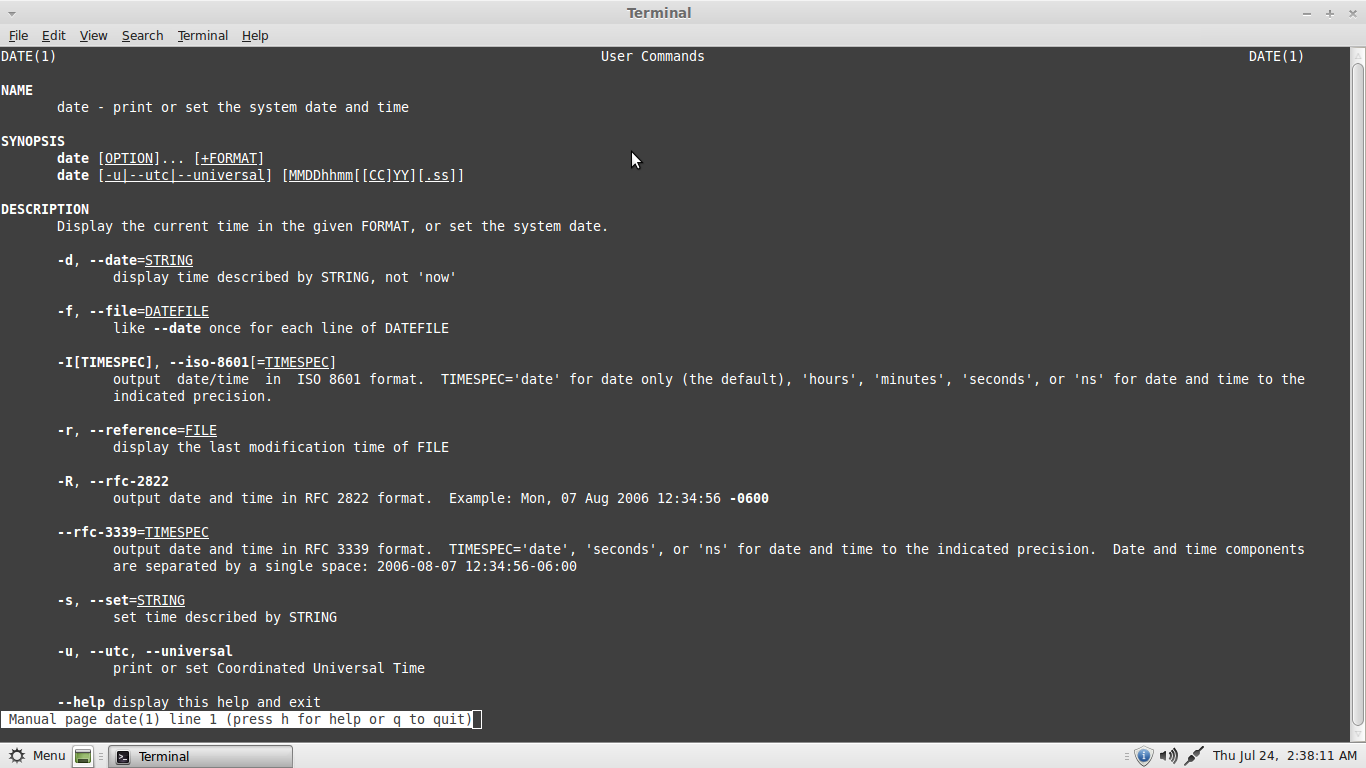


* **Command name:** man

**Syntax:** $man date

**Description:**Displays a manual of the command “date” and shows how “date” can be used in various other ways.

**Output:** The output is as follows:

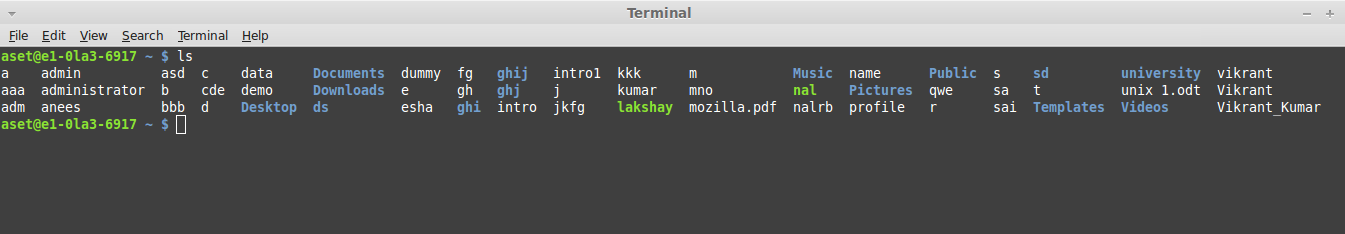


* **Command name:** ls

**Syntax:** $ls

**Description:**Displays the list of files stored in the system.

**Output:** The output is as follows:



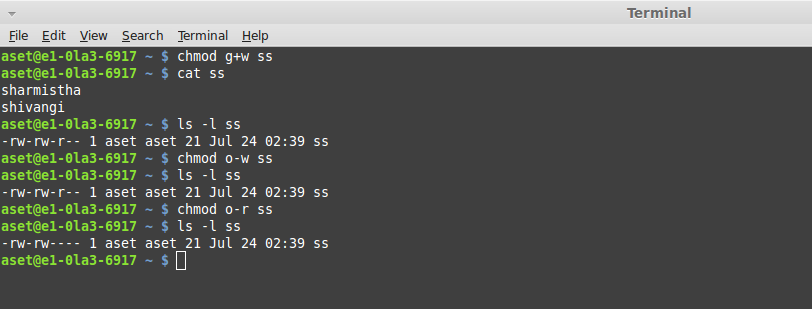
* **Command name:** chmod

**Syntax:** $chmod<category><operation><permission><filename>

$chmodg+wss

**Description:** Used to change the mode and/or permission of the file

**Output:** The output is as follows:

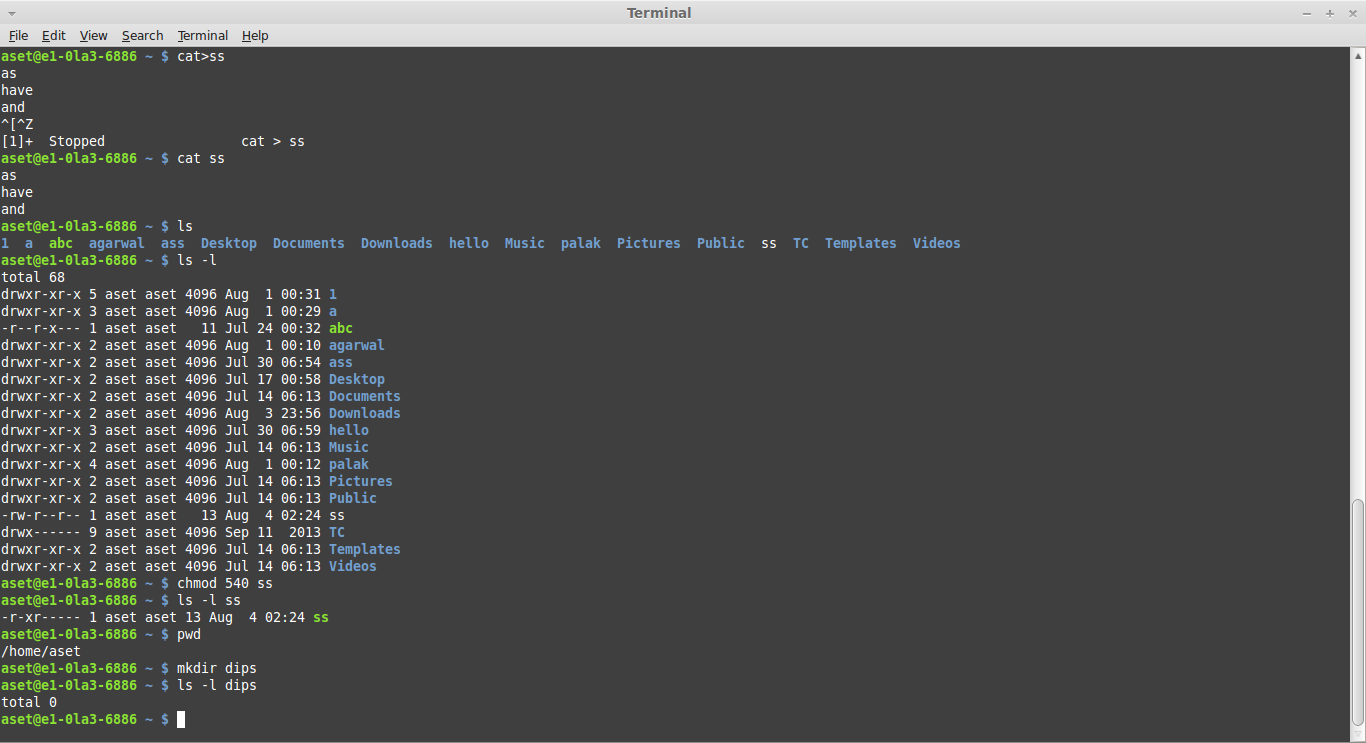


* **Command name:** mkdir<directory name>

**Syntax:** $mkdir dips

**Description:** This command is used for creating a directory.

**Output:** The output is as follows:

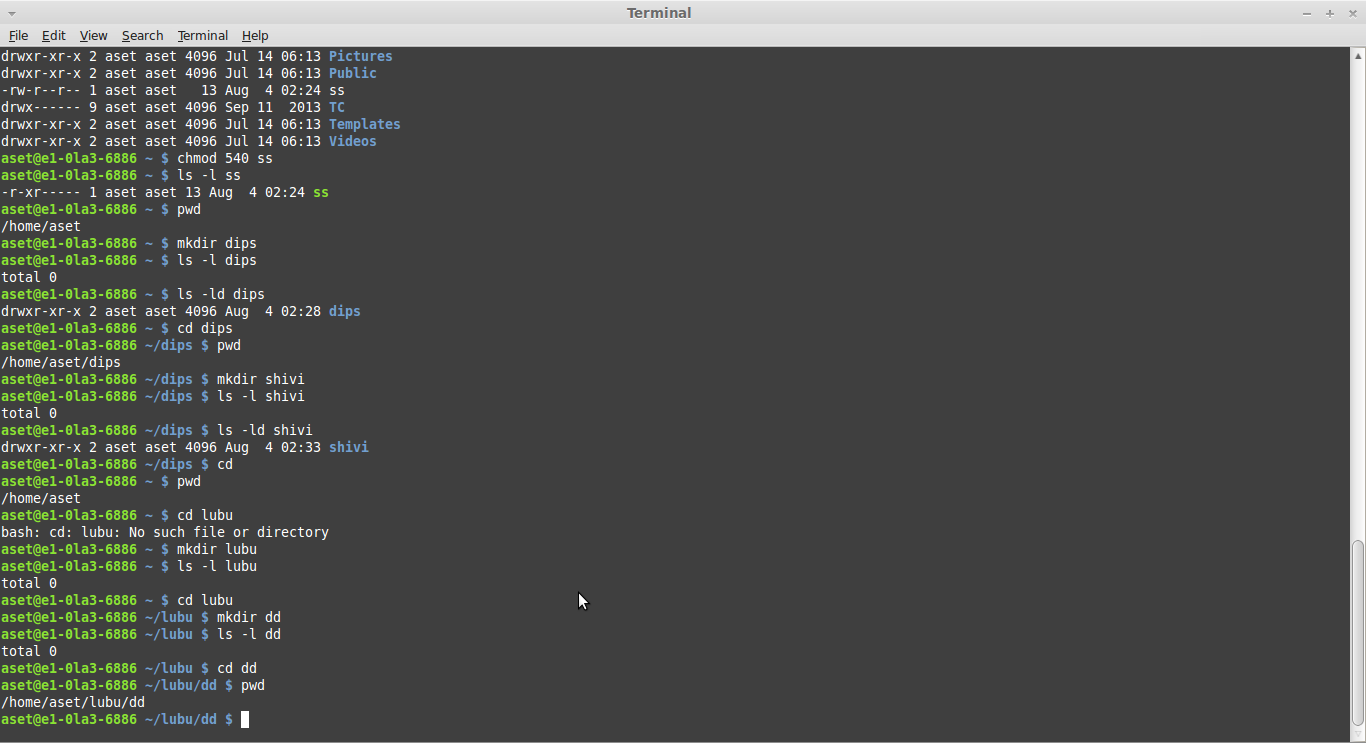


* **Command name:** cd <directory name>

**Syntax:** $cd dips

**Description:** This command is used to change the directory.

**Output:** The output is as follows:

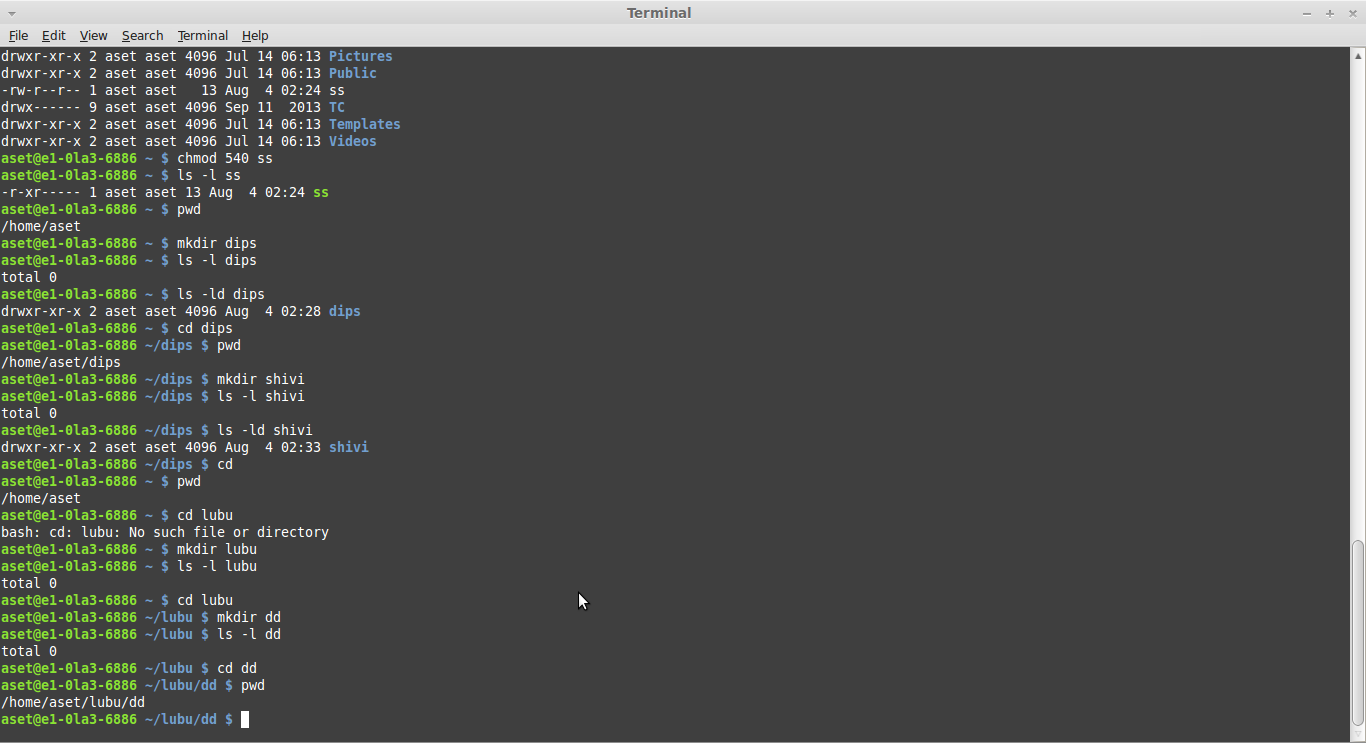


* **Command name:** cd

**Syntax:** cd

**Description:** This command is used for changing path of the directory from one directory to another.

**Output:** The output is as follows:

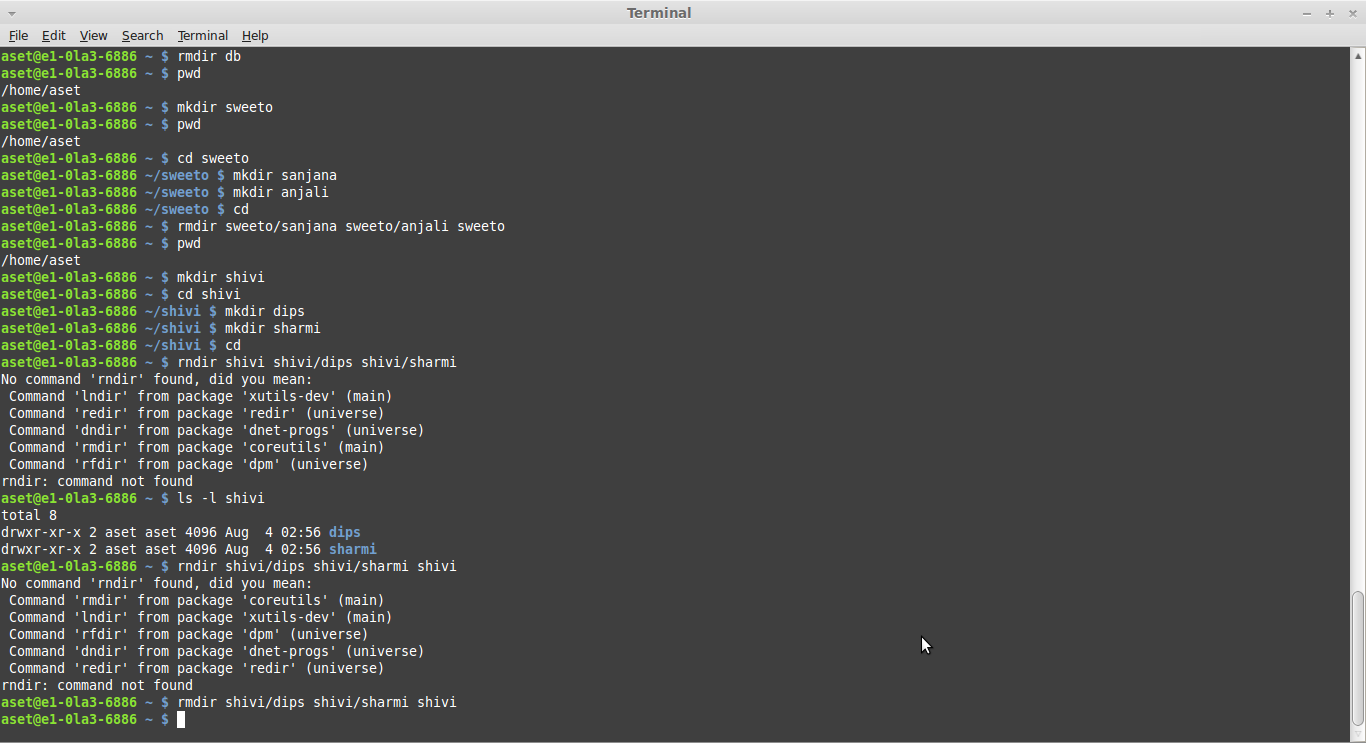


* **Command name:** rmdir

**Syntax:** rmdir

**Description:** This command is used for removing the directory (whenever we remove a directory, it should be empty otherwise the directory will not get deleted and will show error)

**Output:** The output is as follows:

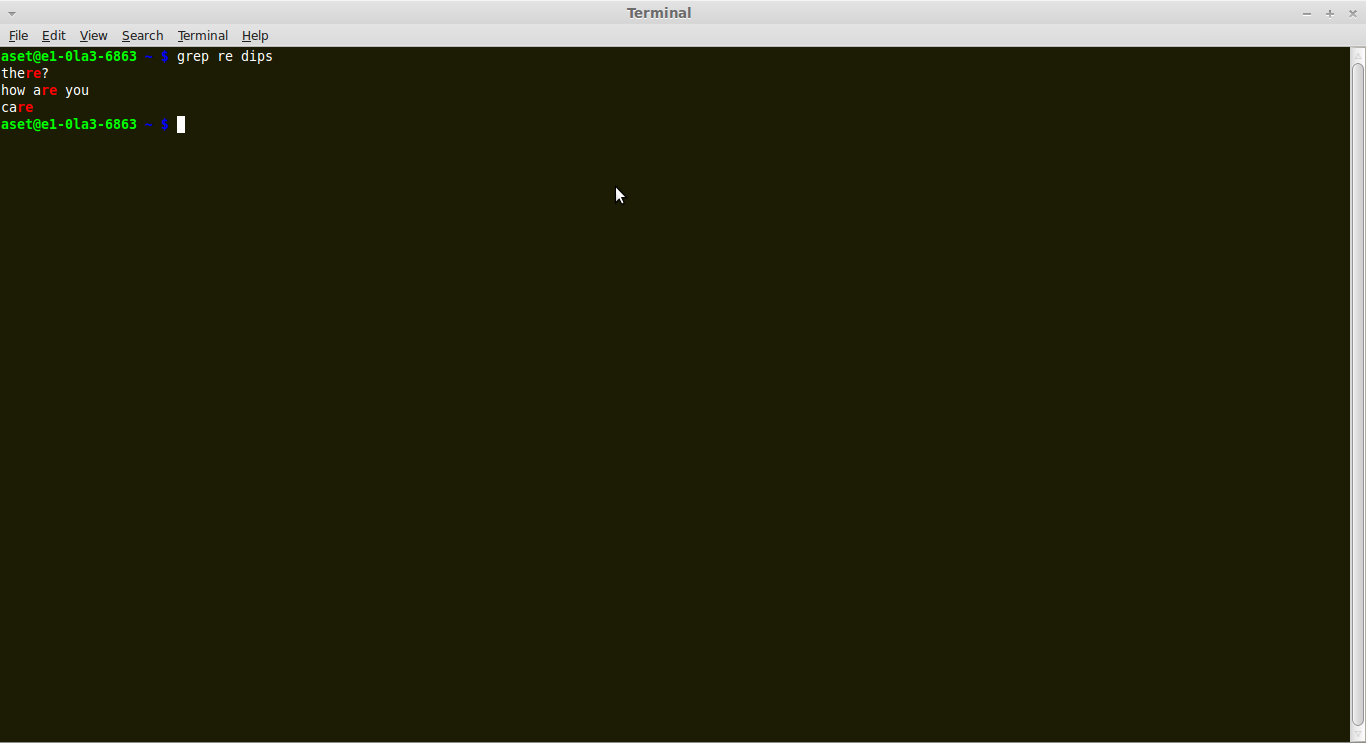


* **Command name:** grep <pattern><filename>

**Syntax:** $grep re dips

**Description:** Shows the list of words in the file and directory which are following a specific ‘re’ pattern.

**Output:** The output is as follows:

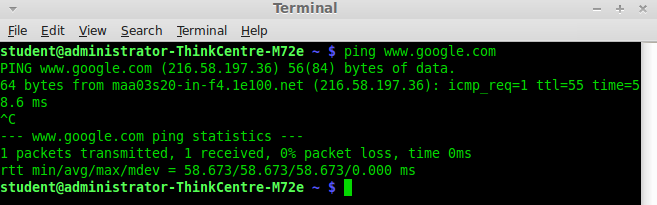


* **Command name:** ping

**Syntax:** ping www.google.com

**Description:** Tells the connectivity between two nodes.

**Output:** The output is as follows:

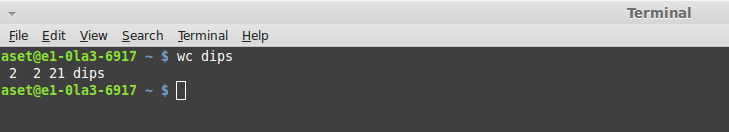


* **Command name:** wc

**Syntax:** $wc dips

**Description:**Displays the number oflines, words, bytesand finally displays the filename of file “dips”.

**Output:** The output is as follows:

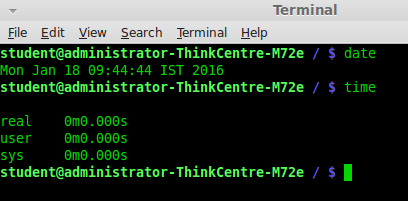


* **Command name:** time

**Syntax:** time

**Description:**Displays the present time.

**Output:** The output is as follows:

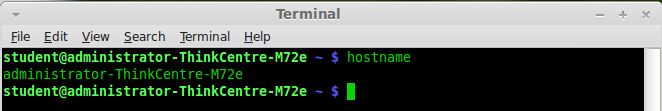


* **Command name:** hostname

**Syntax:** hostname

**Description:**Displays the hostname of the system.

**Output:** The output is as follows:



Result:The commands have been implemented successfully.



**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT – 3(A)**

**Objective: To establish straight configuration for LAN.**

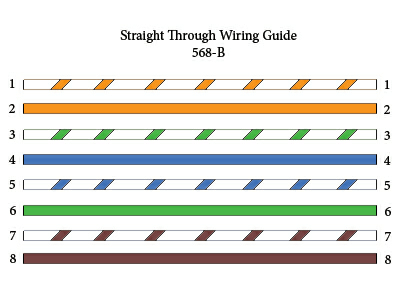
**Equipments Required:**

1. 2 RJ -45 connectors
2. Twisted pair cable
3. Gripping or Crimping tool
4. SLT-Tool

**Color Coding:**

1. Orange white
2. Orange
3. Green white
4. Blue
5. Blue white
6. Green
7. Brown white
8. Brown

Straight Configuration

1---------------------------------------------1

2---------------------------------------------2

3---------------------------------------------3

4---------------------------------------------4

5---------------------------------------------5

6---------------------------------------------6

7---------------------------------------------7

8---------------------------------------------8

**Theory:**

A **local area network (LAN)** is a computer network that connects computers and devices in a limited geographical area such as home, school, computer laboratory or office building. The defining characteristics of LANs, in contrast to [wide area networks (WANs)](http://en.wikipedia.org/wiki/Wide_area_network), include their usually higher data-transfer rates, smaller geographic area, and lack of a need for [leased telecommunication lines](http://en.wikipedia.org/wiki/Leased_line).

[ARCNET](http://en.wikipedia.org/wiki/ARCNET), [Token Ring](http://en.wikipedia.org/wiki/Token_Ring) and other technology standards have been used in the past, but [Ethernet](http://en.wikipedia.org/wiki/Ethernet) over [twisted pair](http://en.wikipedia.org/wiki/Twisted_pair) cabling, and [Wi-Fi](http://en.wikipedia.org/wiki/Wi-Fi) are the two most common technologies currently in use.

**Procedure:**

The outer covering of the wire is peeled off and according to requirement the wires are inserted in RJ-45 connector and punched with the help of punching tool after punching the wire is tested with SLT( Side locator tool).

**Result:**

The straight wiring for the LAN has been established and tested using SLT tool.

**Use:**

This type of wiring is used for connecting to PC or Hub.

**EXPERIMENT – 3(B)**

**Objective: To establish rollover configuration for LAN.**

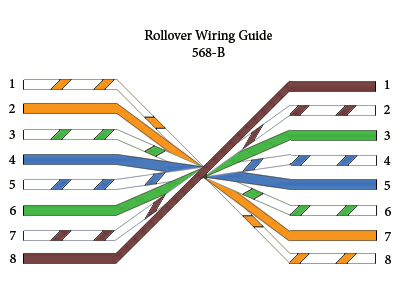
**Equipments Required:**

1. 2 RJ -45 connectors
2. Twisted pair cable
3. Gripping or Crimping tool
4. SLT-Tool

**Color Coding:**

1. Orange white
2. Orange
3. Green white
4. Blue
5. Blue white
6. Green
7. Brown white
8. Brown

Rollover Configuration

1---------------------------------------------8

2---------------------------------------------7

3---------------------------------------------6

4---------------------------------------------5

5---------------------------------------------4

6---------------------------------------------3

7---------------------------------------------2

8---------------------------------------------1

**Theory:**

A **local area network (LAN)** is a computer network that connects computers and devices in a limited geographical area such as home, school, computer laboratory or office building. The defining characteristics of LANs, in contrast to [wide area networks (WANs)](http://en.wikipedia.org/wiki/Wide_area_network), include their usually higher data-transfer rates, smaller geographic area, and lack of a need for [leased telecommunication lines](http://en.wikipedia.org/wiki/Leased_line).

[ARCNET](http://en.wikipedia.org/wiki/ARCNET), [Token Ring](http://en.wikipedia.org/wiki/Token_Ring) and other technology standards have been used in the past, but [Ethernet](http://en.wikipedia.org/wiki/Ethernet) over [twisted pair](http://en.wikipedia.org/wiki/Twisted_pair) cabling, and [Wi-Fi](http://en.wikipedia.org/wiki/Wi-Fi) are the two most common technologies currently in use.

**Procedure:**

The outer covering of the wire is peeled off and according to requirement the wires are inserted in RJ-45 connector and punched with the help of punching tool after punching the wire is tested with SLT( Side locator tool).

**Result:**

The rollover wiring for the LAN has been established and tested using SLT tool.

**Use:**

This type of wiring is used for configuration of the router.

**EXPERIMENT – 3(C)**

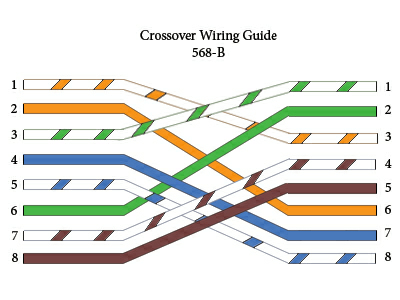
**Objective: To establish crossover configuration for LAN.**

**Equipments Required:**

1. 2 RJ -45 connectors
2. Twisted pair cable
3. Gripping or Crimping tool
4. SLT-Tool

**Color Coding:**

1. Orange white
2. Orange
3. Green white
4. Blue
5. Blue white
6. Green
7. Brown white
8. Brown

1---------------------------------------------3

2---------------------------------------------6

3---------------------------------------------1

4---------------------------------------------4

5---------------------------------------------5

6---------------------------------------------2

7---------------------------------------------7

8---------------------------------------------8

**Theory:**

A **local area network (LAN)** is a computer network that connects computers and devices in a limited geographical area such as home, school, computer laboratory or office building. The defining characteristics of LANs, in contrast to [wide area networks (WANs)](http://en.wikipedia.org/wiki/Wide_area_network), include their usually higher data-transfer rates, smaller geographic area, and lack of a need for [leased telecommunication lines](http://en.wikipedia.org/wiki/Leased_line).

[ARCNET](http://en.wikipedia.org/wiki/ARCNET), [Token Ring](http://en.wikipedia.org/wiki/Token_Ring) and other technology standards have been used in the past, but [Ethernet](http://en.wikipedia.org/wiki/Ethernet) over [twisted pair](http://en.wikipedia.org/wiki/Twisted_pair) cabling, and [Wi-Fi](http://en.wikipedia.org/wiki/Wi-Fi) are the two most common technologies currently in use.

**Procedure:**

The outer covering of the wire is peeled off and according to requirement the wires are inserted in RJ-45 connector and punched with the help of punching tool after punching the wire is tested with SLT (Side locator tool).

**Result:**

The crossover wiring for the LAN has been established and tested using SLT tool.

**Uses:**

This type of wiring is used for connecting PC to another PC.



**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT –4**

**Objective: WAP to generate Hamming code and check if the received codeword is correct.**

**Equipment/Software Used:**

|  |  |  |
| --- | --- | --- |
| S.no. | Hardware | Software |
| 1. | I7 processor | Os(windows 8) |
| 2. | 8 gb ram | Microsoft word |
| 3. | Keyboard | Turbo c/c++ |
| 4. | Mouse |  |
| 5. | Monitor |  |
| 6. | Printer |  |

**Source Code:**

#include<iostream.h>

#include<conio.h>

int xor(int a,int b)

{

if(a==b)

return 0;

else

return 1;

}

void main()

{

clrscr();

int a[8],p1,p2,p3,p4,c1,c2,c3,c4;

int b[12];

cout<<"\n Enter 8 bit dataword :";

for(int i=0;i<8;i++)

cin>>a[i];

b[2]=a[0];b[4]=a[1];b[5]=a[2];b[6]=a[3];b[8]=a[4];b[9]=a[5];

b[10]=a[6];b[11]=a[7];

p1=xor(b[2],b[4]); p1=xor(p1,b[6]); p1=xor(p1,b[8]); p1=xor(p1,b[10]);

p2=xor(b[2],b[5]); p2=xor(p2,b[6]); p2=xor(p2,b[9]); p2=xor(p2,b[10]);

p3=xor(b[4],b[5]); p3=xor(p3,b[6]); p3=xor(p3,b[11]);

p4=xor(b[8],b[9]); p4=xor(p4,b[10]); p4=xor(p4,b[11]);

b[0]=p1;b[1]=p2;b[3]=p3;b[7]=p4;

cout<<"\n The codeword is: ";

for(i=0;i<12;i++)

cout<<" "<<b[i];

c1=xor(b[2],b[4]); c1=xor(c1,b[6]); c1=xor(c1,b[8]); c1=xor(c1,b[10]); c1=xor(c1,b[0]);

c2=xor(b[2],b[5]); c2=xor(c2,b[6]); c2=xor(c2,b[9]); c2=xor(c2,b[10]); c2=xor(c2,b[1]);

c3=xor(b[4],b[5]); c3=xor(c3,b[6]); c3=xor(c3,b[11]); c3=xor(c3,b[3]);

c4=xor(b[8],b[9]); c4=xor(c4,b[10]); c4=xor(c4,b[11]); c4=xor(c4,b[7]);

cout<<"\n c1= "<<c1;

cout<<"\n c2= "<<c2;

cout<<"\n c3= "<<c3;

cout<<"\n c4= "<<c4;

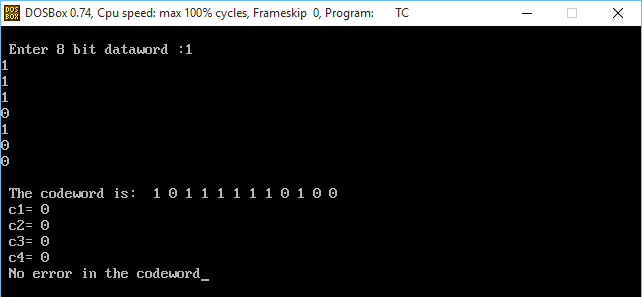
if(c1==c2 && c2==c3 && c3==c4 && c4==0)

cout<<"\n No error in the codeword";

getch();

}

**Output:**



**Result:** The program is successfully written and created in C language



**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT-5**

**OBJECTIVE: WAP which manipulates pure and slotted ALOHA based on user inputs.**

**SOFTWARE USED:** Turbo C++ , Windows Os

**SOURCE CODE:**

#include<iostream.h>

#include<conio.h>

int main()

{

clrscr();

int N=0;

float C=0,T=0,l=0,f=0,u=0,G=0;

cout<<"type c: ";a

cin>>C;

cout<<"\ntype t: ";

cin>>T;

cout<<"\ntype len: ";

cin>>l;

cout<<"\ntype fps: ";

cin>>f;

C\*=1024\*1024\*8;

u=1/l;

N=T\*((C\*u)-f);

G=N/f;

cout<<"\n(N): " <<N;

if(G>0 && G<=0.5)

cout<<" pure ";

else if (G>0.5 && G<=1)

cout<<" slotted";

else

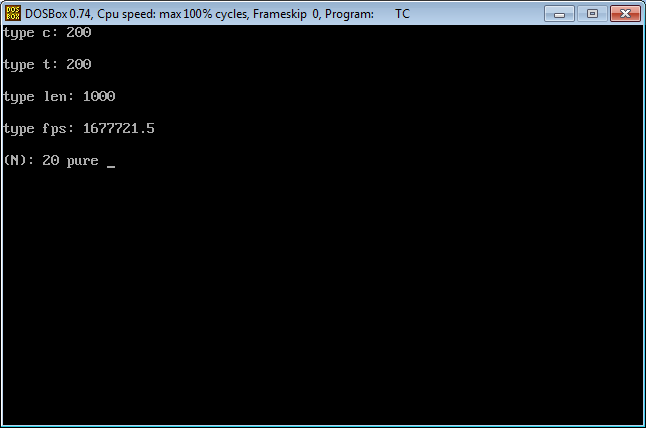
cout<<"\nerror.";

getch();

return 0;

}

**OUTPUT:**



**RESULT:** On entering the mentioned parameters program indicate the pure or slotted aloha.



**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT-6(A)**

**Objective: Write a Program that Translate between dotted decimal form to 32 bit binary.**

#include<iostream.h>

#include<conio.h>

void DEC2BIN(int dec)

{

for (int i = 128; i != 0; i=i>>1)

{

if (dec & i)

cout<<"1";

else

cout<<"0";

}

}

void main()

{

clrscr();

int i,j;

int dec[4];

int bin[8]={128,64,32,16,8,4,2,1};

cout<<"Enter the IP Address: ";

for(i=0;i<4;i++)

cin>>dec[i];

cout<<"The ip address is: "<<dec[0];

cout<<"."<<dec[1]<<"."<<dec[2]<<"."<<dec[3]<<endl;

for(i=0; i<4; i++)

{

DEC2BIN(dec[i]);

if(i!=3)

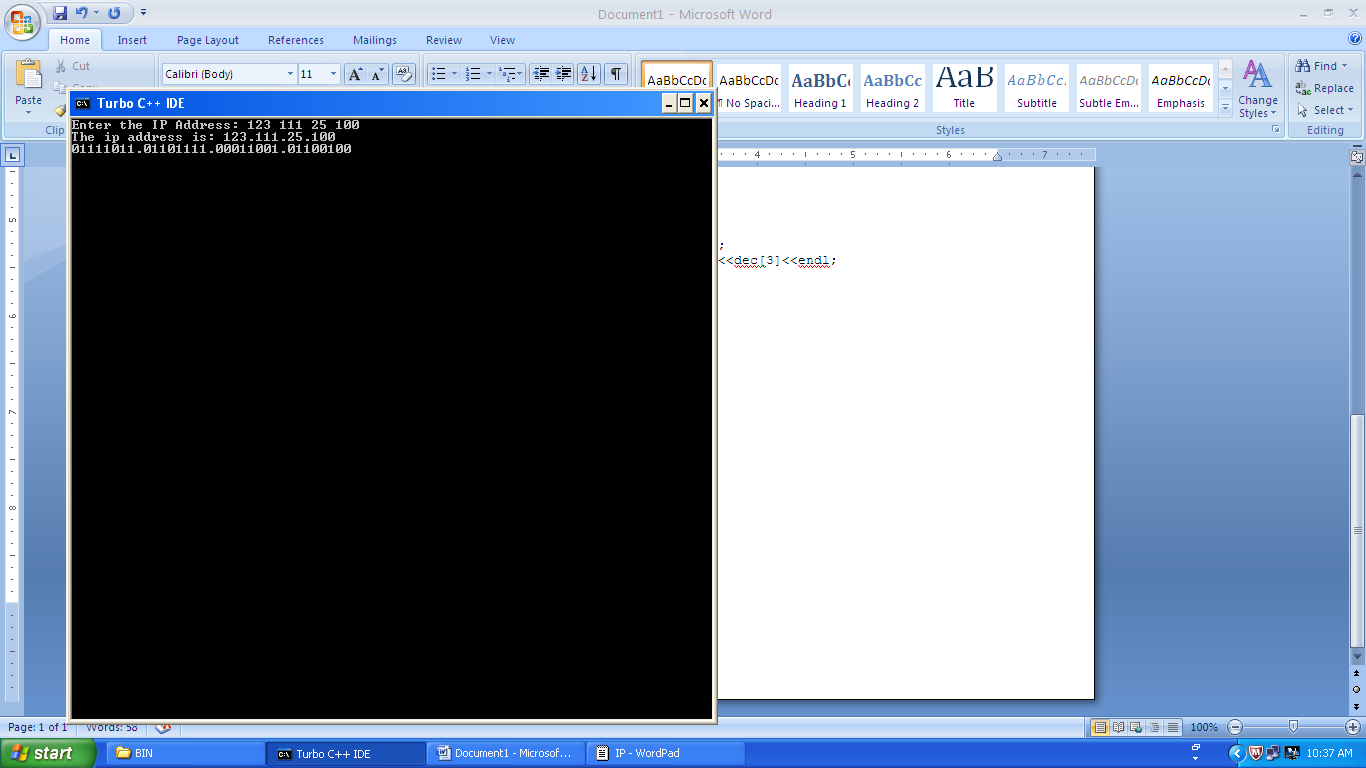
cout<<".";

}

getch();

}

**OUTPUT:**



**EXPERIMENT – 6(B)**

**Objective: Write a Program that Translate between 32-bit number to dotted decimal form.**

#include<iostream.h>

#include<conio.h>

#include<math.h>

int bintodeci(long int a)

{

int i=0,temp=0,temp2=0;

while(a!=0)

{

temp=a%10;

a=a/10;

temp2=temp2+temp\*pow(2,i);

i++;

}

return(temp2);

}

void main()

{

clrscr();

long int binary\_ip[4];

cout<<"Enter the IP address in binary \n";

for(int i=0;i<4;i++)

cin>>binary\_ip[i];

cout<<"The binary ip is: ";

cout<<binary\_ip[0]<<"."<<binary\_ip[1]<<"."<<binary\_ip[2]<<"."<<binary\_ip[3];

int a[4];

for(i=0;i<4;i++)

{

a[i]=bintodeci(binary\_ip[i]);

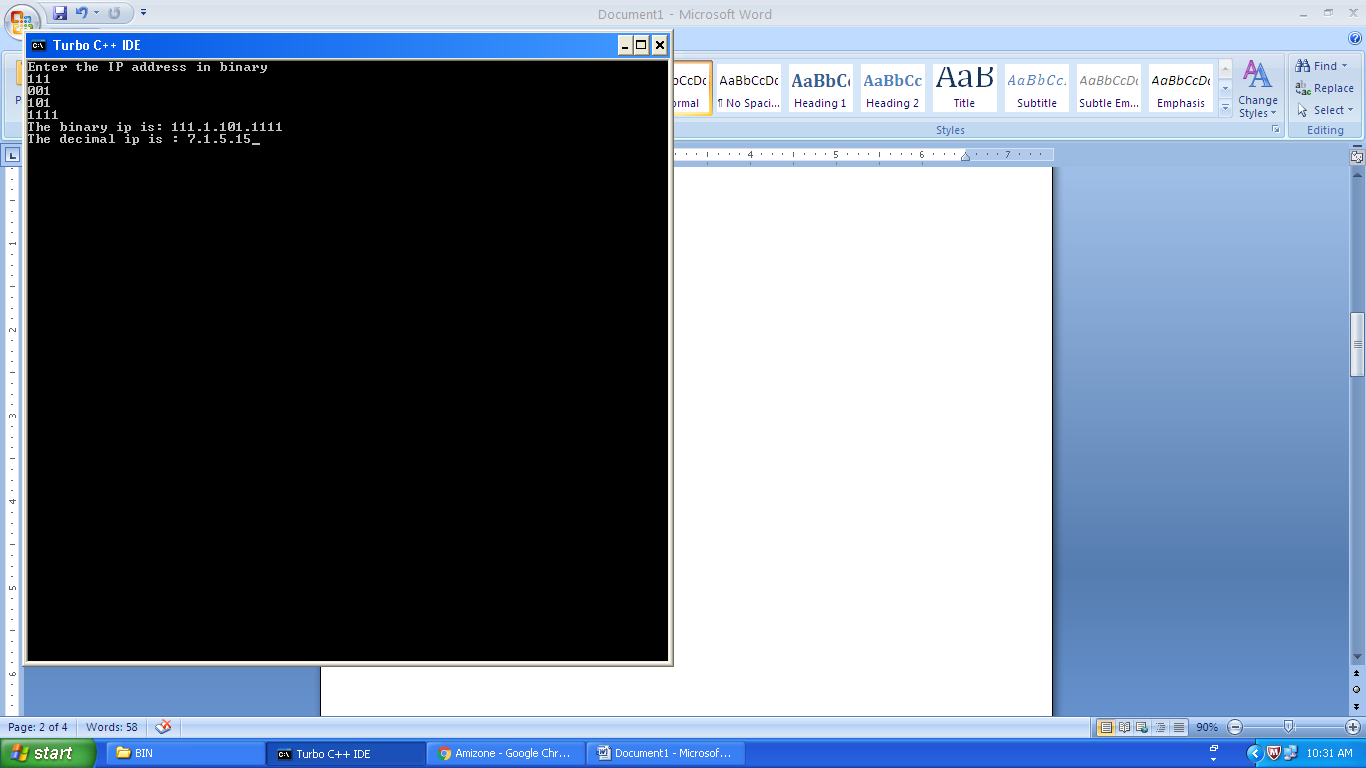
}

cout<<"\nThe decimal ip is : "<<a[0]<<"."<<a[1]<<"."<<a[2]<<"."<<a[3];

getch();

}

**OUTPUT:**





**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT – 7**

**Objective: To enter the IP address in binary or decimal form, and find its class.**

#include<math.h>

#include<conio.h>

#include<iostream.h>

int bintodeci(long int a)

{

int i=0,temp=0,temp2=0;

while(a!=0)

{

temp=a%10;

a=a/10;

temp2=temp2+temp\*pow(2,i);

i++;

}

return(temp2);

}

void main()

{

clrscr();

int ip[4];

long int bip[4];

cout<<"1. BINARY\n";

cout<<"2. DECIMAL \n";

int ch;

cin>>ch;

switch(ch)

{

case 1: cout<<"Enter the IP address \n";

for(int j=0;j<4;j++)

cin>>bip[j];

ip[0]=bintodeci(bip[0]);

ip[1]=bintodeci(bip[1]);

ip[2]=bintodeci(bip[2]);

ip[3]=bintodeci(bip[3]);

break;

case 2: cout<<"Enter the IP address \n";

for(int i=0;i<4;i++)

cin>>ip[i]; break;

}

if((ip[0]>=0)&&(ip[0]<=127))

{

cout<<"CLASS A";

}

else if((ip[0]>=128)&&(ip[0]<=191))

{

cout<<"CLASS B";

}

else if ((ip[0]>=192)&&(ip[0]<=239))

{

cout<<"CLASS C";

}

else if((ip[0]>=240)&&(ip[0]<=247))

{

cout<<"CLASS D";

}

if((ip[0]>=248)&&(ip[0]<=255))

{

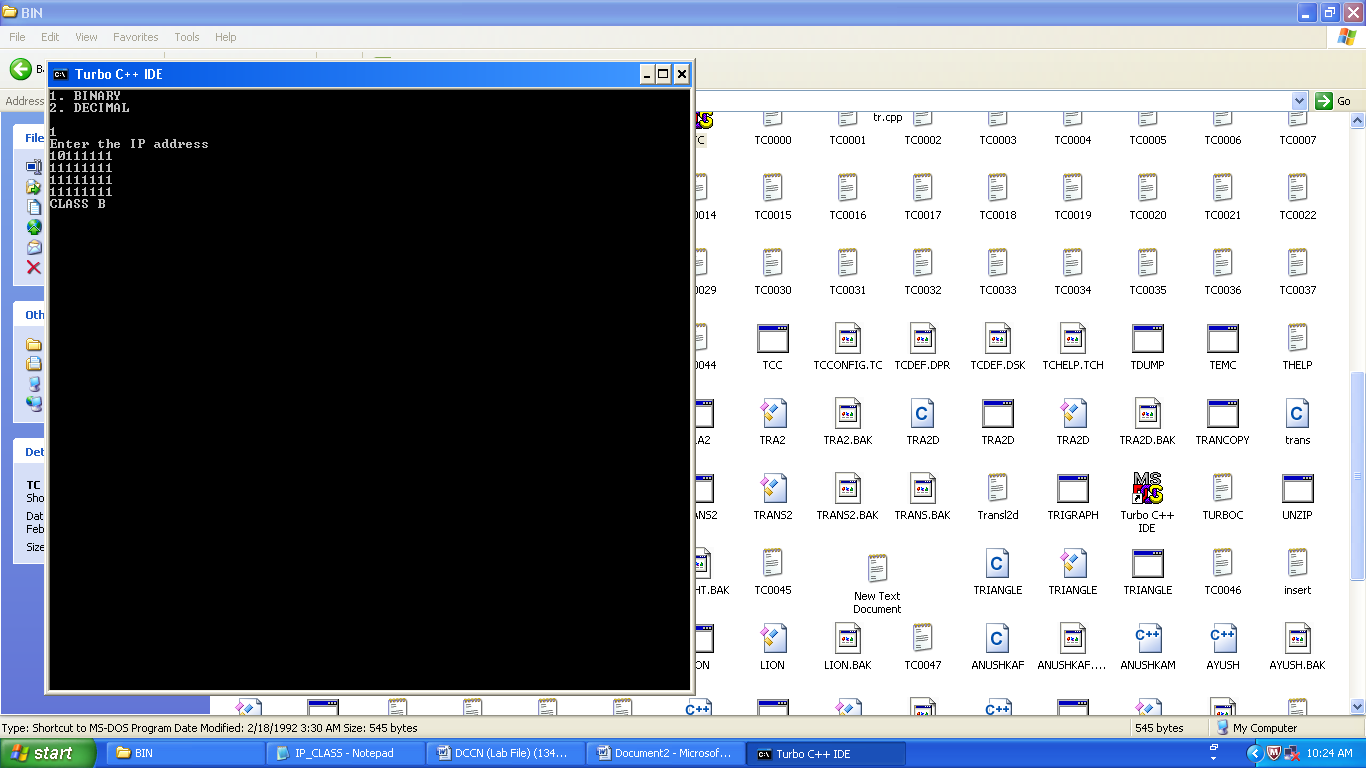
cout<<"CLASS E";

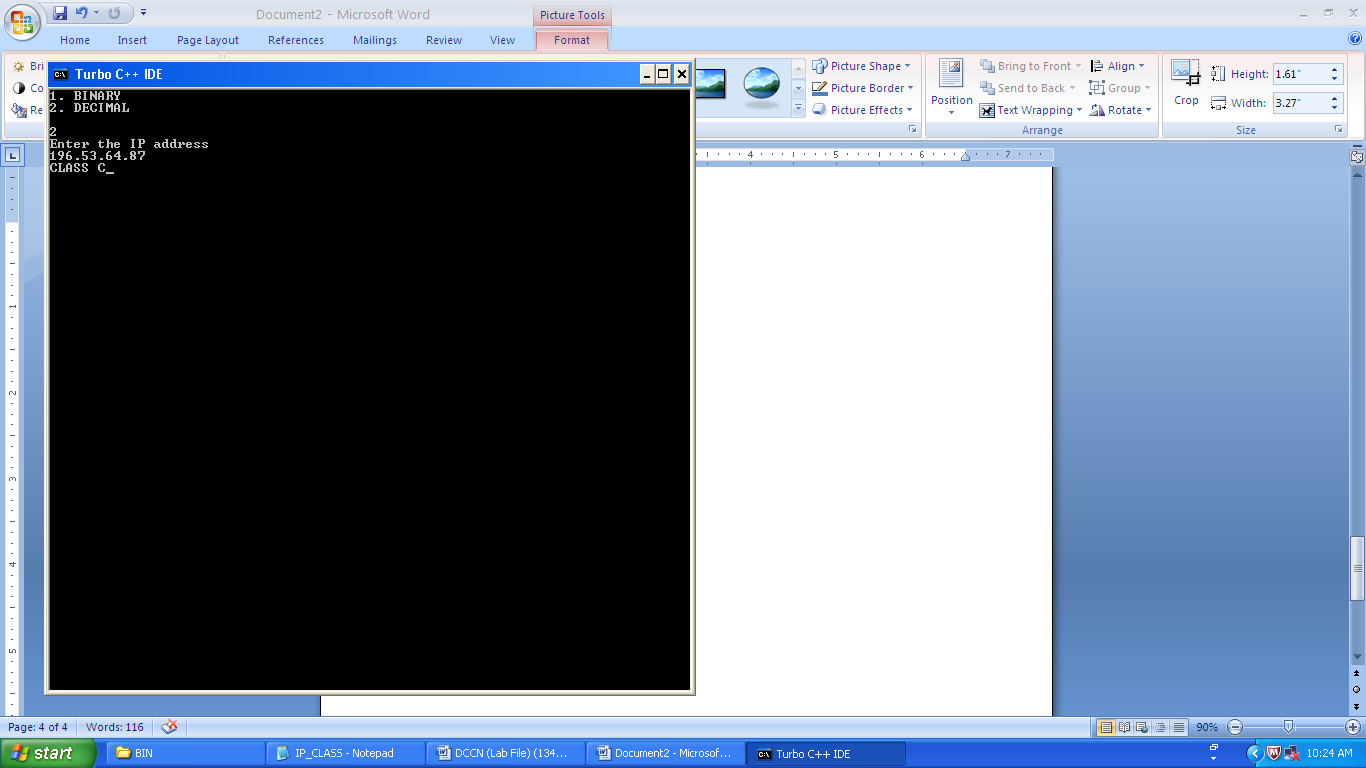
}

getch();

}

**OUTPUT:**





**RESULT:** Thus, we were able to determine the class of the entered IP Address.

**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT – 8**

**Objective:** Write a program to implement stop and wait ARQ.

#include<iostream.h>

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#include<dos.h>

#define time 5

#define max\_seq 1

#define tot\_pack 5

int randn(int n)

{

return rand()%n + 1;

}

typedef struct

{

int data;

}packet;

typedef struct

{

int kind;

int seq;

int ack;

packet info;

}frame;

typedef enum{ frame\_arrival,error,time\_out}event\_type;

frame data1;

//creating prototype

void from\_network\_layer(packet \*);

void to\_physical\_layer(frame \*);

void to\_network\_layer(packet \*);

void from\_physical\_layer(frame\*);

void sender();

void receiver();

void wait\_for\_event\_sender(event\_type \*);

void wait\_for\_event\_receiver(event\_type \*);

//end

#define inc(k) if(k<max\_seq)k++;else k=0;

int i=1;

char turn;

int disc=0;

int main()

{

while(!disc)

{ sender();

// delay(400);

receiver();

}

getchar();

}

void sender()

{ static int frame\_to\_send=0;

static frame s;

packet buffer;

event\_type event;

static int flag=0; //first place

if (flag==0)

{

from\_network\_layer(&buffer);

s.info=buffer;

s.seq=frame\_to\_send;

cout<<"\nsender information \t"<<s.info.data<<"\n";

cout<<"\nsequence no. \t"<<s.seq;

turn='r';

to\_physical\_layer(&s);

flag=1;

}

wait\_for\_event\_sender(&event);

if(turn=='s')

{

if(event==frame\_arrival)

{

from\_network\_layer(&buffer);

inc(frame\_to\_send);

s.info=buffer;

s.seq=frame\_to\_send;

cout<<"\nsender information \t"<<s.info.data<<"\n";

cout<<"\nsequence no. \t"<<s.seq<<"\n";

getch();

turn='r';

to\_physical\_layer(&s);

}

}

} //end of sender function

void from\_network\_layer(packet \*buffer)

{

(\*buffer).data=i;

i++;

} //end of from network layer function

void to\_physical\_layer(frame \*s)

{

data1=\*s;

} //end of to physical layer function

void wait\_for\_event\_sender(event\_type \*e)

{

static int timer=0;

if(turn=='s')

{ timer++;

//timer=0;

return ;

}

else //event is frame arrival

{

timer=0;

\*e=frame\_arrival;

}

} //end of wait for event function

void receiver()

{

static int frame\_expected=0;

frame s,r;

event\_type event;

wait\_for\_event\_receiver(&event);

if(turn=='r')

{ if(event==frame\_arrival)

{

from\_physical\_layer(&r);

if(r.seq==frame\_expected)

{

to\_network\_layer(&r.info);

inc (frame\_expected);

}

else

cout<<"\nrceceiver :acknowledgement resent \n";

getch();

turn='s';

to\_physical\_layer(&s);

}

}

} //end of receiver function

void wait\_for\_event\_receiver(event\_type \*e)

{

if(turn=='r')

{

\*e=frame\_arrival;

}

}

void from\_physical\_layer(frame \*buffer)

{

\*buffer=data1;

}

void to\_network\_layer(packet \*buffer)

{

cout<<"\nreceiver : packet received \t"<< i-1;

cout<<"\n acknowledgement sent \t";

getch();

if(i>tot\_pack)

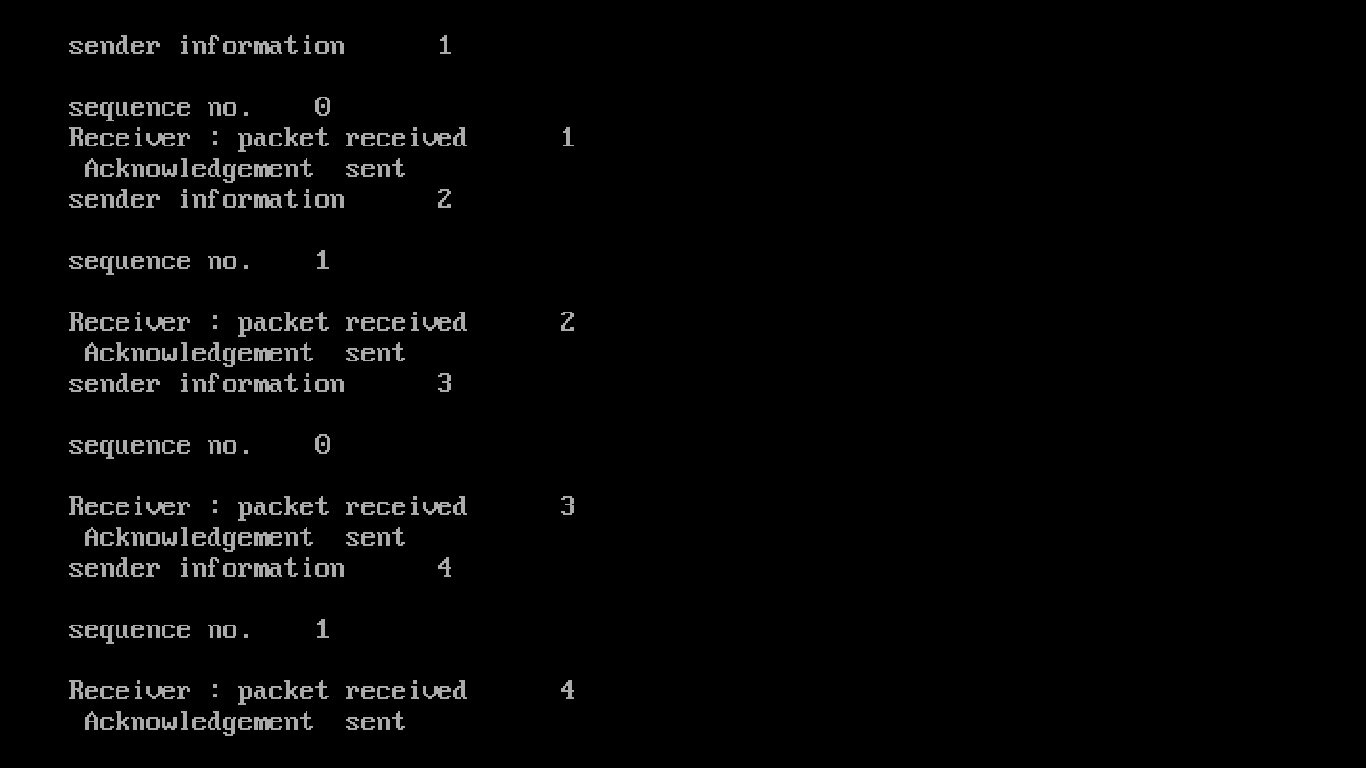
{ disc=1;

cout<<"\ndiscontinue\n";

}

} //end of network layer function

**OUTPUT:**



**RESULT:** Thus, we were able to implement stop and wait ARQ.

**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT – 9**

**OBJECTIVE: Write a Program to find shortest path using Dijkstra’s algorithm.**

**SOFTWARE USED:** Turbo C++ , Windows OS

**SOURCE CODE:**

#include<iostream.h>

#include<conio.h>

#include<limits.h>

#define TRUE 1

#define FALSE 0

class Dijkstra

{

private:

int adjMatrix[15][15];

int predecessor[15],cost[15];

int marking[15];

int source;

int nodes;

public:

void read();

void initialize();

int getClosestUnmarkedNode();

void calcCost();

void output();

void printPath(int);

};

void Dijkstra::read()

{

cout<<"\nnodes: ";

cin>>nodes;

for(int i=0;i<nodes;i++)

{

cout<<"\ntype "<<i+1 <<" : ";

for(int j=0;j<nodes;j++)

{

cin>>adjMatrix[i][j];

}

}

cout<<"\nsrc is: ";

cin>>source;

}

void Dijkstra::initialize()

{

for(int i=0;i<nodes;i++)

{

marking[i] = FALSE;

predecessor[i] = -1;

cost[i] = INT\_MAX;

}

cost[source] = 0;

}

int Dijkstra::getClosestUnmarkedNode()

{

int minCost = INT\_MAX;

int closestUnmarkedNode;

for(int i=0;i<nodes;i++)

{

if((!marking[i]) && (cost[i] <= minCost))

{

minCost = cost[i];

closestUnmarkedNode = i;

}

}

return closestUnmarkedNode;

}

void Dijkstra::calcCost()

{

initialize();

int closestUnmarkedNode;

for(int ctr=0;ctr<nodes;ctr++)

{

closestUnmarkedNode = getClosestUnmarkedNode();

marking[closestUnmarkedNode] = TRUE;

for(int i=0;i<nodes;i++)

if((!marking[i]) && (adjMatrix[closestUnmarkedNode][i] > 0))

if(cost[i] > cost[closestUnmarkedNode] + adjMatrix[closestUnmarkedNode][i])

{

cost[i] = cost[closestUnmarkedNode] + adjMatrix[closestUnmarkedNode][i];

predecessor[i] = closestUnmarkedNode;

}

}

}

void Dijkstra::printPath(int node)

{

if(node==source)

cout<<"\n" <<(char)(node+65);

else if(predecessor[node] == -1)

cout<<"\nNo";

else

{

printPath(predecessor[node]);

cout<<">" <<(char)(node+65);

}

}

void Dijkstra::output()

{

for(int i=0;i<nodes;i++)

{

if(i==source)

cout<<"\n" <<(char)(source+65) <<">" <<(char)(source+65);

else

printPath(i);

cout<<"...C=" <<cost[i];

}

}

int main()

{

clrscr();

Dijkstra ntwrk;

ntwrk.read();

ntwrk.calcCost();

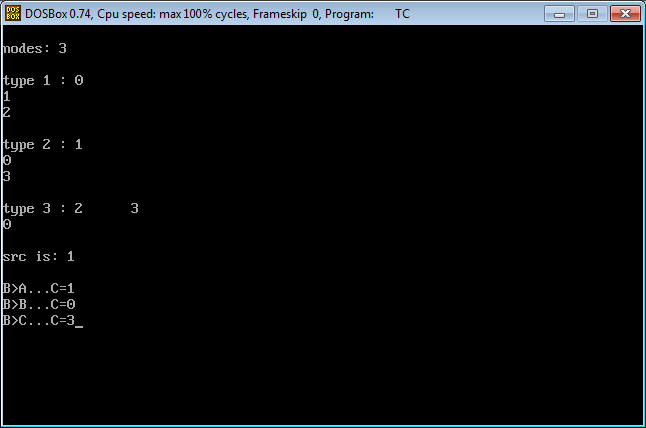
ntwrk.output();

getch();

return 0;

}

**OUTPUT:**

****

**RESULT:** It displays minimum cost involved with that particular path correctly.



**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**EXPERIMENT – 10**

**OBJECTIVE: Write a Program to find shortest path using Bellman Ford algorithm.**

**SOFTWARE USED:** Turbo C++ , Windows OS

**SOURCE CODE:**

#include<iostream.h>

#include<stdio.h>

#include<conio.h>

#define INFINITY 999

struct node

{

int cost;

int value;

int from;

}a[5];

void addEdge(int am[][5],int src,int dest,int cost)

{

am[src][dest] = cost;

return;

}

void bell(int am[][5])

{

int i, j, k, c = 0, temp;

a[0].cost = 0;

a[0].from = 0;

a[0].value = 0;

for (i = 1; i < 5; i++)

{

a[i].from = 0;

a[i].cost = INFINITY;

a[i].value = 0;

}

while (c < 5)

{

int min = 999;

for (i = 0; i < 5; i++)

{

if (min > a[i].cost && a[i].value == 0)

{

min = a[i].cost;

}

else

{

continue;

}

}

for (i = 0; i < 5; i++)

{

if (min == a[i].cost && a[i].value == 0)

{

break;

}

else

{

continue;

}

}

temp = i;

for (k = 0; k < 5; k++)

{

if (am[temp][k] + a[temp].cost < a[k].cost)

{

a[k].cost = am[temp][k] + a[temp].cost;

a[k].from = temp;

}

else

{

continue;

}

}

a[temp].value = 1;

c++;

}

cout<<"Cost"<<"\t"<<"Source Node"<<endl;

for (j = 0; j < 5; j++)

{

cout<<a[j].cost<<"\t"<<a[j].from<<endl;

}

}

int main()

{

int n, am[5][5], c = 0, i, j, cost;

for (i = 0; i < 5; i++)

{

for (j = 0; j < 5; j++)

{

am[i][j] = INFINITY;

}

}

while (c < 8)

{

cout<<"Enter the source, destination and cost of edge\n";

cin>>i>>j>>cost;

addEdge(am, i, j, cost);

c++;

}

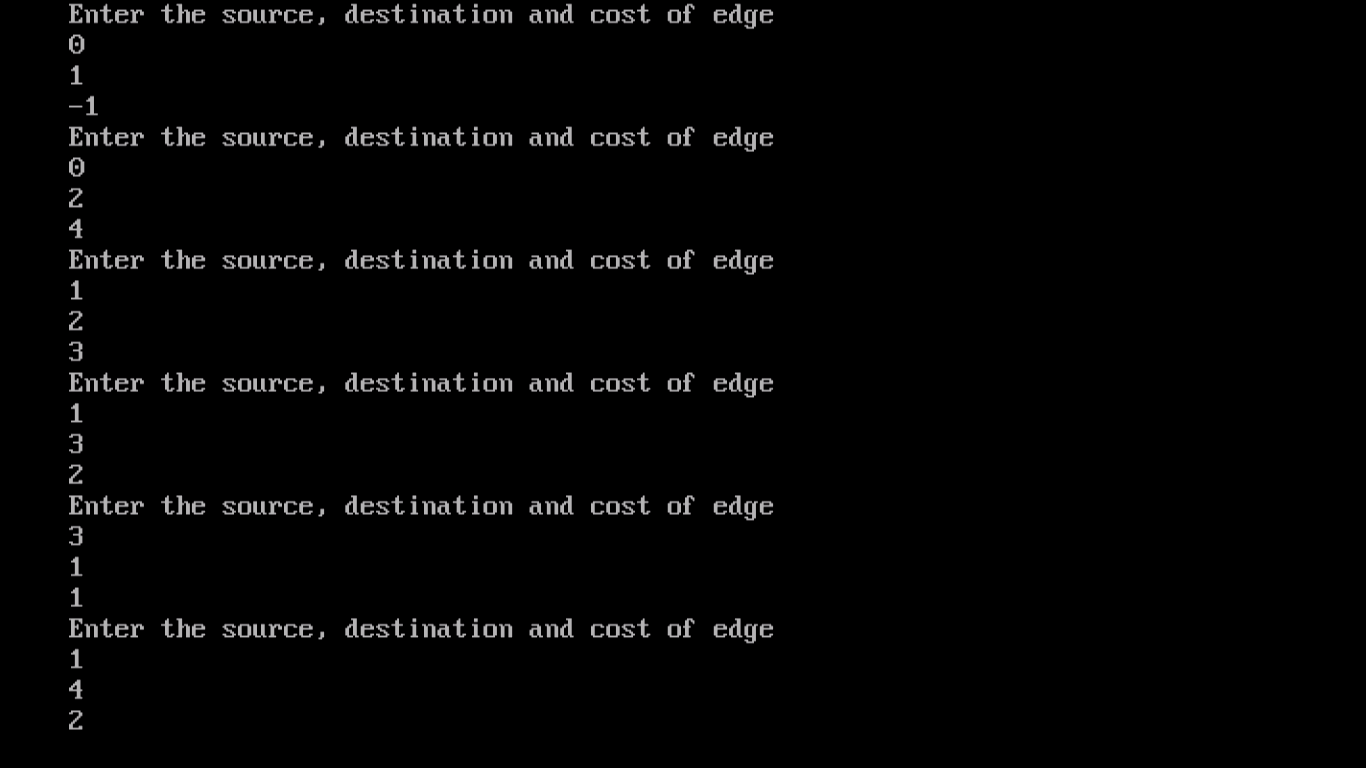
bell(am);

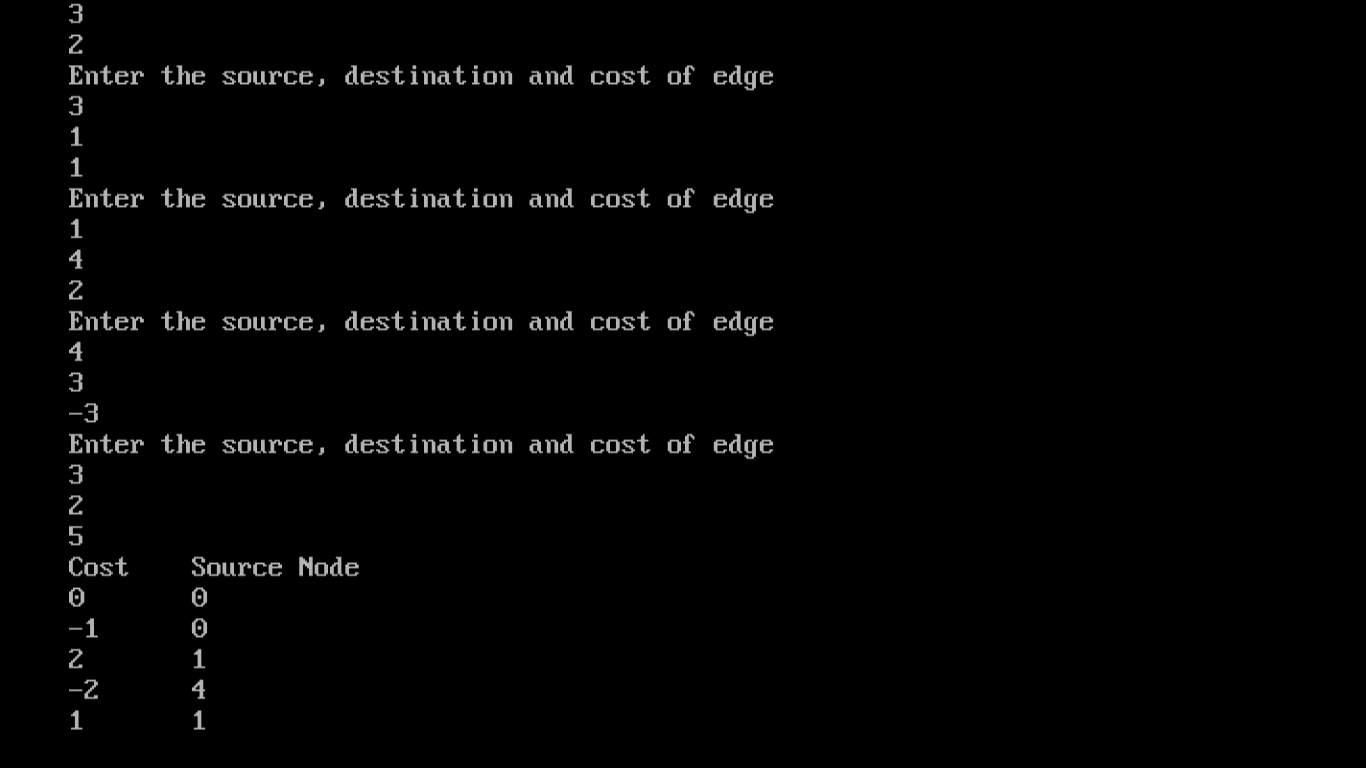
getch();

return(0);

}

**OUTPUT:**





**RESULT:** It displays minimum cost involved with that particular path correctly.



**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**internal evaluation for Mandatory Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept (A) | 2 |  |  |
| Implementation (B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 | | |

**DESIGN BASED OPEN-ENDED EXPERIMENT**

**OBJECTIVE:** WAP to implement bit and byte stuffing for framing.

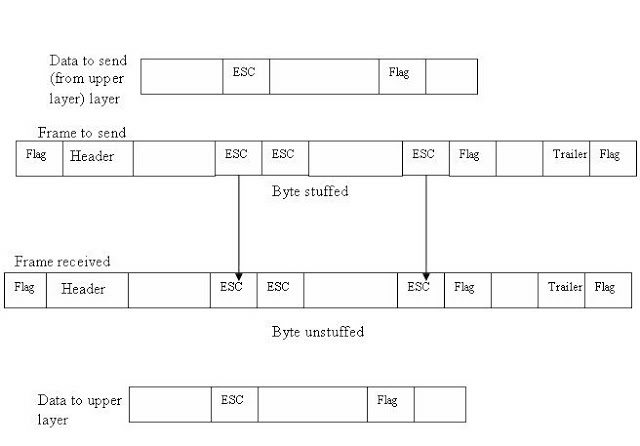
**EXECUTIVE SUMMARY:** While sending data over network, the data link layer divide into frames. Framing have several advantages than send raw very large data. It reduces the probability of error and reduces the amount of retransmission needed.

There exist two general methods for framing: **fixed size framing and variable size framing**. In fixed size framing, the data divided into fixed size frames and send over the transmission media. In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. ATM network use fixed size packets called cells.

In variable size framing, the data divided into variable size frames. Here the network system needs a mechanism to distinguish the end of a packet and beginning of another one. Two protocols are used for this purpose: **character oriented protocol and bit oriented protocol**.

**Character-Oriented Protocols**

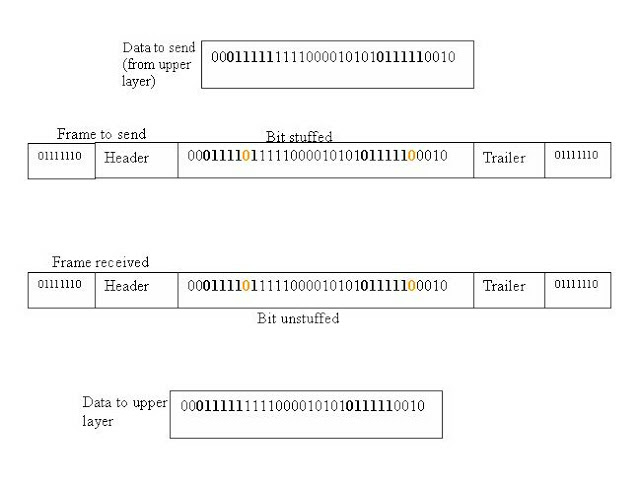
In character-oriented protocol, we add special characters (called flag) to distinguish beginning and end of a frame. Usually flag has 8-bit length. The character-oriented protocols are popular only with text data. While using character–oriented protocol another problem is arises, pattern used for the flag may also part of the data to send. If this happens, the destination node, when it encounters this pattern in the middle of the data, assumes it has reached the end of the frame. To deal with this problem, a **byte stuffing** (**also known as**  
**character stuffing**) approach was included to character-oriented protocol. In byte stuffing a special byte is add to the data part, this is known as **escape character** (ESC). The escape characters have a predefined pattern. The receiver removes the escape character and keeps the data part. It cause to another problem, if the text contains escape characters as part of data. To deal with this, an escape character is prefixed with another escape character. The following figure explains everything we discussed about character stuffing.



**BYTE STUFFING**

**Bit-Oriented Protocols**

In a bit-oriented protocol, the data to send is a series of bits. In order to distinguish frames, most protocols use a bit pattern of 8-bit length (01111110) as flag at the beginning and end of each frame. Here also cause the problem of appearance of flag in the data part to deal with this an extra bit added. This method is called **bitstuffing.** In bit stuffing, if a 0 and five successive 1 bits are encountered, an extra 0 is added. The receiver node removes the extra-added zero. This process is illustrate below:



**BIT STUFFING**

Simply, Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data. Byte stuffing is the method of adding 1 extra byte if there is a flag or escape character in the text.

**IMPLEMENTATION:**

**BIT STUFFING:**

#include<stdio.h>

#include<conio.h>

#include<string.h>

void main()

{

int a[20],b[30],i,j,k,count,n;

clrscr();

printf("Enter frame length:");

scanf("%d",&n);

printf("Enter input frame (0's & 1's only):");

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

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

i=0; count=1; j=0;

while(i<n)

{

if(a[i]==1)

{

b[j]=a[i];

for(k=i+1;a[k]==1 && k<n && count<5;k++)

{

j++;

b[j]=a[k];

count++;

if(count==5)

{

j++;

b[j]=0;

}

i=k;

}}

else

{

b[j]=a[i];

}

i++;

j++;

}

printf("After stuffing the frame is:");

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

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

getch();

}

**BYTE STUFFING:**

#include<stdio.h>

#include<conio.h>

void main()

{

char frame[100],str[50][50];

char flag='z';

char esc='x';

inti,j,k=0,n;

frame[k++]='z';

clrscr();

printf("Enter no.of String ::\t");

scanf("%d",&n);

printf("Enter String \n");

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

  {

gets(str[i]);

  }

printf("You entered ::\n");

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

   {

puts(str[i]);

   }

printf("\n");

for(i=1;i<=n;i++)

  {

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

      {

    if(str[i][j]!=flag&&str[i][j]!=esc)

    {

        frame[k++]=str[i][j];

    }

    else{

        frame[k++]='x';

        frame[k++]=str[i][j];

    }

      }

  }

frame[k++]='z';

frame[k++]='\0';

printf("------------------------------\n");

printf("Byte stuffing at sender side:\n\n");

printf("------------------------------\n");

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

  {

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

  }

printf("\n------------------------------");

printf("\nByte stuffing at receiver side\n\n");

printf("------------------------------\n");

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

  {

if(frame[i]=='x'|| frame[i]=='z')

     {

i++;

     }

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

  }

getch();

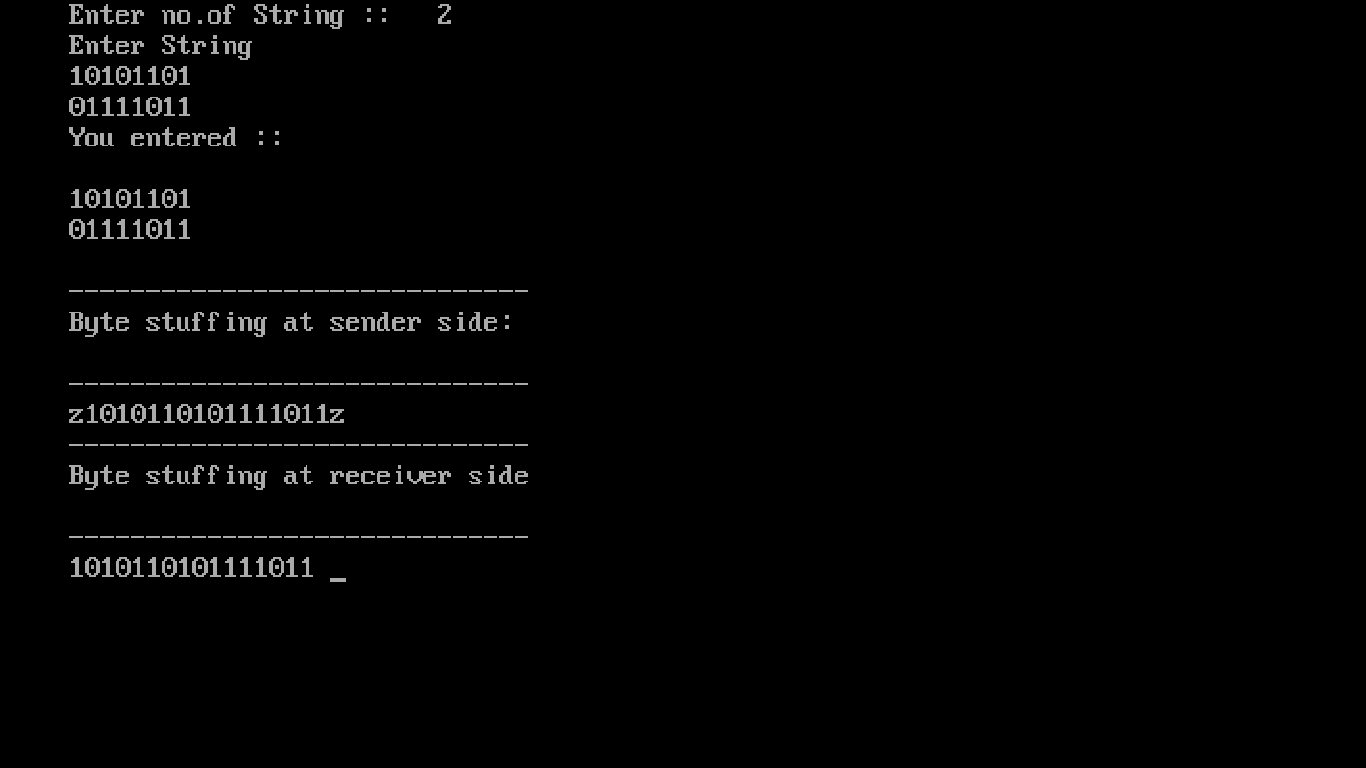
}

**TESTING & RESULTS:**

**BIT STUFFING:**



**BYTE STUFFING:**



**RESULT:** Thus, we implemented bit stuffing and byte stuffing using the above code.



**Faculty Name: Mr. Kunal Gupta**

**Signature:**

**Date:**

**INTERNAL EVALUATION FOR DESIGN BASED EXPERIMENT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Designing Concept (D) | 3 |  |  |
| Application of Knowledge (E) | 2 |  |  |
| Performance (F) | 3 |  |  |
| Result (G) | 2 |  |  |
| Total | 10 |  |  |

**INTERNAL EVALUATION FOR VIVA**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Clarity of the Subject (H) | 2 |  |  |
| Quality of theoretical Discussion (I) | 3 |  |  |
| Total | 5 |  |  |