COMPUTER NETWORKS

(GR22A3050)

LAB: COURSE FILE

Gokaraju Rangaraju Institute of Engineering and Technology
Bachupally, Kukatpally, Hyderabad - 500072

INDEX

Task 1: Implement the following Data Link Layer framing methods.

1a). BIT STUFFING.

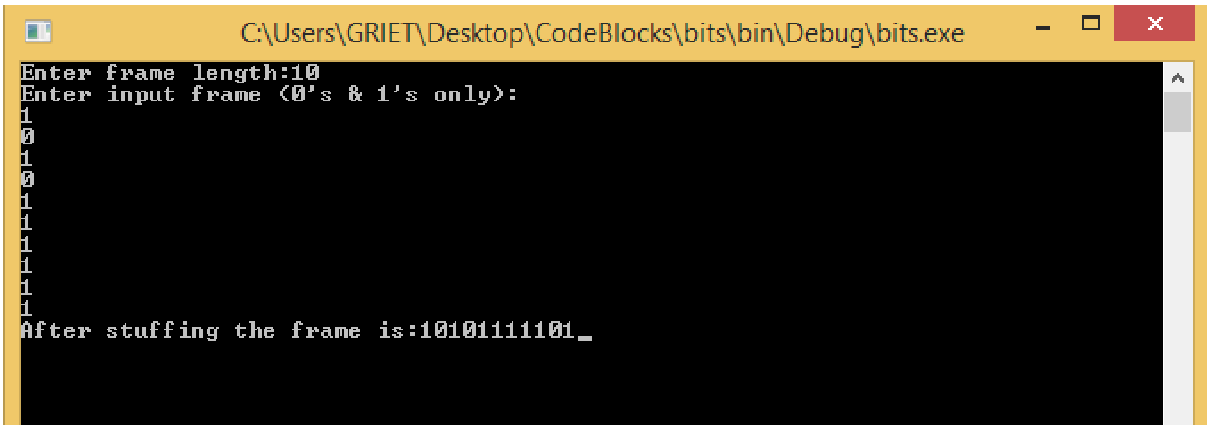
OBJECTIVE: Implement the data link layer framing method.
RESOURCE: Code blocks

PROGRAM LOGIC:
The new technique allows data frames to contain an arbitrary number if bits and allows character codes with an arbitrary no of bits per character. Each frame begins and ends with special bit pattern, 01111110, called a flag byte. Whenever the sender ' s data link layer encounters five consecutive ones in the data ,it automatically stuffs a 0 bit into the out goi ng bit stream. This bit stuffing is analogous to character stuffing, in which a DLE is stuffed into the outgoing g character stream before DLE in the data.

1a.) Bit Stuffing code:

```
#include <stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
int a[20],b[30],i,j,k,count,n;
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();
}
```

Output:



```
Enter frame length:10
Enter input frame (0's & 1's only):
1
0
1
0
1
1
1
1
1
1
After stuffing the frame is:10101111101
```

NAME OF THE EXPERIMENT: 1 b) CHARACTER STUFFING

OBJECTIVE: Implement the data link layer framing method.

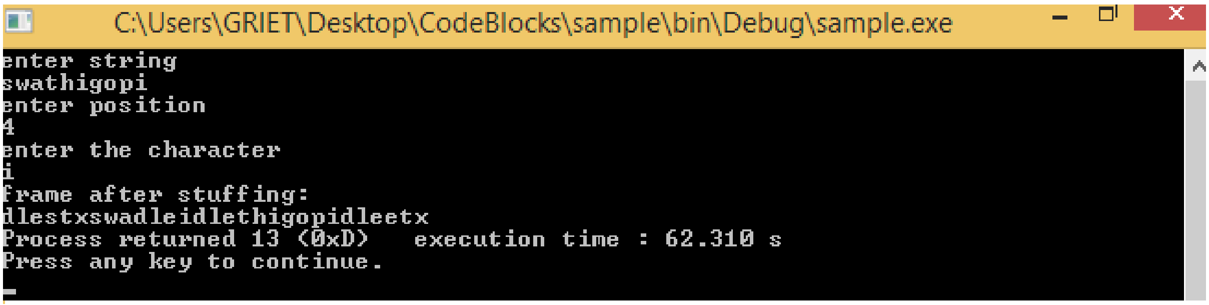
RESOURCE: Codeblocks

PROGRAM LOGIC:
The framing method gets around the problem of resynchronization after an error by having each frame start with the ASCII character sequence DLE STX and the sequence DLE ETX. If the destination ever losses the track of the frame boundaries all it has to do is look for DLE STX or DLE ETX characters to figure out. The data link layer on the receiving g end removes the DLE bbefore the data are given to the network layer. This technique is called character stuffing.

PROGRAM FOR CHARACTER STUFFING

```
#include <stdio.h>
#include <stdlib.h>
#include<string.h>
#include<process.h>
void main()
{
int i=0,j=0,n,pos;char a[20],b[50],ch;
printf("enter string\n");
scanf("%s",&a);
 n=strlen(a);
printf("enter position\n");
scanf("%d",&pos);
 if(pos>n)
{
printf("invalid position, Enter again :");
 scanf("%d",&pos);
}
printf("enter the character\n");
 ch=getche();
b[0]='d';
b[1]='l';
b[2]='e';
b[3]='s';
b[4]='t';
b[5]='x'; j=6;
while(i<n)
{
if(i==pos-1)
{
b[j]='d';
b[j+1]='l';
b[j+2]='e';
b[j+3]=ch; b[j+4]='d';
b[j+5]='l';
b[j+6]='e';
j=j+7;
}
if(a[i]=='d' && a[i+1]=='l' && a[i+2]=='e')
{
b[j]='d';
b[j+1]='l';
b[j+2]='e';
j=j+3;
}
b[j]=a[i]; i++;
j++;
}
b[j]='d';
b[j+1]='l';
b[j+2]='e';
b[j+3]='e';
b[j+4]='t';
b[j+5]='x';
b[j+6]='\0';
printf("\nframe after stuffing:\n");
printf("%s",b);
getch(); }
```

Output:


C:\Users\GRIET\Desktop\CodeBlocks\sample\bin\Debug\sample.exe

```
enter string
swathigopi
enter position
4
enter the character
i
frame after stuffing:
dlestxswadleidlethigopidleetx
Process returned 13 (0xD)   execution time : 62.310 s
Press any key to continue.
```

NAME OF THE EXPERIMENT: **1.c) Character Count**

OBJECTIVE: Implement the data link layer framing method.

RESOURCE: Code blocks

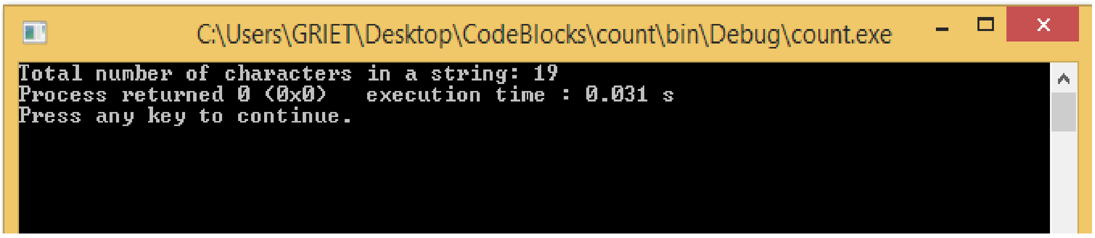
PROGRAM LOGIC:
The framing method To count the number of characters present in the string, we will iterate through the string and count the characters. In above example, total number of characters present in the string are 19.

Program for character count

```
#include <stdlib.h>
#include <string.h>
int main()
{
    char string[] = "The best of both worlds";
    int count = 0;
    //Counts each character except space
    for(int i = 0; i < strlen(string); i++) {
        if(string[i] != ' ')
            count++;
    }
    //Displays the total number of characters present in the given string
    printf("Total number of characters in a string: %d", count);

    return 0;
}
```

Output:

Total number of characters in a string: 19



```
Total number of characters in a string: 19
Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

NAME OF THE EXPERIMENT: 2.a) Simplex protocol (stop and wait protocol)

OBJECTIVE: Implement the data link layer framing method.

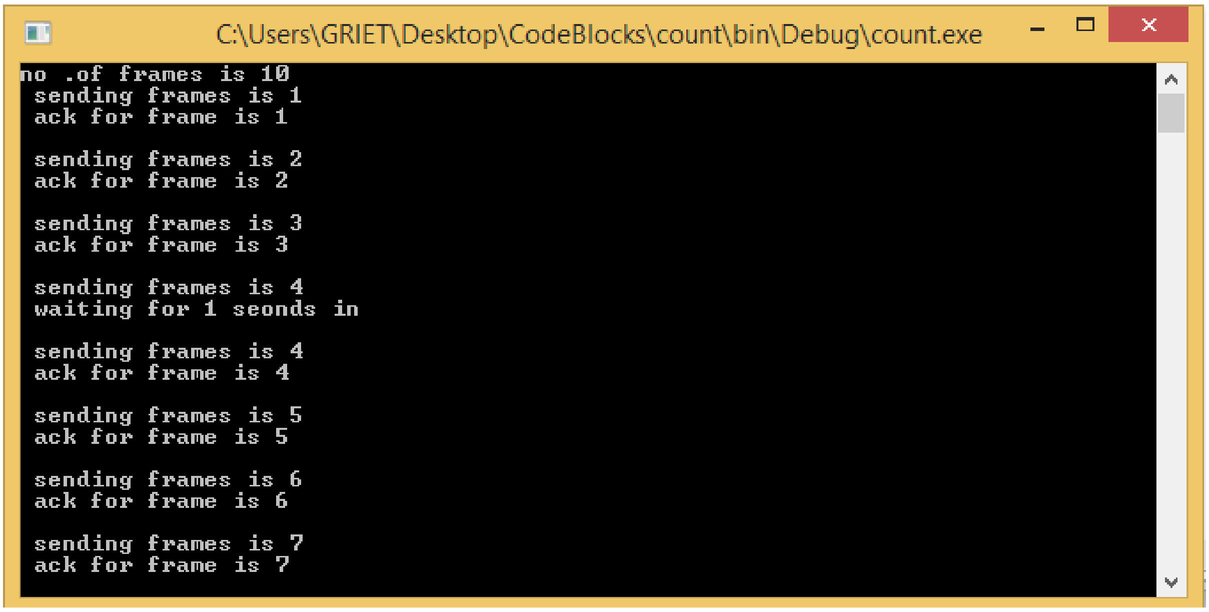
RESOURCE: Code blocks

PROGRAM LOGIC: Stop and wait protocol is a simple and reliable protocol for flow control. Stop and wait protocol is a data link layer protocol. In this protocol, the sender will not send the next packet to the receiver until the acknowledgment of the previous packet is received.

Program:

```
#include <stdio.h>
#include <stdlib.h>
void main()
{
   int i,j,n,x,x1=10,x2;
   n=10;
   i=1;j=1;
   printf("no .of frames is %d",n);
   getch();
   while(n>0)
   {
     printf("\n sending frames is %d",i);
     x=rand()%10;
     if(x%10==0)
     {
        for(x2=1;x2<2;x2++)
        {
           printf("\n waiting for %d seonds in \n ",x2);
           sleep(x2);
        }
        printf("\n sending frames is %d ",i);
        x=rand()%10;
     }
     printf("\n ack for frame is %d \n ",j);
     n=n-1;
     i++;
     j++;
   }
   printf("\n end of stop and wait protocol \n");
   getch();
}
```

Output:



```
no .of frames is 10
 sending frames is 1
 ack for frame is 1

 sending frames is 2
 ack for frame is 2

 sending frames is 3
 ack for frame is 3

 sending frames is 4
 waiting for 1 seonds in

 sending frames is 4
 ack for frame is 4

 sending frames is 5
 ack for frame is 5

 sending frames is 6
 ack for frame is 6

 sending frames is 7
 ack for frame is 7
```

NAME OF THE EXPERIMENT: 2.b) Sliding Window protocol(Go Back N)

OBJECTIVE: Implement the data link layer framing method.
RESOURCE: Code blocks

PROGRAM LOGIC: Go-Back-N is a data link layer protocol that uses a sliding window method. In this, if any frame is corrupted or lost, all subsequent frames have to be sent again.

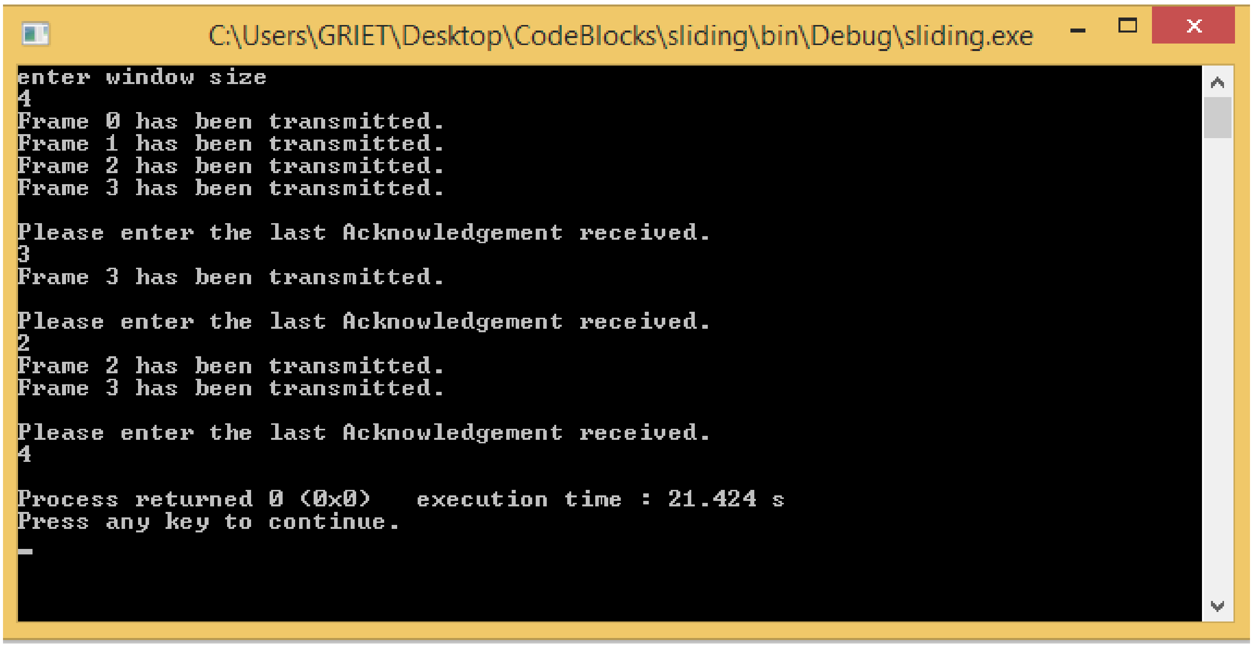
The size of the sender window is N in this protocol. For example, Go-Back-8, the size of the sender window, will be 8. The receiver window size is always 1.

If the receiver receives a corrupted frame, it cancels it. The receiver does not accept a corrupted frame. When the timer expires, the sender sends the correct frame again.

Program:

```
#include<stdio.h>
int main()
{
        int windowsize,sent=0,ack,i;
        printf("enter window size\n");
        scanf("%d",&windowsize);
        while(1)
        {
                for( i = 0; i < windowsize; i++)
                        {
                                printf("Frame %d has been transmitted.\n",sent);
                                sent++;
                                if(sent == windowsize)
                                        break;
                        }
                        printf("\nPlease enter the last Acknowledgement received.\n");
                        scanf("%d",&ack);
                        if(ack == windowsize)
                                break;
                        else
                                sent = ack;
        }
return 0;
}
```

Output:



C:\Users\GRIET\Desktop\CodeBlocks\sliding\bin\Debug\sliding.exe

```
enter window size
4
Frame 0 has been transmitted.
Frame 1 has been transmitted.
Frame 2 has been transmitted.
Frame 3 has been transmitted.

Please enter the last Acknowledgement received.
3
Frame 3 has been transmitted.

Please enter the last Acknowledgement received.
2
Frame 2 has been transmitted.
Frame 3 has been transmitted.

Please enter the last Acknowledgement received.
4

Process returned 0 (0x0)   execution time : 21.424 s
Press any key to continue.
```

NAME OF THE EXPERIMENT: 2.c) Sliding window protocol (selective repeat)

OBJECTIVE: Implement the data link layer framing method.

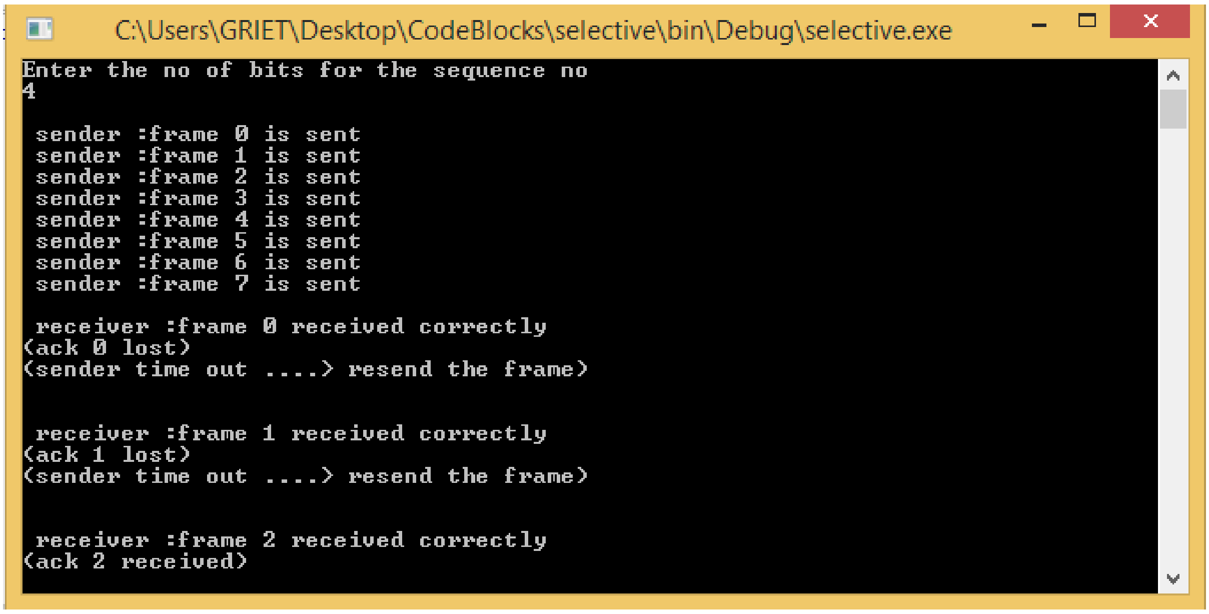
RESOURCE: Code blocks

PROGRAM LOGIC: Selective Repeat ARQ is also known as the Selective Repeat Automatic Repeat Request. It is a data link layer protocol that uses a sliding window method. The Go-back-N ARQ protocol works well if it has fewer errors. But if there is a lot of error in the frame, lots of bandwidth loss in sending the frames again. So, we use the Selective Repeat ARQ protocol. In this protocol, the size of the sender window is always equal to the size of the receiver window.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main()
{
   int f,n;
   printf("Enter the no of bits for the sequence no \n");
   scanf("%d",&n);
   f=pow(2,n-1);
   for(int j=0;j<f;j++)
   {
      printf("\n sender :frame %d is sent",j);
   }
   for(int i=0;i<f;i++)
   {
      printf("\n");
      int flag=rand()%2;
      if(!flag)
      {
         printf("\n receiver :frame %d received correctly \n(ack %d received) \n",i,i);
      }
      else
         {
           printf("\n receiver :frame %d received correctly \n(ack %d lost) \n",i,i);
         printf("(sender time out ....> resend the frame) \n");
      }
   }
printf("\n want to continue");
   return 0;
}
```

Output:


C:\Users\GRIET\Desktop\CodeBlocks\selective\bin\Debug\selective.exe

```
Enter the no of bits for the sequence no
4

 sender :frame 0 is sent
 sender :frame 1 is sent
 sender :frame 2 is sent
 sender :frame 3 is sent
 sender :frame 4 is sent
 sender :frame 5 is sent
 sender :frame 6 is sent
 sender :frame 7 is sent

 receiver :frame 0 received correctly
(ack 0 lost)
(sender time out ....> resend the frame)


 receiver :frame 1 received correctly
(ack 1 lost)
(sender time out ....> resend the frame)


 receiver :frame 2 received correctly
(ack 2 received)
```

NAME OF THE EXPERIMENT: 3a) Shortest Path routing protocol

OBJECTIVE: Implement the data link layer framing method.

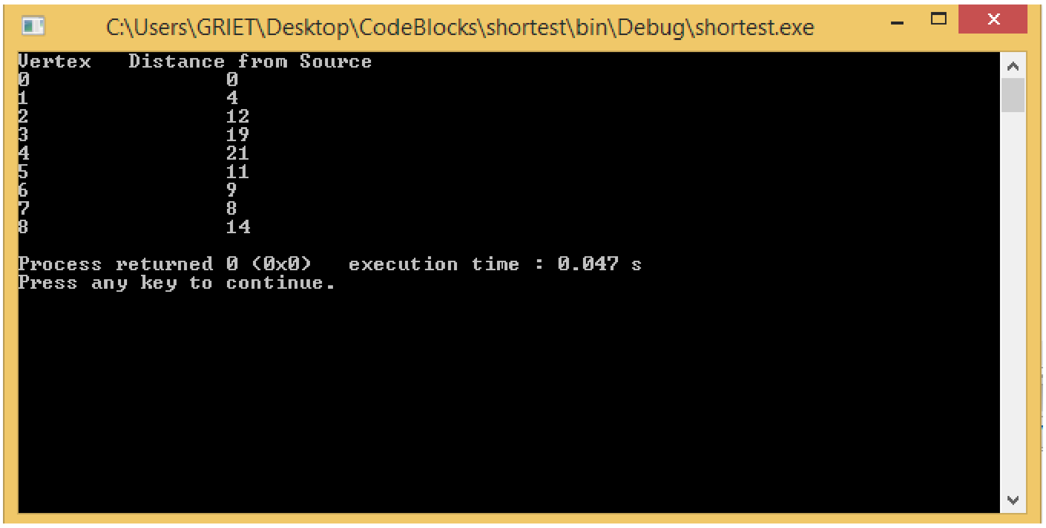
RESOURCE: Code blocks

PROGRAM LOGIC: Dijkstra shortest path (SP): This algorithm finds the shortest route from a given source to a destination in a graph. The route is a path whose cost is the least possible one. K-shortest path (K-SP): K-shortest-path algorithms find more than one route for each source and destination pair.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>
#define V 9
int minDistance(int dist[], int sptSet[])
{
   int min = INT_MAX, min_index;
   int v;
   for (v = 0; v < V; v++)
      if (sptSet[v] == 0 && dist[v] <= min)
         min = dist[v], min_index = v;
   return min_index;
}
void printSolution(int dist[], int n) {
   printf("Vertex   Distance from Source\n");
   int i;
   for (i = 0; i < V; i++)
      printf("%d \t\t %d\n", i, dist[i]);
}
void dijkstra(int graph[V][V], int src) {
   int dist[V];
   int sptSet[V];
   int i, count, v;
   for (i = 0; i < V; i++)
      dist[i] = INT_MAX, sptSet[i] = 0;
   dist[src] = 0;
   for (count = 0; count < V - 1; count++) {
      int u = minDistance(dist, sptSet);
      sptSet[u] = 1;
      for (v = 0; v < V; v++)
         if (!sptSet[v] && graph[u][v] && dist[u] != INT_MAX && dist[u]
               + graph[u][v] < dist[v])
            dist[v] = dist[u] + graph[u][v];
   }
   printSolution(dist, V);
}
int main() {
   /* Let us create the example graph discussed above */
   int graph[V][V] =  {{0, 4, 0, 0, 0, 0, 0, 8, 0},
                        {4, 0, 8, 0, 0, 0, 0, 11, 0},
                        {0, 8, 0, 7, 0, 4, 0, 0, 2},
                        {0, 0, 7, 0, 9, 14, 0, 0, 0},
                        {0, 0, 0, 9, 0, 10, 0, 0, 0},
                        {0, 0, 4, 0, 10, 0, 2, 0, 0},
                        {0, 0, 0, 14, 0, 2, 0, 1, 6},
                        {8, 11, 0, 0, 0, 0, 1, 0, 7},
                        {0, 0, 2, 0, 0, 0, 6, 7, 0}
                       };
   dijkstra(graph, 0);
   return 0;
}
```

Output:



C:\Users\GRIET\Desktop\CodeBlocks\shortest\bin\Debug\shortest.exe

```
Vertex   Distance from Source
0                0
1                4
2                12
3                19
4                21
5                11
6                9
7                8
8                14

Process returned 0 (0x0)   execution time : 0.047 s
Press any key to continue.
```

NAME OF THE EXPERIMENT: 3b)Distance Vector routing protocol

OBJECTIVE: Implement the data link layer framing method.

RESOURCE: Code blocks

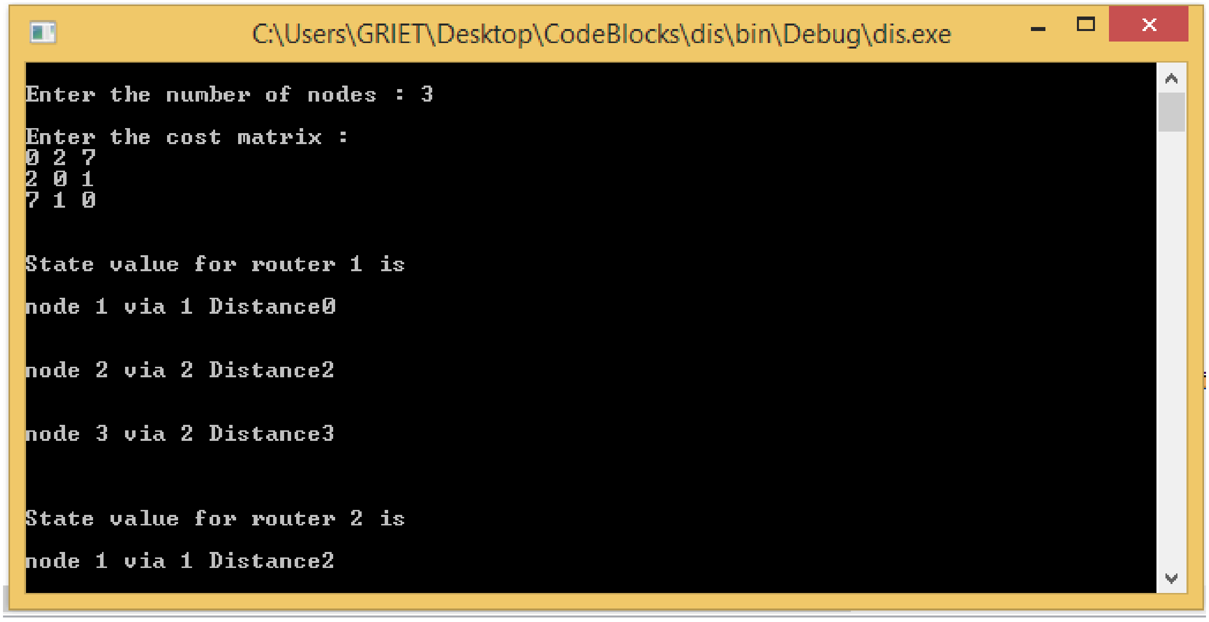
PROGRAM LOGIC: The distance vector routing algorithm is one of the most commonly used routing algorithms. It is a distributed algorithm, meaning that it is run on each router in the network. The algorithm works by each router sending updates to its neighbours about the best path to each destination.

Program:

```
#include <stdio.h>
#include <stdlib.h>
struct node  {
        unsigned dist[20];
```

```
    unsigned from[20];
}rt[10];
int main()  {
    int dmat[20][20],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\nState value for router %d is \n",i+1);
            for(j=0;j<n;j++)   {
                printf("\t\nnode %d via %d
Distance%d",j+1,rt[i].from[j]+1,rt[i].dist[j]);
    printf("\n\n");
        }
}
}
```

Output:



```
Enter the number of nodes : 3

Enter the cost matrix :
0 2 7
2 0 1
7 1 0


State value for router 1 is

node 1 via 1 Distance0


node 2 via 2 Distance2


node 3 via 2 Distance3


State value for router 2 is

node 1 via 1 Distance2
```

NAME OF THE EXPERIMENT: 3.C)Token Bucket algorithm

OBJECTIVE: Implement the data link layer framing method.

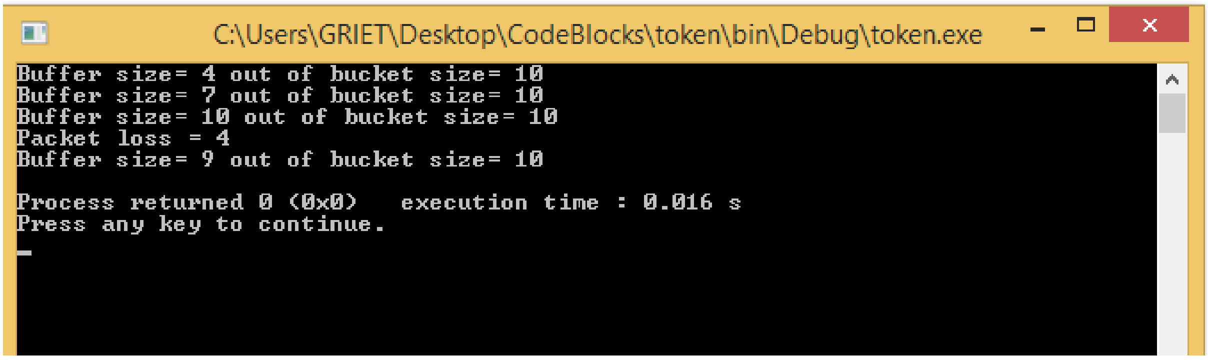
RESOURCE: Code blocks

PROGRAM LOGIC: Token bucket algorithm is one of the techniques for congestion control algorithms. When too many packets are present in the network it causes packet delay and loss of packet which degrades the performance of the system.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   int no_of_queries, storage, output_pkt_size;
   int input_pkt_size, bucket_size, size_left;
   storage = 0;
   no_of_queries = 4;
   bucket_size = 10;
   input_pkt_size = 4;
   output_pkt_size = 1;
   for (int i = 0; i < no_of_queries; i++)
   {
      size_left = bucket_size - storage;
      if (input_pkt_size <= size_left) {
         storage += input_pkt_size;
      }
      else {
         printf("Packet loss = %d\n", input_pkt_size);
      }
      printf("Buffer size= %d out of bucket size= %d\n",
           storage, bucket_size);
      storage -= output_pkt_size;
   }
   return 0;
}
```

Output:



```
Buffer size= 4 out of bucket size= 10
Buffer size= 7 out of bucket size= 10
Buffer size= 10 out of bucket size= 10
Packet loss = 4
Buffer size= 9 out of bucket size= 10

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

NAME OF THE EXPERIMENT: 4.a) DES algorithm

OBJECTIVE: Implement the **Data Encryption Standard is a symmetric-key algorithm.**

RESOURCE: Code blocks

PROGRAM LOGIC: The DES (Data Encryption Standard) algorithm is a symmetric-key block cipher created in the early 1970s by an IBM team and adopted by the National Institute of Standards and Technology (NIST). The algorithm takes the plain text in 64-bit blocks and converts them into ciphertext using 48-bit keys.

Program:

```
#include <stdio.h>
#include <stdlib.h>
int Original_key [64] = { // you can change key if required
0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0,
0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1,
1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0,
1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1
};
int Permutated_Choice1[56] = {
  57, 49, 41, 33, 25, 17,  9,
   1, 58, 50, 42, 34, 26, 18,
  10,  2, 59, 51, 43, 35, 27,
  19, 11,  3, 60, 52, 44, 36,
  63, 55, 47, 39, 31, 23, 15,
   7, 62, 54, 46, 38, 30, 22,
  14,  6, 61, 53, 45, 37, 29,
  21, 13,  5, 28, 20, 12,  4
};

int Permutated_Choice2[48] = {
  14, 17, 11, 24,  1,  5,
   3, 28, 15,  6, 21, 10,
  23, 19, 12,  4, 26,  8,
  16,  7, 27, 20, 13,  2,
  41, 52, 31, 37, 47, 55,
  30, 40, 51, 45, 33, 48,
  44, 49, 39, 56, 34, 53,
  46, 42, 50, 36, 29, 32
};

int Iintial_Permutation [64] = {
  58, 50, 42, 34, 26, 18, 10, 2,
  60, 52, 44, 36, 28, 20, 12, 4,
  62, 54, 46, 38, 30, 22, 14, 6,
  64, 56, 48, 40, 32, 24, 16, 8,
  57, 49, 41, 33, 25, 17,  9, 1,
  59, 51, 43, 35, 27, 19, 11, 3,
  61, 53, 45, 37, 29, 21, 13, 5,
  63, 55, 47, 39, 31, 23, 15, 7
};

int Final_Permutation[] =
{
  40, 8, 48, 16, 56, 24, 64, 32,
  39, 7, 47, 15, 55, 23, 63, 31,
  38, 6, 46, 14, 54, 22, 62, 30,
  37, 5, 45, 13, 53, 21, 61, 29,
  36, 4, 44, 12, 52, 20, 60, 28,
  35, 3, 43, 11, 51, 19, 59, 27,
  34, 2, 42, 10, 50, 18, 58, 26,
  33, 1, 41,  9, 49, 17, 57, 25
};
int P[] =
{
  16,  7, 20, 21,
  29, 12, 28, 17,
   1, 15, 23, 26,
   5, 18, 31, 10,
   2,  8, 24, 14,
  32, 27,  3,  9,
  19, 13, 30,  6,
  22, 11,  4, 25
};

int E[] =
{
  32,  1,  2,  3,  4,  5,
   4,  5,  6,  7,  8,  9,
   8,  9, 10, 11, 12, 13,
  12, 13, 14, 15, 16, 17,
  16, 17, 18, 19, 20, 21,
  20, 21, 22, 23, 24, 25,
  24, 25, 26, 27, 28, 29,
  28, 29, 30, 31, 32,  1
};

int S1[4][16] =
{
14,  4, 13,  1,  2, 15, 11,  8,  3, 10,  6, 12,  5,  9,  0,  7,
0, 15,  7,  4, 14,  2, 13,  1, 10,  6, 12, 11,  9,  5,  3,  8,
4,  1, 14,  8, 13,  6,  2, 11, 15, 12,  9,  7,  3, 10,  5,  0,
15, 12,  8,  2,  4,  9,  1,  7,  5, 11,  3, 14, 10,  0,  6, 13
};

int S2[4][16] =
{
15,  1,  8, 14,  6, 11,  3,  4,  9,  7,  2, 13, 12,  0,  5, 10,
3, 13,  4,  7, 15,  2,  8, 14, 12,  0,  1, 10,  6,  9, 11,  5,
0, 14,  7, 11, 10,  4, 13,  1,  5,  8, 12,  6,  9,  3,  2, 15,
13,  8, 10,  1,  3, 15,  4,  2, 11,  6,  7, 12,  0,  5, 14,  9
};

int S3[4][16] =
{
10,  0,  9, 14,  6,  3, 15,  5,  1, 13, 12,  7, 11,  4,  2,  8,
13,  7,  0,  9,  3,  4,  6, 10,  2,  8,  5, 14, 12, 11, 15,  1,
13,  6,  4,  9,  8, 15,  3,  0, 11,  1,  2, 12,  5, 10, 14,  7,
1, 10, 13,  0,  6,  9,  8,  7,  4, 15, 14,  3, 11,  5,  2, 12
};

int S4[4][16] =
{
7, 13, 14,  3,  0,  6,  9, 10,  1,  2,  8,  5, 11, 12,  4, 15,
13,  8, 11,  5,  6, 15,  0,  3,  4,  7,  2, 12,  1, 10, 14,  9,
10,  6,  9,  0, 12, 11,  7, 13, 15,  1,  3, 14,  5,  2,  8,  4,
3, 15,  0,  6, 10,  1, 13,  8,  9,  4,  5, 11, 12,  7,  2, 14
};

int S5[4][16] =
{
2, 12,  4,  1,  7, 10, 11,  6,  8,  5,  3, 15, 13,  0, 14,  9,
14, 11,  2, 12,  4,  7, 13,  1,  5,  0, 15, 10,  3,  9,  8,  6,
4,  2,  1, 11, 10, 13,  7,  8, 15,  9, 12,  5,  6,  3,  0, 14,
11,  8, 12,  7,  1, 14,  2, 13,  6, 15,  0,  9, 10,  4,  5,  3
};

int S6[4][16] =
{
12,  1, 10, 15,  9,  2,  6,  8,  0, 13,  3,  4, 14,  7,  5, 11,
10, 15,  4,  2,  7, 12,  9,  5,  6,  1, 13, 14,  0, 11,  3,  8,
9, 14, 15,  5,  2,  8, 12,  3,  7,  0,  4, 10,  1, 13, 11,  6,
4,  3,  2, 12,  9,  5, 15, 10, 11, 14,  1,  7,  6,  0,  8, 13
};

int S7[4][16]=
{
4, 11,  2, 14, 15,  0,  8, 13,  3, 12,  9,  7,  5, 10,  6,  1,
13,  0, 11,  7,  4,  9,  1, 10, 14,  3,  5, 12,  2, 15,  8,  6,
1,  4, 11, 13, 12,  3,  7, 14, 10, 15,  6,  8,  0,  5,  9,  2,
6, 11, 13,  8,  1,  4, 10,  7,  9,  5,  0, 15, 14,  2,  3, 12
};

int S8[4][16]=
{
13,  2,  8,  4,  6, 15, 11,  1, 10,  9,  3, 14,  5,  0, 12,  7,
1, 15, 13,  8, 10,  3,  7,  4, 12,  5,  6, 11,  0, 14,  9,  2,
7, 11,  4,  1,  9, 12, 14,  2,  0,  6, 10, 13, 15,  3,  5,  8,
2,  1, 14,  7,  4, 10,  8, 13, 15, 12,  9,  0,  3,  5,  6, 11
```

```
};

int shifts_for_each_round[16] = { 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1 };
int _56bit_key[56];
int _48bit_key[17][48];
int text_to_bits[99999], bits_size=0;
int Left32[17][32], Right32[17][32];
int EXPtext[48];
int XORtext[48];
int X[8][6];
int X2[32];
int R[32];
int chiper_text[64];
int encrypted_text[64];

int XOR(int a, int b) {
return (a ^ b);
}

void Dec_to_Binary(int n)
{
    int binaryNum[1000];
    int i = 0;
    while (n > 0) {
        binaryNum[i] = n % 2;
        n = n / 2;
        i++;
    }
    for (int j = i - 1; j >= 0; j--) {
text_to_bits[bits_size++] = binaryNum[j];
}
}
int F1(int i)
{
int r, c, b[6];
for (int j = 0; j < 6; j++)
b[j] = X[i][j];

r = b[0] * 2 + b[5];
c = 8 * b[1] + 4 * b[2] + 2 * b[3] + b[4];
if (i == 0)
return S1[r][c];
else if (i == 1)
return S2[r][c];
else if (i == 2)
return S3[r][c];
else if (i == 3)
return S4[r][c];
else if (i == 4)
return S5[r][c];
```

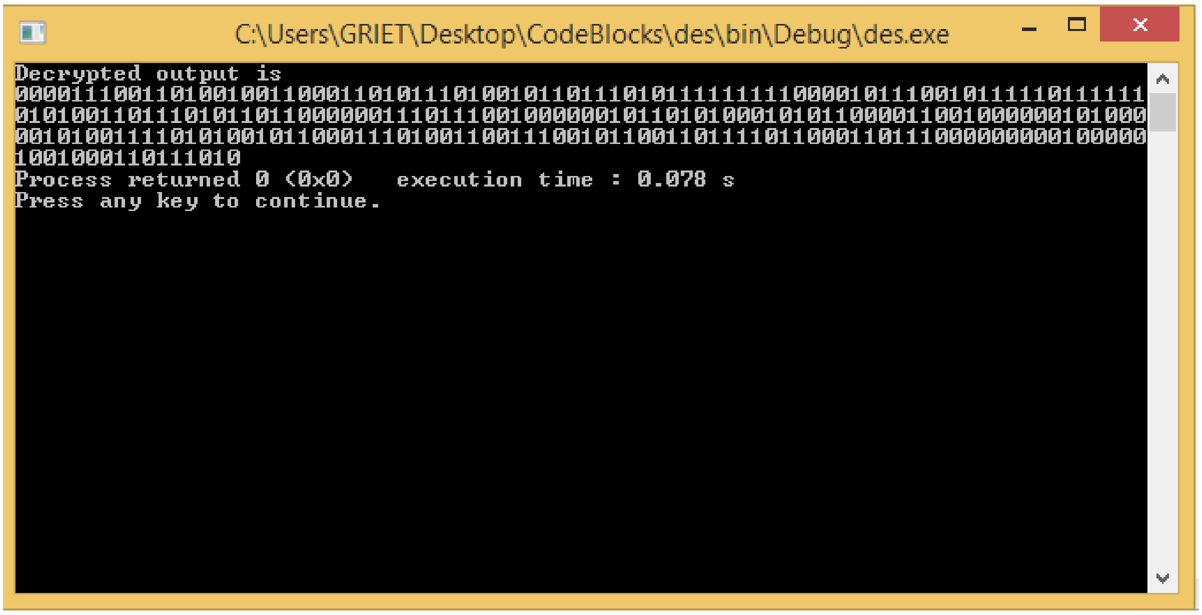
```
else if (i == 5)
return S6[r][c];
else if (i == 6)
return S7[r][c];
else if (i == 7)
return S8[r][c];
}
int PBox(int pos, int bit)
{
int i;
for (i = 0; i < 32; i++)
if (P[i] == pos + 1)
break;
R[i] = bit;
}
int ToBits(int value)
{
int k, j, m;
static int i;
if (i % 32 == 0)
i = 0;
for (j = 3; j >= 0; j--)
{
m = 1 << j;
k = value & m;
if (k == 0)
X2[3 - j + i] = '0' - 48;
else
X2[3 - j + i] = '1' - 48;
}
i = i + 4;
}
int SBox(int XORtext[])
{
int k = 0;
for (int i = 0; i < 8; i++)
for (int j = 0; j < 6; j++)
X[i][j] = XORtext[k++];

int value;
for (int i = 0; i < 8; i++)
{
value = F1(i);
ToBits(value);
}
}
void expansion_function(int pos, int bit)
{
for (int i = 0; i < 48; i++)
if (E[i] == pos + 1)
```

```
EXPtext[i] = bit;
}
void cipher(int Round, int mode)
{
for (int i = 0; i < 32; i++)
expansion_function(i, Right32[Round - 1][i]);
for (int i = 0; i < 48; i++)
{
if (mode == 0)
XORtext[i] = XOR(EXPtext[i], _48bit_key[Round][i]);
else
XORtext[i] = XOR(EXPtext[i], _48bit_key[17 - Round][i]);
}

SBox(XORtext);
for (int i = 0; i < 32; i++)
PBox(i, X2[i]);
for (int i = 0; i < 32; i++)
Right32[Round][i] = XOR(Left32[Round - 1][i], R[i]);
}
void finalPermutation(int pos, int bit)
{
int i;
for (i = 0; i < 64; i++)
if (Final_Permutation[i] == pos + 1)
break;
encrypted_text[i] = bit;
}
void Encrypt_each_64_bit (int plain_bits [])
{
int IP_result [64] , index=0;
for (int i = 0; i < 64; i++) {
IP_result [i] = plain_bits[ Iintial_Permutation[i] ];
}
for (int i = 0; i < 32; i++)
Left32[0][i] = IP_result[i];
for (int i = 32; i < 64; i++)
Right32[0][i - 32] = IP_result[i];
for (int k = 1; k < 17; k++)
{ // processing through all 16 rounds
cipher(k, 0);
for (int i = 0; i < 32; i++)
Left32[k][i] = Right32[k - 1][i]; // right part comes as it is to next round left part
}
for (int i = 0; i < 64; i++)
{ // 32bit swap as well as Final Inverse Permutation
if (i < 32)
chiper_text[i] = Right32[16][i];
else
chiper_text[i] = Left32[16][i - 32];
```

```
finalPermutation(i, chiper_text[i]);
}
for (int i = 0; i < 64; i++)
printf("%d", encrypted_text[i]);
}
void convert_Text_to_bits(char *plain_text){
for(int i=0;plain_text[i];i++){
int asci = plain_text[i];
Dec_to_Binary(asci);
}
}
void key56to48(int round, int pos, int bit)
{
int i;
for (i = 0; i < 56; i++)
if (Permutated_Choice2[i] == pos + 1)
break;
_48bit_key[round][i] = bit;
}
int key64to56(int pos, int bit)
{
int i;
for (i = 0; i < 56; i++)
if (Permutated_Choice1[i] == pos + 1)
break;
_56bit_key[i] = bit;
}
void key64to48(int key[])
{
int k, backup[17][2];
int CD[17][56];
int C[17][28], D[17][28];

for (int i = 0; i < 64; i++)
key64to56(i, key[i]);

for (int i = 0; i < 56; i++)
if (i < 28)
C[0][i] = _56bit_key[i];
else
D[0][i - 28] = _56bit_key[i];
for (int x = 1; x < 17; x++)
{
int shift = shifts_for_each_round[x - 1];
for (int i = 0; i < shift; i++)
backup[x - 1][i] = C[x - 1][i];
for (int i = 0; i < (28 - shift); i++)
C[x][i] = C[x - 1][i + shift];
k = 0;
for (int i = 28 - shift; i < 28; i++)
```

```
C[x][i] = backup[x - 1][k++];
for (int i = 0; i < shift; i++)
backup[x - 1][i] = D[x - 1][i];
for (int i = 0; i < (28 - shift); i++)
D[x][i] = D[x - 1][i + shift];
k = 0;
for (int i = 28 - shift; i < 28; i++)
D[x][i] = backup[x - 1][k++];
}
for (int j = 0; j < 17; j++)
{
for (int i = 0; i < 28; i++)
CD[j][i] = C[j][i];
for (int i = 28; i < 56; i++)
CD[j][i] = D[j][i - 28];
}
for (int j = 1; j < 17; j++)
for (int i = 0; i < 56; i++)
key56to48(j, i, CD[j][i]);
}
int main(){
char plain_text[] = "tomarrow we wiil be declaring war";
convert_Text_to_bits(plain_text);
key64to48(Original_key); // it creates all keys for all rounds
int _64bit_sets = bits_size/64;
printf("Decrypted output is\n");
for(int i=0;i<= _64bit_sets ;i++) {
Encrypt_each_64_bit (text_to_bits + 64*i);
}
return 0;
}
```

Output:



```
Decrypted output is
00001110011010010011000110101110100101101110101111111111000010111001011111011111
01010011011101011011000000111011100100000010110101000101011000011001000000101000
00101001111010100101100011101001100111001011001101111011000110111000000000100000
10010000110111010
Process returned 0 (0x0)   execution time : 0.078 s
Press any key to continue.
```

NAME OF THE EXPERIMENT: 4.b) RSA algorithm

OBJECTIVE: Implement the RSA asymmetric cryptographic algorithm

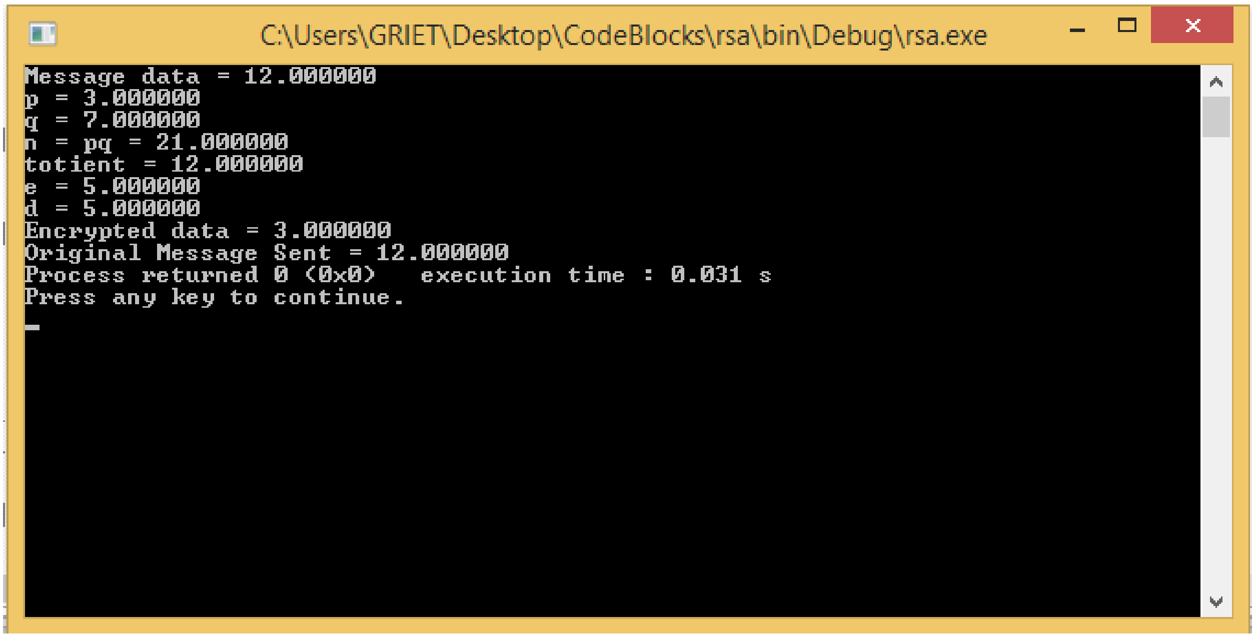
RESOURCE: Code blocks

PROGRAM LOGIC: RSA is a widely used cryptographic algorithm that was first introduced in 1977. It uses public and private key pairs to encrypt and decrypt data. Though RSA can be used in several applications, its computational complexity makes it unsuitable for encrypting large messages or files.

Program:

```
#include <stdio.h>
#include <stdlib.h>
#include<math.h>
//to find gcd
int gcd(int a, int h)
{
    int temp;
    while(1)
    {
        temp = a%h;
        if(temp==0)
        return h;
        a = h;
        h = temp;
    }
}
int main()
{
    //2 random prime numbers
    double p = 3;
    double q = 7;
    double n=p*q;
    double count;
    double totient = (p-1)*(q-1);

    //public key
    //e stands for encrypt
    double e=2;
    //for checking co-prime which satisfies e>1
    while(e<totient){
    count = gcd(e,totient);
    if(count==1)
        break;
    else
        e++;
    }
    //private key
    //d stands for decrypt
    double d;
    //k can be any arbitrary value
    double k = 2;
    //choosing d such that it satisfies d*e = 1 + k * totient
    d = (1 + (k*totient))/e;
    double msg = 12;
    double c = pow(msg,e);
    double m = pow(c,d);
    c=fmod(c,n);
    m=fmod(m,n);
    printf("Message data = %lf",msg);
    printf("\np = %lf",p);
    printf("\nq = %lf",q);
    printf("\nn = pq = %lf",n);
    printf("\ntotient = %lf",totient);
    printf("\ne = %lf",e);
    printf("\nd = %lf",d);
    printf("\nEncrypted data = %lf",c);
    printf("\nOriginal Message Sent = %lf",m);
    return 0;
}
```

Output:

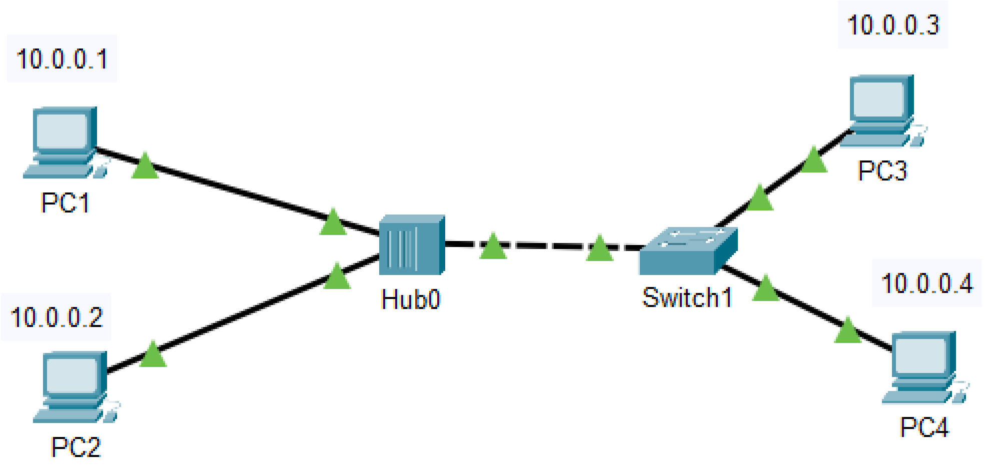


C:\Users\GRIET\Desktop\CodeBlocks\rsa\bin\Debug\rsa.exe

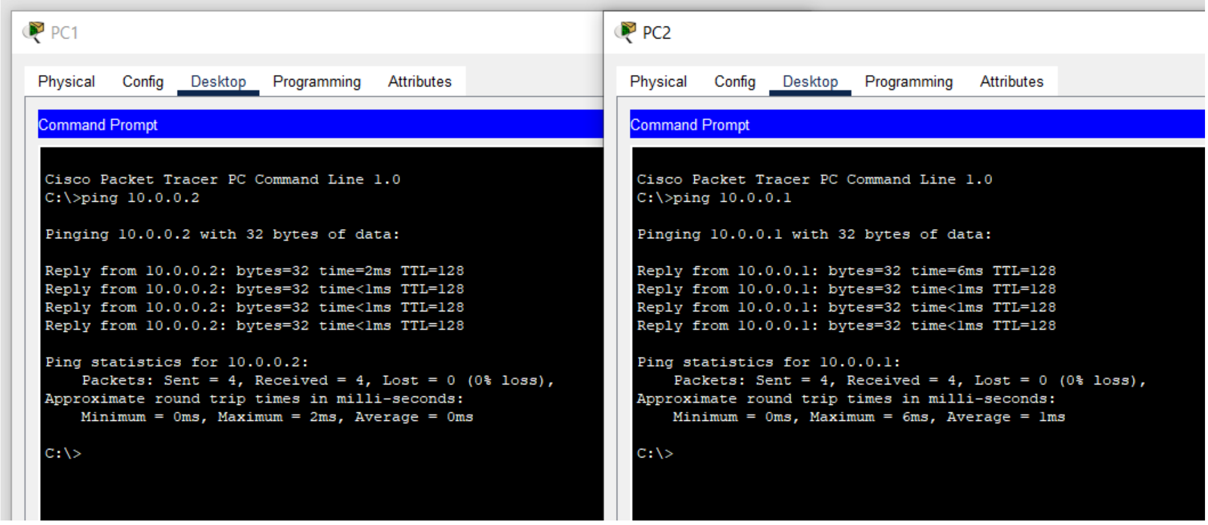
```
Message data = 12.000000
p = 3.000000
q = 7.000000
n = pq = 21.000000
totient = 12.000000
e = 5.000000
d = 5.000000
Encrypted data = 3.000000
Original Message Sent = 12.000000
Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

Task 5 :

a). Configure network devices, such as hubs and switches within a network topology using Packet Tracer software.



Configure the network systems with the above-mentioned IP address and then ping from other systems to test the flow of packets.



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=2ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

C:\>
```

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=6ms TTL=128
Reply from 10.0.0.1: bytes=32 time<1ms TTL=128
Reply from 10.0.0.1: bytes=32 time<1ms TTL=128
Reply from 10.0.0.1: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 6ms, Average = 1ms

C:\>
```

b).Construct a single LAN and understand the concepts and operation of ARP.

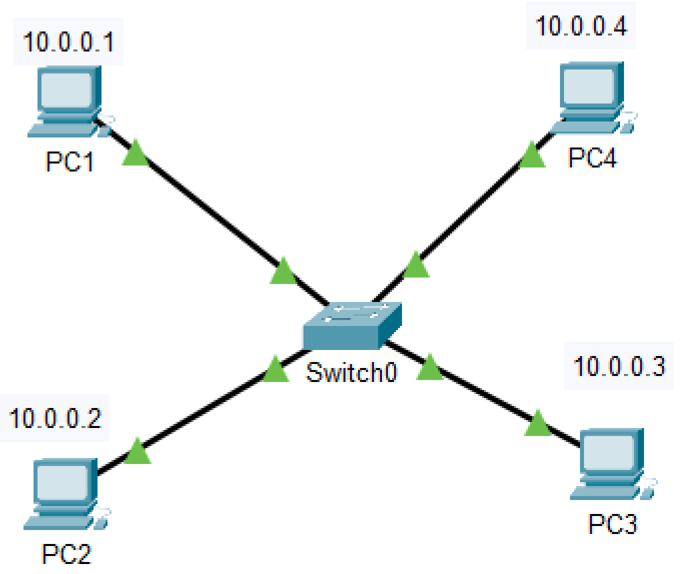
What is ARP?

The **Address Resolution Protocol** (**ARP**) is a communication protocol used for discovering the physical address, such as a MAC address, associated with a given internet layer address, typically an IPv4 address.

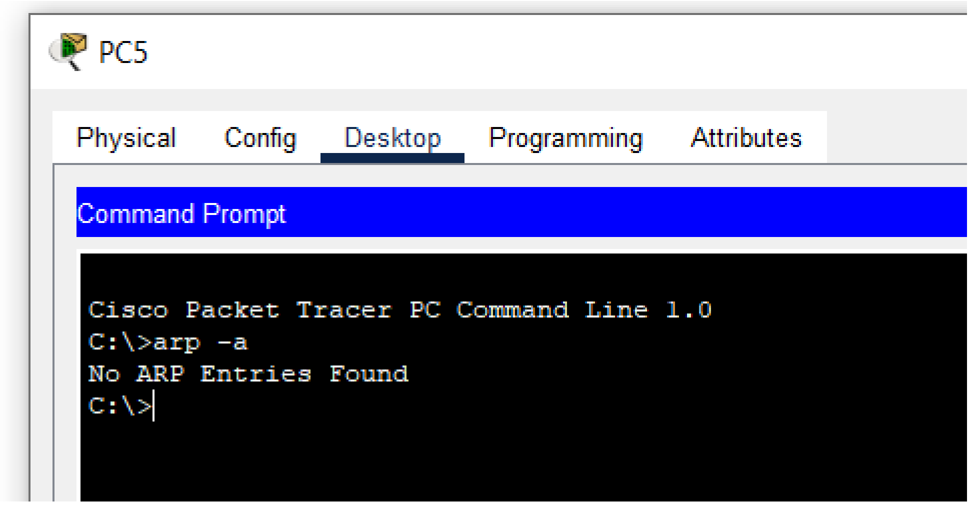
How ARP Works?

Imagine a device that wants to communicate with others over the internet. What does ARP do? It broadcast a packet to all the devices of the source network. The devices of the network peel the header of the data link layer from the **Protocol Data Unit (PDU)** called frame and transfer the packet to the network layer (layer 3 of OSI) where the network ID of the packet is validated with the destination IP's network ID of the packet and if it's equal then it responds to the source with the MAC address of the destination, else the packet reaches the gateway of the network and broadcasts packet to the devices it is connected with and validates their network ID. The above process continues till the second last network device in the path reaches the destination where it gets validated and ARP, in turn, responds with the destination MAC address.

Below topology illustrates 4 PCs with IP address ranging from 10.0.0.1- 10.0.0.4

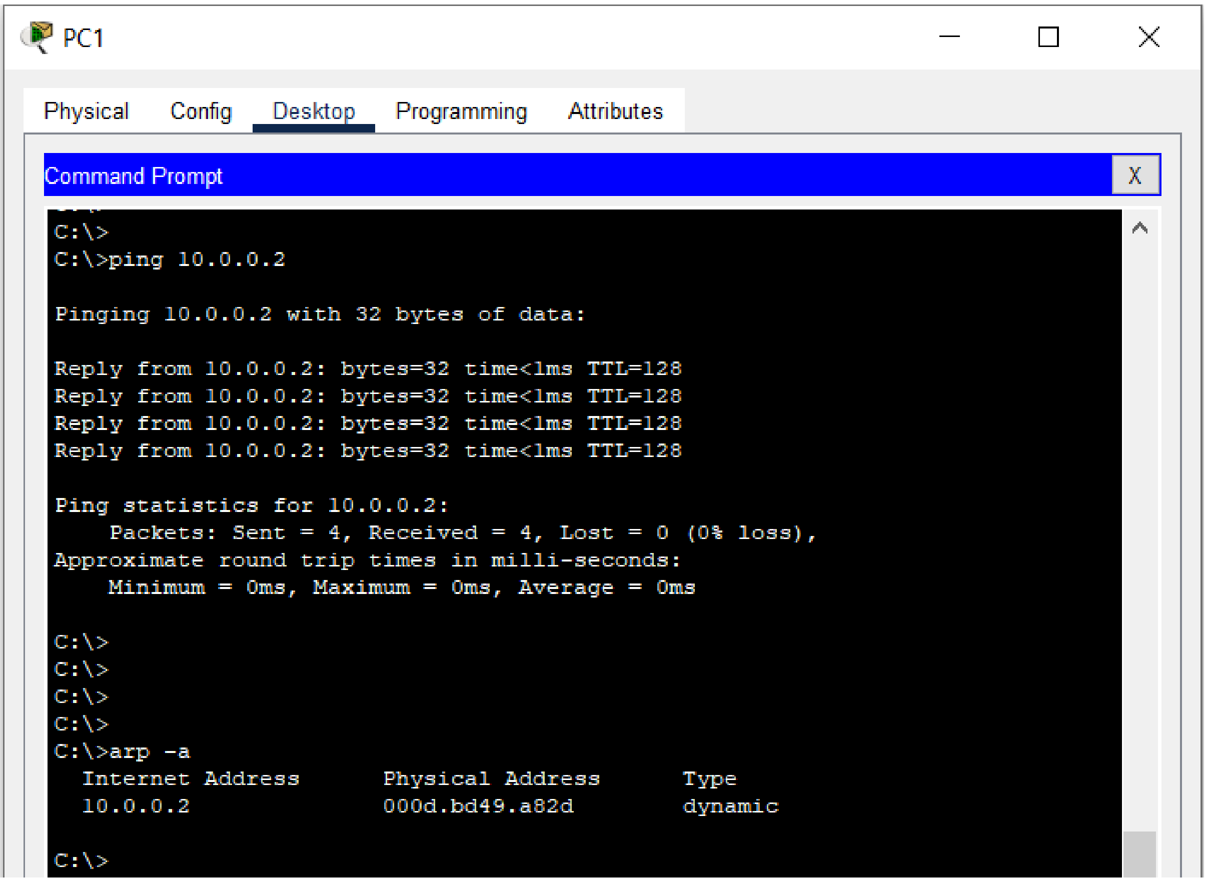


Initially if we look into ARP table of PC 1 there are no entries



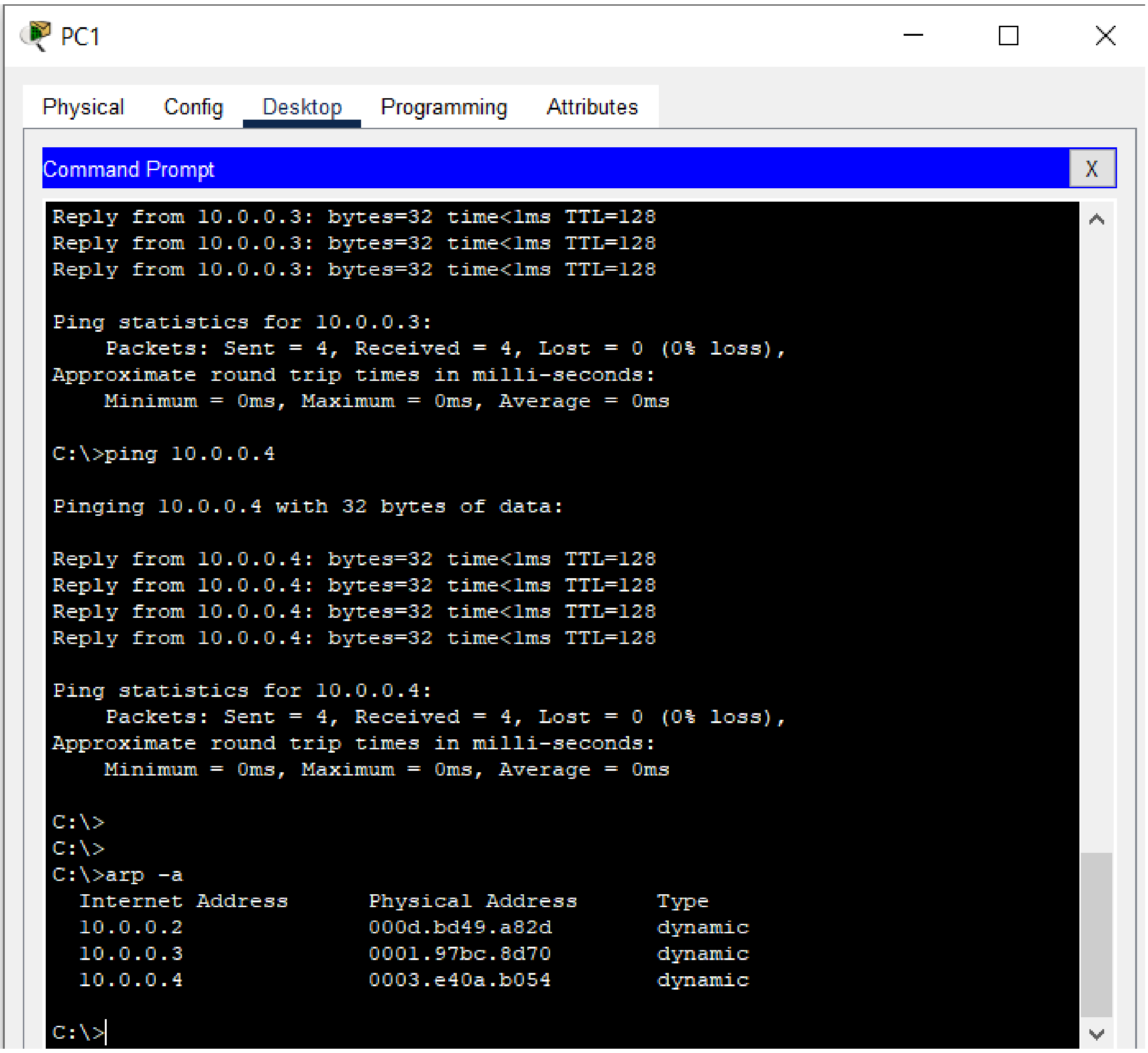
```
Cisco Packet Tracer PC Command Line 1.0
C:\>arp -a
No ARP Entries Found
C:\>
```

Now ping to PC2 from PC1 as illustrated below. And check the arp rable.



```
C:\>
C:\>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
C:\>
C:\>
C:\>
C:\>arp -a
  Internet Address      Physical Address      Type
  10.0.0.2              000d.bd49.a82d        dynamic

C:\>
```

Now from PC1 ping PC3 and PC4 and watch the arp tables



```
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 10.0.0.4

Pinging 10.0.0.4 with 32 bytes of data:

Reply from 10.0.0.4: bytes=32 time<1ms TTL=128
Reply from 10.0.0.4: bytes=32 time<1ms TTL=128
Reply from 10.0.0.4: bytes=32 time<1ms TTL=128
Reply from 10.0.0.4: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
C:\>
C:\>arp -a
  Internet Address      Physical Address      Type
  10.0.0.2              000d.bd49.a82d        dynamic
  10.0.0.3              0001.97bc.8d70        dynamic
  10.0.0.4              0003.e40a.b054        dynamic

C:\>
```

Now you can ping from other systems and see how the ARP tables are constructed.

TASK 6A: a). Configure and implementation of a Switch within a Network using Packet Tracer..

What is a network switch, and how does it work?

The Switch is a network device that is used to segment the networks into different subnetworks called subnets or LAN segments. It is responsible for filtering and forwarding the packets between LAN segments based on MAC address.

Switches have many ports, and when data arrives at any port, the destination address is examined first and some checks are also done and then it is processed to the devices. Different types of communication are supported here like unicast, multicast, and broadcast communication.

Modes of operation:

Mode	Purpose	Prompt	Command to enter	Command to exit
User EXEC	Allow you to connect with remote devices, perform basic tests, temporary change terminal setting and list system information	Router >	Default mode after booting. Login with password, if configured.	Use **exit** command
Privileged EXEC	Allow you to set operating parameters. It also includes high level testing and list commands like show, copy and debug.	Router #	Use **enable** command from user exec mode	Use **exit** command
Global Configuration	Contain commands those affect the entire system	Router(config)#	Use **configure terminal** command from privileged exec mode	Use **exit** command
Interface Configuration	Contain commands those modify the operation of an interface	Router(config-if)#	Use **interface *type number*** command from global configuration mode	Use **exit** command to return in global configuration mode
Sub-Interface Configuration	Configure or modify the virtual interface created from physical interface	Router(config-subif)	Use **interface *type sub interface*** number command from global configuration mode or interface configure mode	Use **exit** to return in previous mode. Use **end** command to return in privileged exec mode.

Step 1. Open the packet tracer desktop and take a switch (PT-Switch) from the devices.

Step 2: Configure the Host name of the swicth0.

- Click on switch0 and go to Command Line Interface.
- Then change the hostname to “sh”

Command:

```
switch>
switch>en
switch#conf t
switch(config)#hostname sh
```

Step 3: Set a message of the day (MOTD) banner for the users.

Command:

```
sh(config)#banner motd $
```

Step 4: Set up line control password and enable secret password.

To configure the Line Control password and Enable secret follow the below commands:

```
sh#conf t
sh(config)#
sh(config)#line con 0
sh(config-line)#password griet123
sh(config-line)#login
sh(config-line)#exit
sh(config)#enable secret griet12345
sh(config)#service password-encryption  // encrypts the password
sh(config)#exit
```

Step 5: Verify the password

- When you try to log in first, it will ask for the **line control password.**
- Then, to configure the terminal it will ask to **enable a secret password.**

To save the run configuration to startup file use the below command:

Command:

```
sh#copy run startup-config   (OR)    write
```

sh# no ip domain-lookup // used to prevent the router from trying to resolve incorrectly pasted commands in the cli by sending out a DNS query.

Select the switch – goto cli mode and type the below configuration commands.

```
Switch>
Switch>enable
Switch#config terminal
Switch(config)#hostname sh
sh(config)#banner motd #Warning Unauthorised access is prohibited#

sh(config)#line con 0
sh(config-line)#password griet1234
sh(config-line)#login
sh(config-line)#exit

sh(config)#enable secret griet5678
sh(config)#service password-encryption

sh(config)#no ip domain-lookup

sh#copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]


sh#show start
sh#show startup-config
Using 1238 bytes
!
version 15.0
no service timestamps log datetime msec
no service timestamps debug datetime msec
service password-encryption
!
hostname sh
!
enable secret 5 $1$mERr$vyUGBRk3bfoMV8qV.wJrB0
!
!
!
no ip domain-lookup
!
!
!
spanning-tree mode pvst
spanning-tree extend system-id
!
interface FastEthernet0/1
!
```

```
interface FastEthernet0/2
!
interface FastEthernet0/3
!
interface FastEthernet0/4
!
interface FastEthernet0/5

!< deleted some part>
!
interface FastEthernet0/20
!
interface FastEthernet0/21
!
interface FastEthernet0/22
!
interface FastEthernet0/23
!
interface FastEthernet0/24
!
interface GigabitEthernet0/1
!
interface GigabitEthernet0/2
!
interface Vlan1
no ip address
shutdown
!
banner motd ^CWarning Unauthorised access is prohibited^C
!
!
!
line con 0
password 7 08265E470C0D5445415F
login
!
line vty 0 4
login
line vty 5 15
login
!
!
!
!
End
```

TASK 6B : Learn and Implement basic commands.

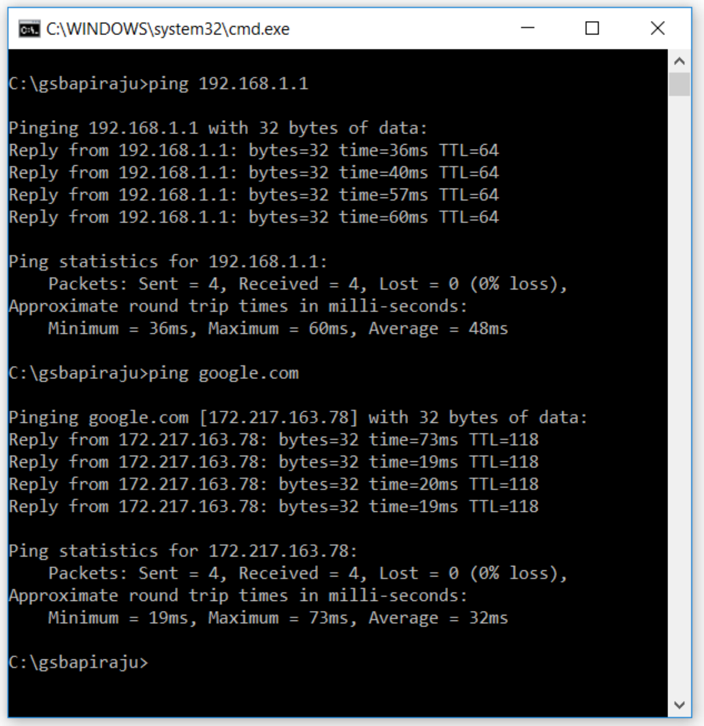
1. Ping

Ping is most commonly used network tool used to test the connection between the source and destination host.

Ping command uses Internet Control Message Protocol (ICMP) to send an echo packet from the source host to a destination host and listen to the response. If the source host receives a response from the destination host, this host is reachable. If not there is a connection error.

Using Ping command the user can identify in which area the connection problem is there, is it local or outside their LAN.

Ex: You can ping either by using the IP address or by the website name or URL. In the below example I pinged to my wireless router with its IP Address and google.com by its domain name.

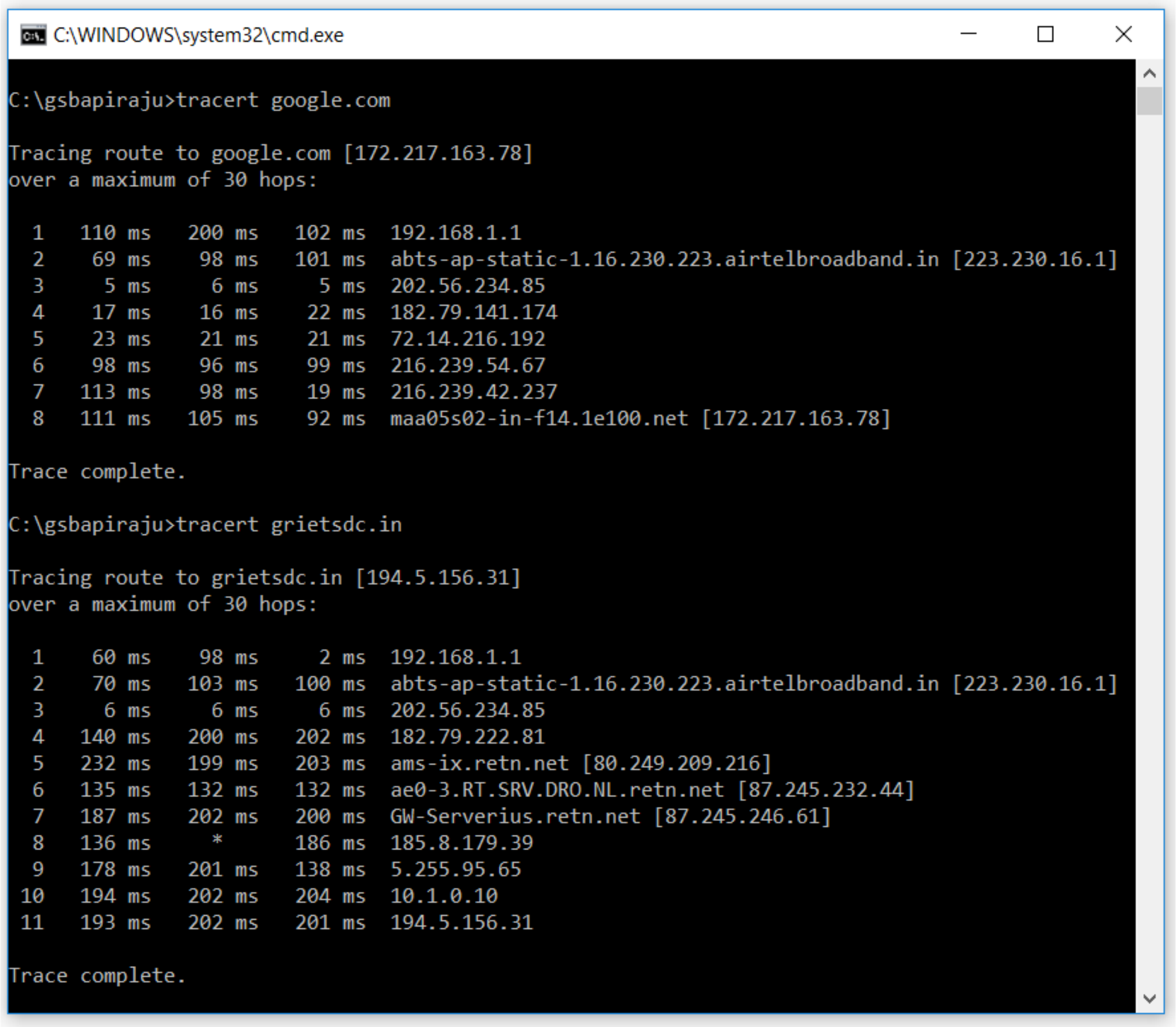


```
C:\gsbapiraju>ping 192.168.1.1

Pinging 192.168.1.1 with 32 bytes of data:
Reply from 192.168.1.1: bytes=32 time=36ms TTL=64
Reply from 192.168.1.1: bytes=32 time=40ms TTL=64
Reply from 192.168.1.1: bytes=32 time=57ms TTL=64
Reply from 192.168.1.1: bytes=32 time=60ms TTL=64

Ping statistics for 192.168.1.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 36ms, Maximum = 60ms, Average = 48ms

C:\gsbapiraju>ping google.com

Pinging google.com [172.217.163.78] with 32 bytes of data:
Reply from 172.217.163.78: bytes=32 time=73ms TTL=118
Reply from 172.217.163.78: bytes=32 time=19ms TTL=118
Reply from 172.217.163.78: bytes=32 time=20ms TTL=118
Reply from 172.217.163.78: bytes=32 time=19ms TTL=118

Ping statistics for 172.217.163.78:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 19ms, Maximum = 73ms, Average = 32ms

C:\gsbapiraju>
```

2. Tracert/traceroute.

Ping is a basic tool to check the basic connectivity. But if you want to identify the complete path from the source node to the destination node than tracert/traceroute utility is very useful.

The tracert utility for windows and traceroute utility for Linux gives you the entire path, including the number of hops packet travelled.

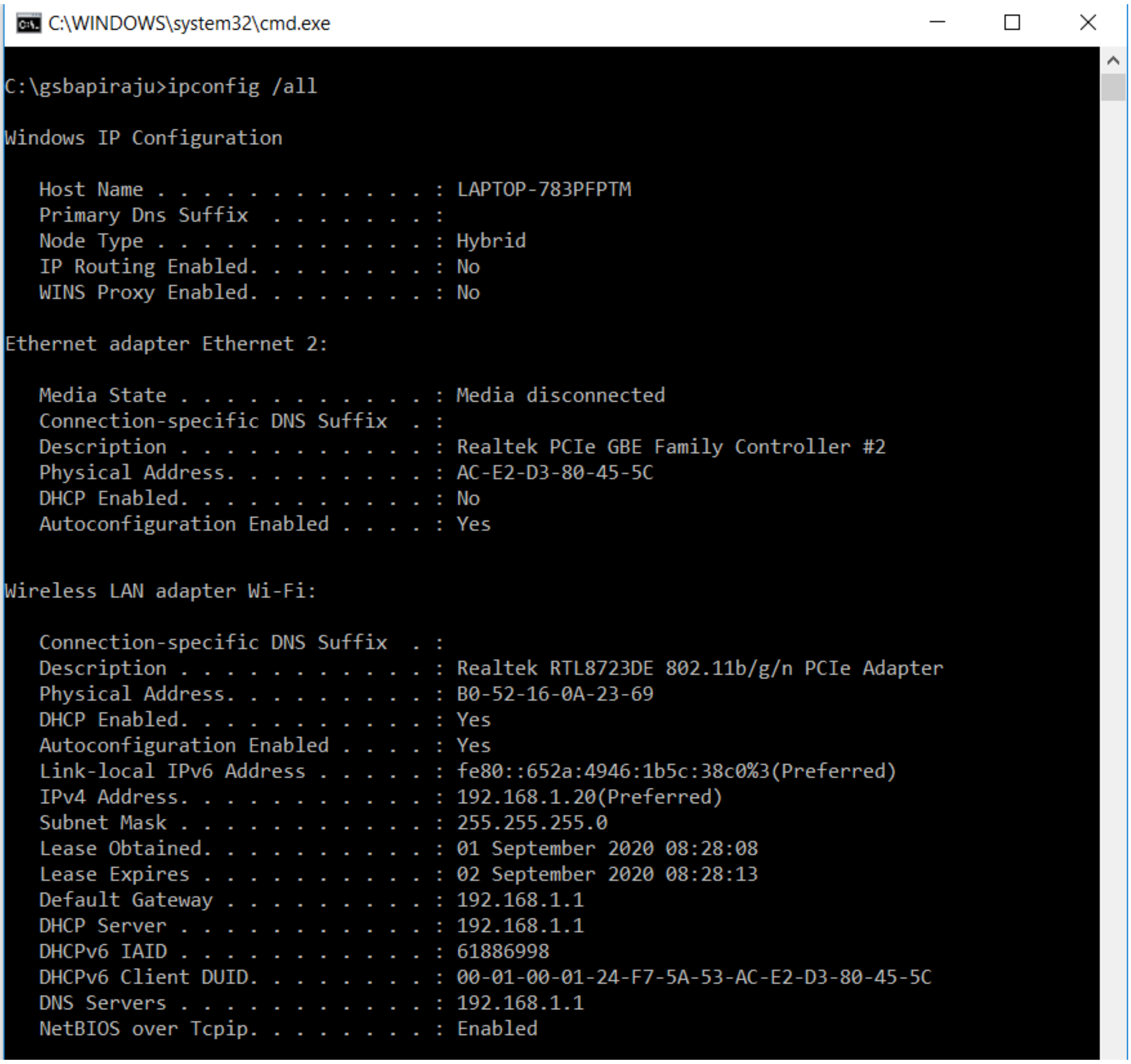


```
C:\gsbapiraju>tracert google.com

Tracing route to google.com [172.217.163.78]
over a maximum of 30 hops:

  1   110 ms   200 ms   102 ms  192.168.1.1
  2    69 ms    98 ms   101 ms  abts-ap-static-1.16.230.223.airtelbroadband.in [223.230.16.1]
  3     5 ms     6 ms     5 ms  202.56.234.85
  4    17 ms    16 ms    22 ms  182.79.141.174
  5    23 ms    21 ms    21 ms  72.14.216.192
  6    98 ms    96 ms    99 ms  216.239.54.67
  7   113 ms    98 ms    19 ms  216.239.42.237
  8   111 ms   105 ms    92 ms  maa05s02-in-f14.1e100.net [172.217.163.78]

Trace complete.

C:\gsbapiraju>tracert grietsdc.in

Tracing route to grietsdc.in [194.5.156.31]
over a maximum of 30 hops:

  1    60 ms    98 ms     2 ms  192.168.1.1
  2    70 ms   103 ms   100 ms  abts-ap-static-1.16.230.223.airtelbroadband.in [223.230.16.1]
  3     6 ms     6 ms     6 ms  202.56.234.85
  4   140 ms   200 ms   202 ms  182.79.222.81
  5   232 ms   199 ms   203 ms  ams-ix.retn.net [80.249.209.216]
  6   135 ms   132 ms   132 ms  ae0-3.RT.SRV.DRO.NL.retn.net [87.245.232.44]
  7   187 ms   202 ms   200 ms  GW-Serverius.retn.net [87.245.246.61]
  8   136 ms     *      186 ms  185.8.179.39
  9   178 ms   201 ms   138 ms  5.255.95.65
 10   194 ms   202 ms   204 ms  10.1.0.10
 11   193 ms   202 ms   201 ms  194.5.156.31

Trace complete.
```

3. Ipconfig

Ipconfig is one of the most important tool for system admins for troubleshooting networking issue. It is a command-line tool that shows the current TCP/IP configuration of the installed networking stack of a computer connected to a network.

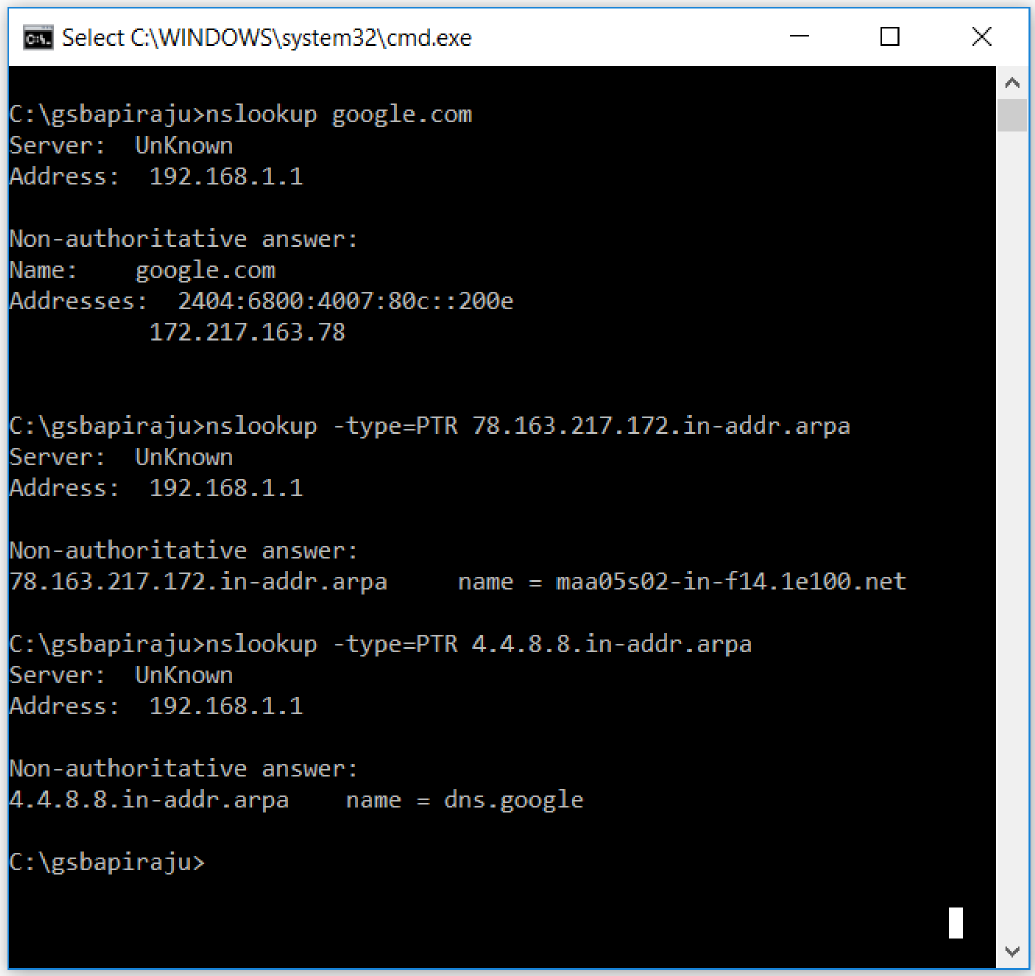
This tool includes a number of switches to perform different actions. In the below example I am using /all which Produces a detailed configuration report for all interfaces. You can observe the 48 bit MAC address, IPaddress, DHCP details etc**.**



```
C:\gsbapiraju>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : LAPTOP-783PFPTM
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Ethernet 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Realtek PCIe GBE Family Controller #2
   Physical Address. . . . . . . . . : AC-E2-D3-80-45-5C
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes


Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Realtek RTL8723DE 802.11b/g/n PCIe Adapter
   Physical Address. . . . . . . . . : B0-52-16-0A-23-69
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::652a:4946:1b5c:38c0%3(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.1.20(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : 01 September 2020 08:28:08
   Lease Expires . . . . . . . . . . : 02 September 2020 08:28:13
   Default Gateway . . . . . . . . . : 192.168.1.1
   DHCP Server . . . . . . . . . . . : 192.168.1.1
   DHCPv6 IAID . . . . . . . . . . . : 61886998
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-24-F7-5A-53-AC-E2-D3-80-45-5C
   DNS Servers . . . . . . . . . . . : 192.168.1.1
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

4. Nslookup

Some of the most common networking issues revolve around issues with Dynamic Name System (DNS) address resolution issues. nslookup or "name server lookup" is a network administration command-line tool used for querying the Domain Name System to obtain domain name or IP address mapping, or other DNS records. This utility can be used to lookup the specific IP address(es) associated with a domain name. If this utility is unable to resolve this information, there is a DNS issue**.**

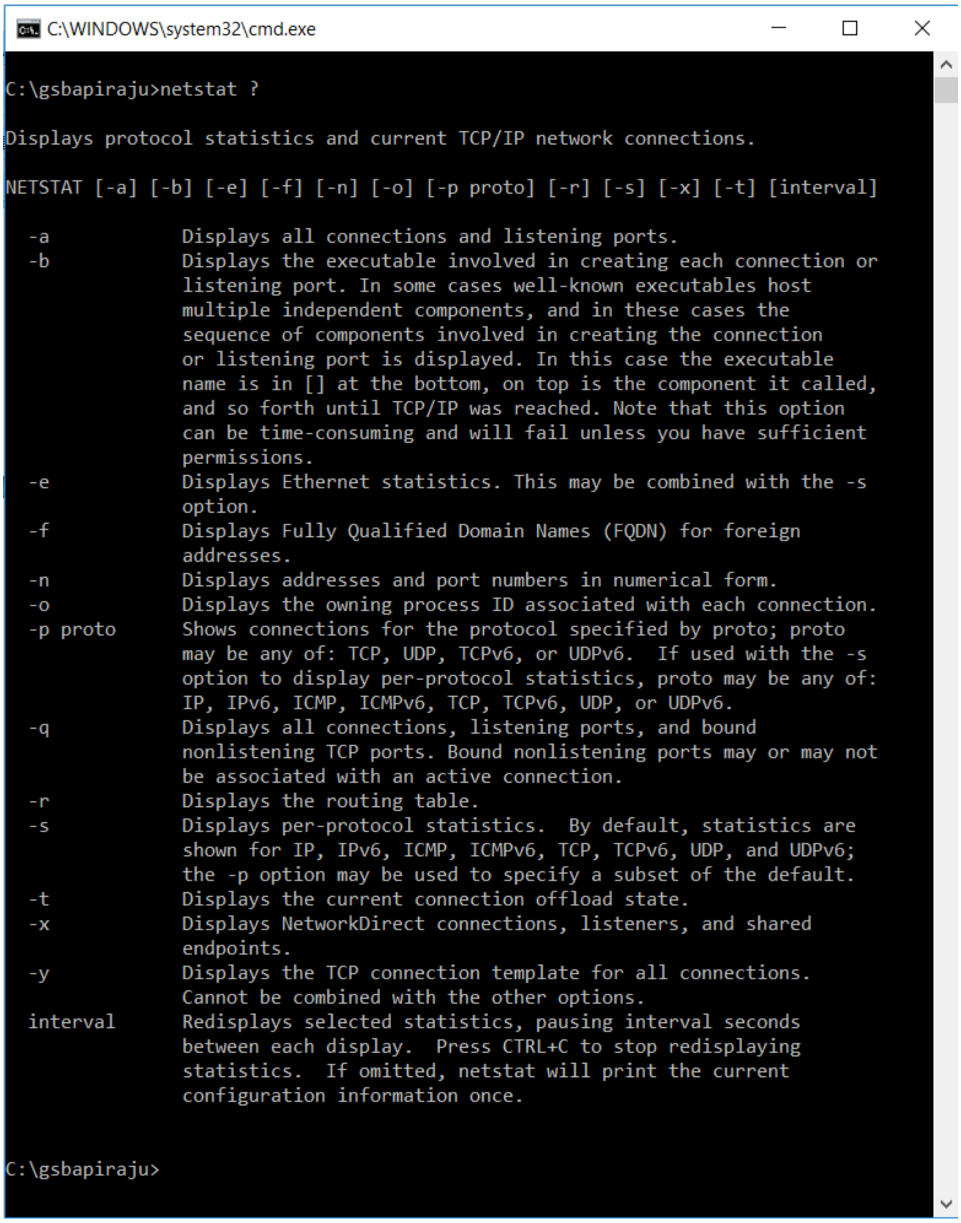


```
C:\gsbapiraju>nslookup google.com
Server:  UnKnown
Address:  192.168.1.1

Non-authoritative answer:
Name:    google.com
Addresses:  2404:6800:4007:80c::200e
          172.217.163.78


C:\gsbapiraju>nslookup -type=PTR 78.163.217.172.in-addr.arpa
Server:  UnKnown
Address:  192.168.1.1

Non-authoritative answer:
78.163.217.172.in-addr.arpa     name = maa05s02-in-f14.1e100.net

C:\gsbapiraju>nslookup -type=PTR 4.4.8.8.in-addr.arpa
Server:  UnKnown
Address:  192.168.1.1

Non-authoritative answer:
4.4.8.8.in-addr.arpa    name = dns.google

C:\gsbapiraju>
```

A typical DNS lookup is used to determine which IP address is associated with a hostname. A reverse DNS lookup is used for the opposite, to determine which hostname is associated with an IP address. Sometimes reverse DNS lookups are required for diagnostic purposes.

5. Netstat.

Netstat (*network statistics)* is a program that's controlled via commands issued in the command line. It delivers basic statistics on all network activities and informs users on which **ports and addresses** the corresponding connections (TCP, UDP) are running and which ports are open for tasks. The below example illustrates various switches of netstat.



```
C:\gsbapiraju>netstat ?

Displays protocol statistics and current TCP/IP network connections.

NETSTAT [-a] [-b] [-e] [-f] [-n] [-o] [-p proto] [-r] [-s] [-x] [-t] [interval]

  -a            Displays all connections and listening ports.
  -b            Displays the executable involved in creating each connection or
                listening port. In some cases well-known executables host
                multiple independent components, and in these cases the
                sequence of components involved in creating the connection
                or listening port is displayed. In this case the executable
                name is in [] at the bottom, on top is the component it called,
                and so forth until TCP/IP was reached. Note that this option
                can be time-consuming and will fail unless you have sufficient
                permissions.
  -e            Displays Ethernet statistics. This may be combined with the -s
                option.
  -f            Displays Fully Qualified Domain Names (FQDN) for foreign
                addresses.
  -n            Displays addresses and port numbers in numerical form.
  -o            Displays the owning process ID associated with each connection.
  -p proto      Shows connections for the protocol specified by proto; proto
                may be any of: TCP, UDP, TCPv6, or UDPv6.  If used with the -s
                option to display per-protocol statistics, proto may be any of:
                IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, or UDPv6.
  -q            Displays all connections, listening ports, and bound
                nonlistening TCP ports. Bound nonlistening ports may or may not
                be associated with an active connection.
  -r            Displays the routing table.
  -s            Displays per-protocol statistics.  By default, statistics are
                shown for IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, and UDPv6;
                the -p option may be used to specify a subset of the default.
  -t            Displays the current connection offload state.
  -x            Displays NetworkDirect connections, listeners, and shared
                endpoints.
  -y            Displays the TCP connection template for all connections.
                Cannot be combined with the other options.
  interval      Redisplays selected statistics, pausing interval seconds
                between each display.  Press CTRL+C to stop redisplaying
                statistics.  If omitted, netstat will print the current
                configuration information once.


C:\gsbapiraju>
```

Task 7 : a). Configure and implementation of a Router within a Network using Packet Tracer.

Configure Global Parameters:

Command	**Purpose**
configure terminal Router> enable Router# configure terminal Router(config)#	Enters global configuration mode, when using the console port. If you are connecting to the router using a remote terminal, use the following: telnet router name or address Login: login id Password: ********* Router> enable
Router(config)# hostname Router Router(config)#	hostname name : Specifies the name for the router.
Router(config)# enable secret Griet@14#02&24 Router(config)#	enable secret password: Specifies an encrypted password to prevent unauthorized access to the router
Router(config)# no ip domain-lookup Router(config)#	no ip domain-lookup : Disables the router from translating unfamiliar words (typos) into IP addresses

Configure the Fast Ethernet WAN Interface

Command	**Purpose**

interface type number
Example:
Router(config)# interface fastethernet 4
Router(config-int)
	Enters the configuration mode for a Fast Ethernet WAN interface on the router

Example: Router(config-int)# ip address 192.168.12.2 255.255.255.0 Router(config-int)#	ip address ip-address mask : Sets the IP address and subnet mask for the specified Fast Ethernet interface
no shutdown Example: Router(config-int)# no shutdown Router(config-int)#	Enables the Ethernet interface, changing its state from administratively down to administratively up
exit Example: Router(config-int)# exit Router(config)#	Exits configuration mode for the Fast Ethernet interface and returns to global configuration mode.

Configuring Command-Line Access to the Router:

Command	**Purpose**
line [aux | console | tty | vty] line-number Router(config)# line console 0 Router(config)#	Enters line configuration mode, and specifies the type of line. This example specifies a console terminal for access
password password Router(config)# password Gsbr$$# Router(config)#	Specifies a unique password for the console terminal line
login Router(config)# login Router(config)#	Enables password checking at terminal session login.
exec-timeout minutes [seconds] Router(config)# exec-timeout 5 30 Router(config)#	Sets the interval that the EXEC command interpreter waits until user input is detected. The default is 10 minutes. Optionally, add seconds to the interval value. This example shows a timeout of 5 minutes and 30 seconds. Entering a timeout of 0 0 specifies never to time out
line [aux | console | tty | vty] line-number Router(config)# line vty 0 4 Router(config)#	Specifies a virtual terminal for remote console access.
password password Router(config)# password Gsbr@98# Router(config)#	Specifies a unique password for the virtual terminal line
login Router(config)# login Router(config)#	Enables password checking at the virtual terminal session login.
end Router(config)# end Router#	Exits line configuration mode, and returns to privileged EXEC mode

Summarizing different memories that are used in Cisco Router –

- **RAM** stores the currently working tasks.
- **NVRAM** where startup configuration takes place.
- **ROM** where the information of POST and bootstrap program is available.
- And **Flash Memory** where the operating system of Router IOS is present

b). Configure and examine Network Address Translation (NAT).

NAT Inside and Outside Addresses:

The term inside in a NAT context refers to networks owned by an organization that must be translated. Similarly, the term outside refers to those networks to which the stub network connects, and which are generally not under the control of the organization.

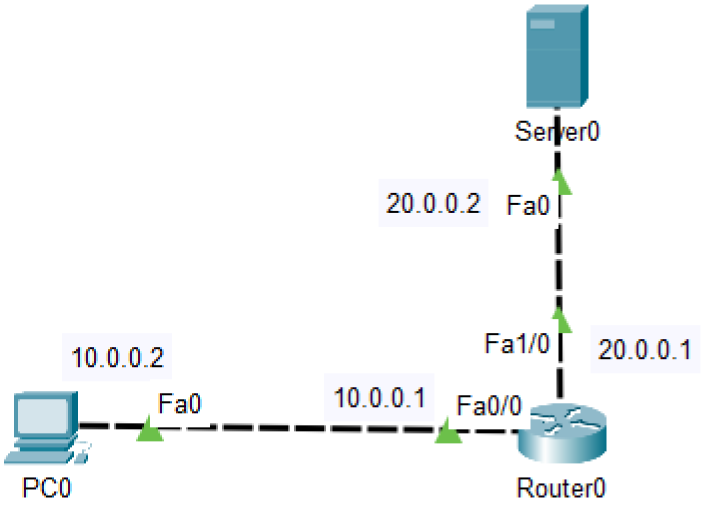
NAT uses the following definitions:

* Inside local address--The IP address that is assigned to a host on the inside network. The address is probably not a legitimate IP address assigned by the NIC or service provider.
* Inside global address--A legitimate IP address (assigned by the NIC or service provider) that represents one or more inside local IP addresses to the outside world
* Outside local address--The IP address of an outside host as it appears to the inside network. Not necessarily a legitimate address, it is allocated from the address space routable on the inside.
* Outside global address--The IP address assigned to a host on the outside network by the owner of the host. The address is allocated from a globally routable address or network space.

NAT types include:

* Static address translation (static NAT)--allows one-to-one mapping between local and global addresses.
* Dynamic address translation (dynamic NAT)--maps unregistered IP addresses to registered IP addresses from a pool of registered IP addresses.
* Overloading--a form of dynamic NAT that maps multiple unregistered IP addresses to a single registered IP address (many to one) using different ports. This method is also known as
* Port Address Translation (PAT). By using PAT (NAT overload), thousands of users can be connected to the Internet using only one real global IP address

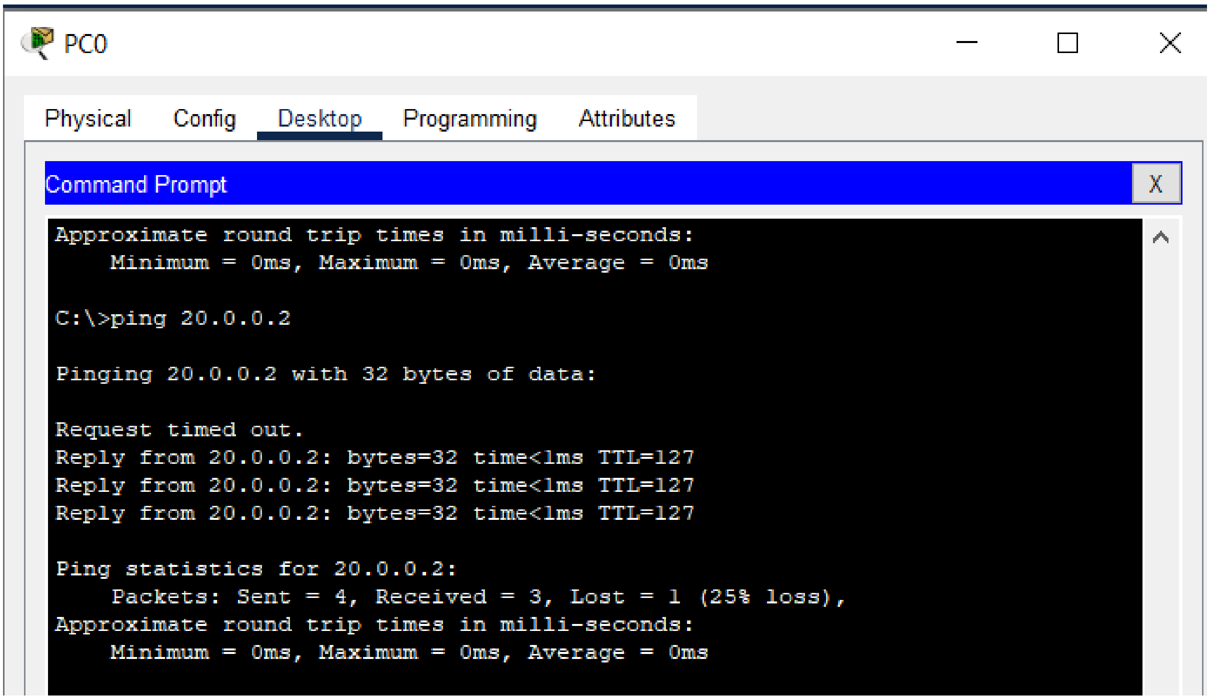
i).Static routing



Router configuration:

```
Router>
Router>en
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int fa0/0
Router(config-if)#ip add 10.0.0.1 255.0.0.0
Router(config-if)#no shut

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to
up

Router(config-if)#ip nat inside
Router(config-if)#int fa1/0
Router(config-if)#ip add 20.0.0.1 255.0.0.0
Router(config-if)#no shut

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet1/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to
up

Router(config-if)#
Router(config-if)#ip nat outside
Router(config-if)#exit

Router(config)#ip nat inside source static 10.0.0.2 200.100.50.25
Router(config)#end
Router#
%SYS-5-CONFIG_I: Configured from console by console

Router#
Router#sh ip nat translations
Pro Inside global Inside local Outside local Outside global
--- 200.100.50.25 10.0.0.2 --- ---

Router#
```

To test the connection ping from the terminal



```
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 20.0.0.2

Pinging 20.0.0.2 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.2: bytes=32 time<1ms TTL=127
Reply from 20.0.0.2: bytes=32 time<1ms TTL=127
Reply from 20.0.0.2: bytes=32 time<1ms TTL=127

Ping statistics for 20.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

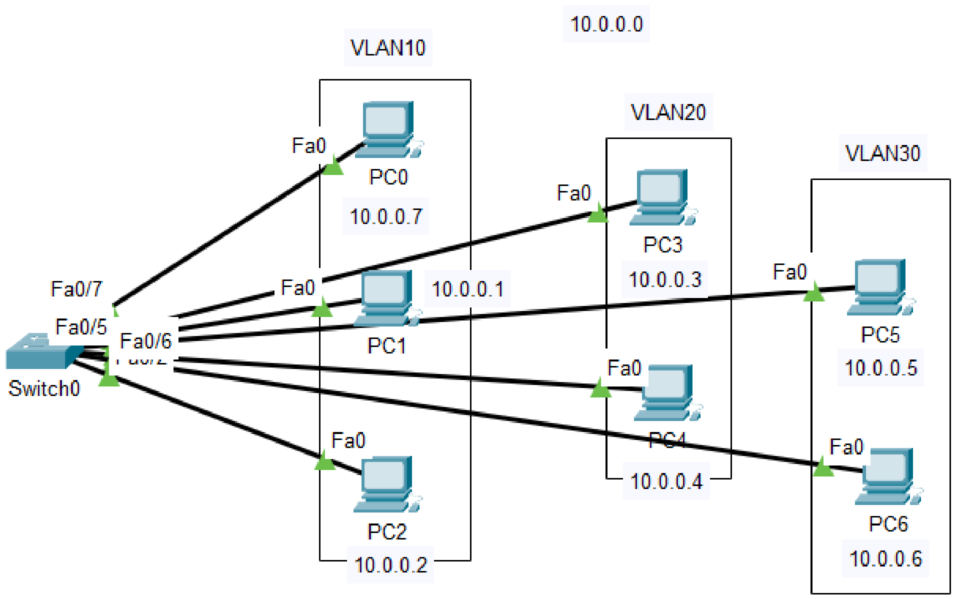
ii).Dynamic routing

```
Router>
Router>en
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int fa0/0
Router(config-if)#ip add 10.0.0.1 255.0.0.0
Router(config-if)#no shut

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to
up

Router(config-if)#
Router(config-if)#
Router(config-if)#int fa1/0
Router(config-if)#ip add 20.0.0.1 255.0.0.0
Router(config-if)#no shut

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet1/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to
up
```

```
Router(config-if)#
Router(config-if)#exit
Router(config)#
Router(config)#
Router(config)#int fa0/0
Router(config-if)#ip nat inside
Router(config-if)#int fa1/0
Router(config-if)#ip nat outside
Router(config-if)#
Router(config-if)#exit
Router(config)#
Router(config)#access-list 1 permit 10.0.0.0 0.0.0.5
Router(config)#ip nat pool mypool 50.25.10.5 50.25.10.10 netmask 255.0.0.0
Router(config)#ip nat inside source list 1 pool mypool
Router(config)#
```

Task-8 : a).Configure network topology to implement VLANs with using Packet Tracer software.

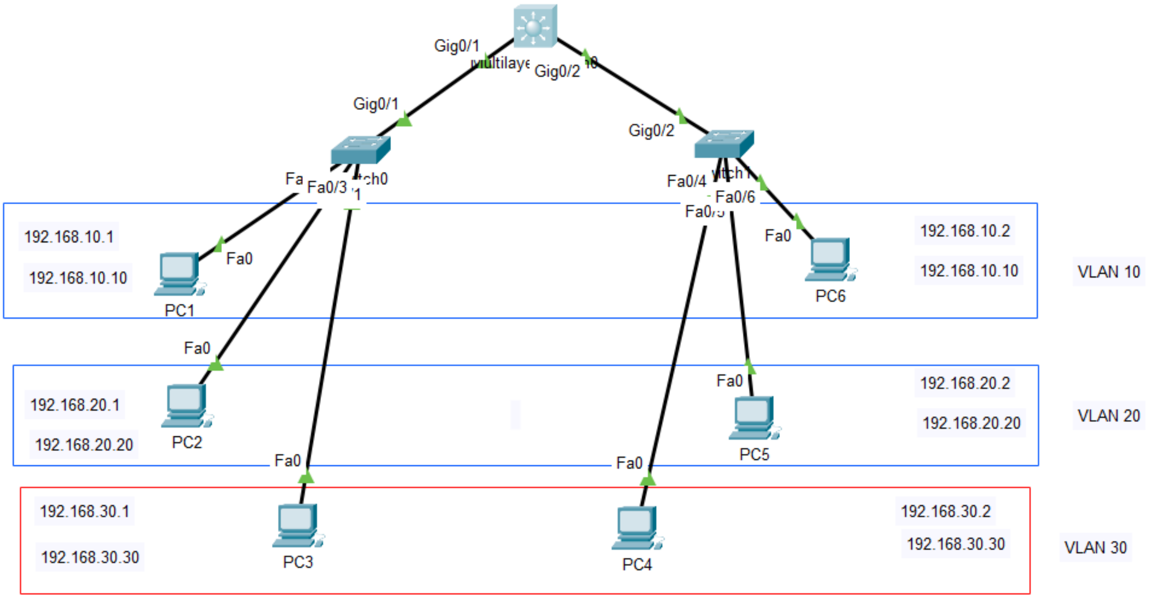
INTRA VLAN



```
Switch>
Switch>en
Switch#
Switch#conf t
Enter configuration commands, one per line. End
with CNTL/Z.
Switch(config)#vlan 10
Switch(config-vlan)#name project1
Switch(config-vlan)#int f0/1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 10

Switch(config-if)#int f0/2
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 10

Switch(config-if)#int f0/7
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 10

Switch(config-if)#vlan 20
Switch(config-vlan)#int f0/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 20

Switch(config-if)#int f0/4
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 20

Switch(config-if)#vlan 30
Switch(config-vlan)#name project3

Switch(config-vlan)#int f0/5
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 30

Switch(config-if)#int f0/6
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 30
Switch(config-if)#end
Switch#wr
```

Observe that the PCs of same vlan will ping and other vlans are isolated.

INTER VLAN



```
Switch>
Switch>EN
Switch#
Switch#conf t
Switch(config)#
Switch(config)#vlan 10
Switch(config-vlan)#vlan 20
Switch(config-vlan)#vlan 30

Switch(config-vlan)#int f0/1
Switch(config-if)#no shut
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 10
Switch(config-if)#int f0/2
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 20
Switch(config-if)#int f0/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 30
Switch(config-if)#int g0/1

Switch(config-if)#switchport mode trunk
Switch(config-if)#end
Switch#wr
Building configuration...
[OK]
Switch#
```

```
Switch>
Switch>en
Switch#conf t
Switch(config)#hostname SW2
SW2(config)#
SW2(config)#vlan 10
SW2(config-vlan)#vlan 20
SW2(config-vlan)#vlan 30

SW2(config-vlan)#int f0/4
SW2(config-if)#switchport mode access
SW2(config-if)#switchport access vlan 30
SW2(config-if)#int f0/5
SW2(config-if)#switchport mode access
SW2(config-if)#switchport access vlan 20
SW2(config-if)#int f0/6
SW2(config-if)#switchport mode access
SW2(config-if)#switchport access vlan 10
SW2(config-if)#int g0/2
SW2(config-if)#sw

SW2(config-if)#switchport mode trunk
SW2(config-if)#end
SW2#wr
Building configuration...
[OK]
SW2#
```

Multilayer Switch Configuration.

```
Switch>
Switch>en
Switch#
Switch#conf t
Switch(config)#hostname MLSW
MLSW(config)#vlan 10
MLSW(config-vlan)#vlan 20
MLSW(config-vlan)#vlan 30
MLSW(config-vlan)#exit

MLSW(config)#
MLSW(config)#int range g0/1-2
MLSW(config-if-range)#switchport trunk encapsulation dot1q
MLSW(config-if-range)#switchport mode trunk
MLSW(config-if-range)#end
```

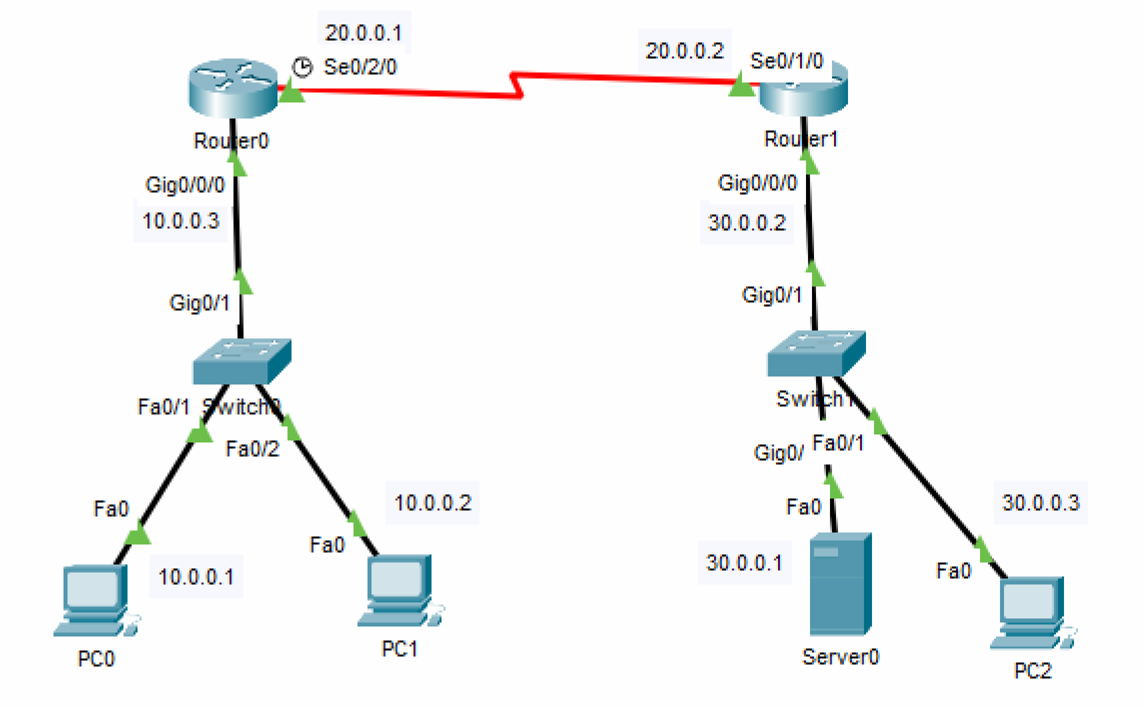
After this you will be able to ping in between the same VLANs. But not other vlans.

Ex: 192.168.10.1 to 10.2
But you cannot access
192.168.10.1 to 20.1

Below conf. is to access from one pc to all PCs.

```
MLSW#conf t
Enter configuration commands, one per line. End with CNTL/Z.
MLSW(config)#
MLSW(config)#ip routing
MLSW(config)#int vlan 10
MLSW(config-if)#
MLSW(config-if)#ip add 192.168.10.10 255.255.255.0
MLSW(config-if)#no shut
MLSW(config-if)#int vlan 20
MLSW(config-if)#
MLSW(config-if)#ip add 192.168.20.20 255.255.255.0
MLSW(config-if)#no shut
MLSW(config-if)#
MLSW(config-if)#int vlan 30
MLSW(config-if)#
MLSW(config-if)#ip add 192.168.30.30 255.255.255.0
MLSW(config-if)#no shut
MLSW(config-if)#
MLSW(config-if)#end
MLSW#
MLSW#wr
Building configuration...
[OK]
MLSW#
```

b).Configure network topology and implement static routing using Packet Tracer Software. .STATIC ROUTING:

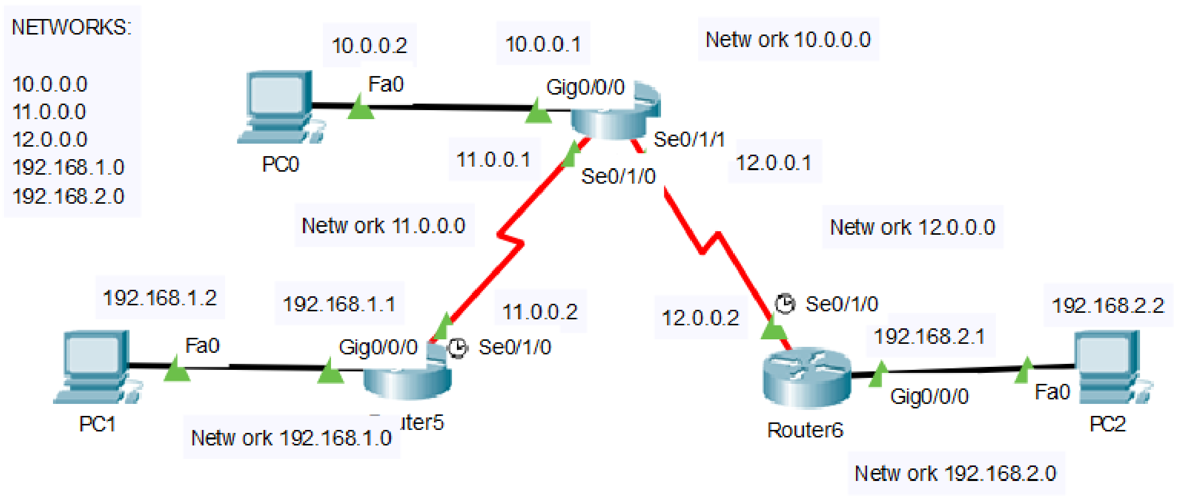


Router 0	Router 1
Router>en Router# Router#conf t Router(config)# Router(config)#int g0/0/0 Router(config-if)#ip address 10.0.0.3 255.0.0.0 Router(config-if)#no shut Router(config-if)#int s0/2/0 Router(config-if)#ip address 20.0.0.1 255.0.0.0 Router(config-if)#no shut Router(config-if)#exit Router(config)# **Router(config)#ip route 30.0.0.0 255.0.0.0 s0/2/0**	Router# Router#conf t Router(config)# Router(config)#int g0/0/0 Router(config-if)#ip address 30.0.0.2 255.0.0.0 Router(config-if)#no shut Router(config-if)# Router(config-if)#int s0/1/0 Router(config-if)#ip add 20.0.0.2 255.0.0.0 Router(config-if)#no shut Router(config-if)#exit Router(config)# **Router(config)#ip route 10.0.0.0 255.0.0.0 s0/1/0** %Default route without gateway, if not a point-to-point interface, may impact performance Router(config)#

Task-9

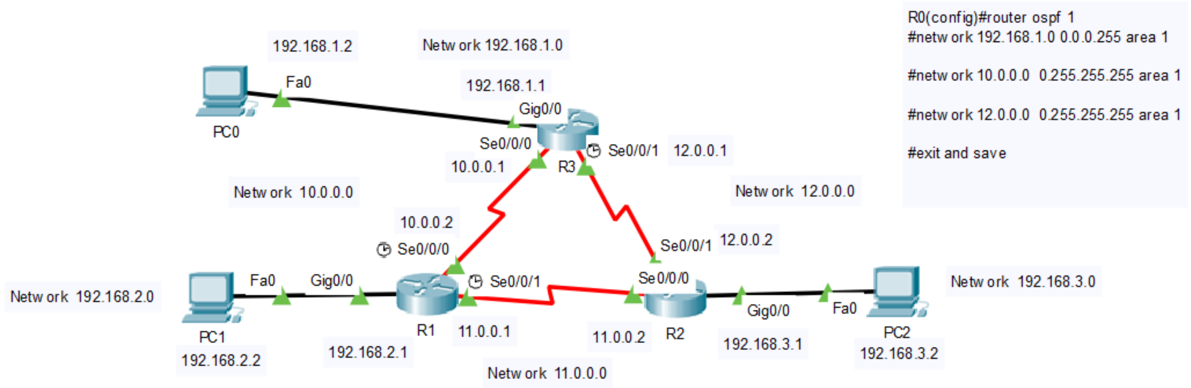
a). Configure network topology and implement dynamic routing protocol such as RIP, OSPF using Packet Tracer.

RIP



R4 (Router 4)	R5 (Router 5)	R6 (Router 6)
router rip network 10.0.0.0 network 11.0.0.0 network 12.0.0.0 interface GigabitEthernet0/0/0 ip address 10.0.0.1 255.0.0.0 interface Serial0/1/0 ip address 11.0.0.1 255.0.0.0 interface Serial0/1/1 ip address 12.0.0.1 255.0.0.0 no shutdown	router rip network 11.0.0.0 network 192.168.1.0 interface GigabitEthernet0/0/0 ip address 192.168.1.1 255.255.255.0 no shutdown interface Serial0/1/0 ip address 11.0.0.2 255.0.0.0 clock rate 2000000 no shutdown	router rip network 12.0.0.0 network 192.168.2.0 interface GigabitEthernet0/0/0 ip address 192.168.2.1 255.255.255.0 no shutdown interface Serial0/1/0 ip address 12.0.0.2 255.0.0.0 clock rate 2000000 no shutdown

OSPF



```
R0(config)#router ospf 1
#network 192.168.1.0 0.0.0.255 area 1

#network 10.0.0.0  0.255.255.255 area 1

#network 12.0.0.0  0.255.255.255 area 1

#exit and save
```

R1 (Router 1)	R3 (Router 3)
Router(config)#router ospf 1 Router(config-router)#network 192.168.2.0 255.0.0.0 area 1 Router(config-router)#network 192.168.2.0 0.0.0.255 area 1 Router(config-router)#network 10.0.0.0 0.255.255.255 area 1 Router(config-router)#network 10.0.0.0 0.255.255.255 area 1 Router(config-router)#network 11.0.0.0 0.255.255.255 area 1 Router(config-router)#^Z Router#copy run startup-config	router ospf 1 network 192.168.1.0 0.0.0.255 area 1 network 10.0.0.0 0.255.255.255 area 1 network 12.0.0.0 0.255.255.255 area 1
Router>enable Router# Router#configure terminal Router(config)#interface GigabitEthernet0/0 Router(config-if)#ip address 192.168.2.1 255.255.255.0 Router(config-if)#no shutdown Router(config-if)#exit Router(config)#interface Serial0/0/0 Router(config-if)#clock rate 125000 **************** Router(config-if)#ip address 10.0.0.2 255.0.0.0 Router(config-if)#no shutdown Router(config-if)#exit	interface GigabitEthernet0/0 ip address 192.168.1.1 255.255.255.0 interface Serial0/0/0 ip address 10.0.0.1 255.0.0.0 no shut down interface Serial0/0/0 ip address 10.0.0.1 255.0.0.0 interface Serial0/0/1 ip address 12.0.0.1 255.0.0.0 clock rate 125000 ****************** no shut down

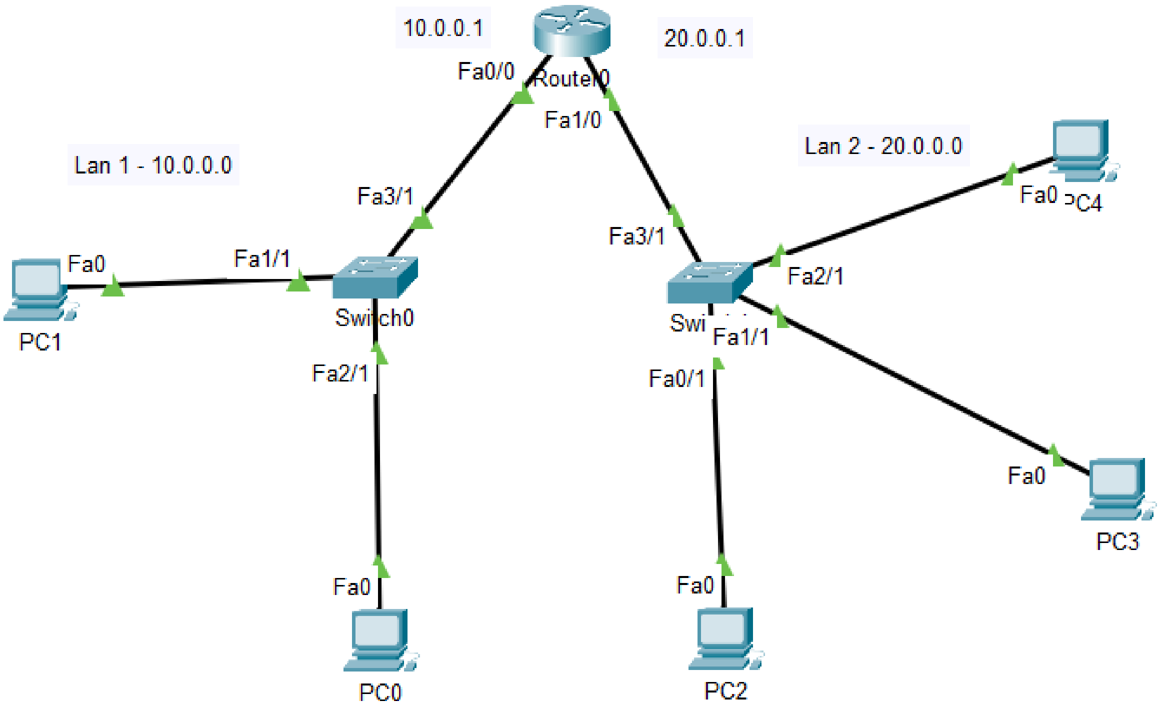
Router(config)#interface Serial0/0/1 Router(config-if)# clock rate 125000 Router(config-if)#ip address 11.0.0.1 255.0.0.0 Router(config-if)#ip address 11.0.0.1 255.0.0.0 Router(config-if)#no shutdown	

Router(config)#router ospf 1 Router(config-router)#network 192.168.3.0 0.0.0.255 area 1 Router(config-router)#network 11.0.0.0 0.255.255.255 area 1 Router(config-router)#network 12.0.0.0 0.255.255.255 area 1 Router(config-router)# Router(config)#interface GigabitEthernet0/0 Router(config-if)#ip address 192.168.3.1 255.255.255.0 Router(config-if)#ip address 192.168.3.1 255.255.255.0 Router(config-if)#no shutdown	Router(config)#interface Serial0/0/0 Router(config-if)#ip address 11.0.0.2 255.0.0.0 Router(config-if)#ip address 11.0.0.2 255.0.0.0 Router(config-if)#no clock rate This command applies only to DCE interfaces Router(config-if)#no shutdown Router(config)#interface Serial0/0/1 Router(config-if)# no clock rate This command applies only to DCE interfaces Router(config-if)#ip address 12.0.0.2 255.0.0.0 Router(config-if)#ip address 12.0.0.2 255.0.0.0 Router(config-if)#no shutdown

Task 10:

a) Configure DHCP Server in the Network using packet tracer software.
b) Configure a remote login using SSH and Telnet.

DHCP server:



```
Router>
Router>en
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#int f0/0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shut

Router(config-if)#ip dhcp pool lan1
Router(dhcp-config)#network 10.0.0.1 255.0.0.0

Router(dhcp-config)#
Router(dhcp-config)#int f1/0
Router(config-if)#ip address 20.0.0.1 255.0.0.0
Router(config-if)#no shut

Router(config-if)#
Router(config-if)#ip dhcp pool lan2
Router(dhcp-config)#network 20.0.0.1 255.0.0.0
Router(dhcp-config)#
```

After configuring the router, goto every PC and select DHCP to get the ip address from the DHCP pool. After completing this process run the below command to see the IP address allocated.

```
Router#sh ip dhcp binding
IP address Client-ID/ Lease expiration Type
Hardware address
10.0.0.2 00D0.D3CD.5175 -- Automatic
10.0.0.3 000A.41CD.3BB2 -- Automatic
20.0.0.2 0060.5C8D.DBBE -- Automatic
20.0.0.3 0005.5EE5.2223 -- Automatic
20.0.0.4 00E0.B070.0842 -- Automatic
```

```
Router# sh run

!
ip dhcp pool lan1
network 10.0.0.0 255.0.0.0
ip dhcp pool lan2
network 20.0.0.0 255.0.0.0
```

DHCP

Configuring DHCP Server

Beginning in privileged EXEC mode, follow these steps to configure DHCP server.
The following example configures the DHCP server:

```
Router# configure terminal
Router(config)# ip dhcp included-address 192.168.1.101 192.168.1.150
Router(config)# ip dhcp pool
Router(dhcp-config)# network 192.168.1.0 255.255.255.0
Router(dhcp-config)# domain-name cisco.com
Router(dhcp-config)# dns-server 8.8.8.8
Router(dhcp-config)# default-router 192.168.1.1
Router(dhcp-config)# exit
Router(config)# service dhcp vlan1
```

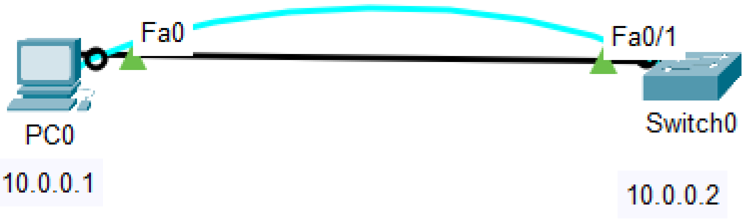
Command Purpose

1. **configure terminal** Enter global configuration mode.
2. **ip dhcp pool** Create a DHCP server address pool and enters DHCP pool configuration mode.

Note: If you have changed the parameters of the DHCP server, you must perform a refresh using the **no service dhcp** *interface-type number* command and **service dhcp** *interface-type number* commands.

3. network *network-number mask* Specify the subnet network number and mask of the DHCP address pool.
4. domain-name *domain* Specify the domain name for the client.
5. dns-server *address* Specify the IP address of a DNS server that is available to a DHCP client.
6. default-router *address* Specify the IP address of the default router for a DHCP client.
7. exit Return to privileged EXEC mode.
8. service dhcp *interface-type number* Enable DHCP server on the interface.

10 b) Configure a remote login using SSH and Telnet.



First connect the console cable from PC0 RS232 port to Switch console port.

From PC0 Terminal Switch> Switch>en Switch#conf terminal Switch(config)# Switch(config)#interface vlan 1 Switch(config-if)#ip address 10.0.0.2 255.0.0.0 Switch(config-if)#no shut Switch(config-if)#end Switch# Switch#sh run Before Telnet configuration *C:\>telnet 10.0.0.2* *Trying 10.0.0.2 ...* *% Connection timed out; remote host not responding* *C:\>telnet 10.0.0.2* *Trying 10.0.0.2 ...Open* *[Connection to 10.0.0.2 closed by foreign host]*	Switch# Switch#conf t Switch(config)# Switch(config)#username myname password mypass Switch(config)#line vty 0 15 Switch(config-line)#login local Switch(config-line)#transport input telnet Switch(config-line)# After Telnet configuration C:\>telnet 10.0.0.2 Trying 10.0.0.2 ...Open User Access Verification Username: myname Password: Switch>

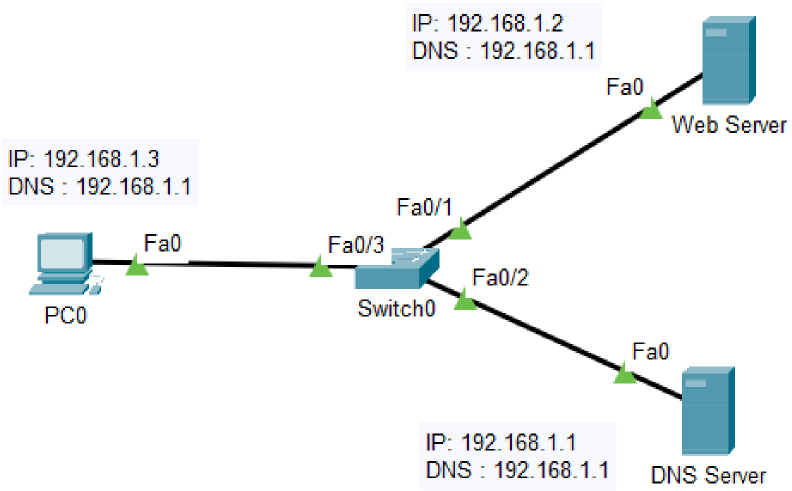
SSH

```
Switch#conf t
Switch(config)#
Switch(config)#hostname myswitch
myswitch(config)#ip domain-name mydomain
myswitch(config)#username myname1 password mypass1
myswitch(config)#line vty 0 15
myswitch(config-line)#login local
myswitch(config-line)#transport input ssh
myswitch(config-line)#exit
myswitch(config)#crypto key generate rsa

The name for the keys will be: myswitch.mydomain
Choose the size of the key modulus in the range of 360 to 2048 for your
General Purpose Keys. Choosing a key modulus greater than 512 may take
a few minutes.

How many bits in the modulus [512]:
% Generating 512 bit RSA keys, keys will be non-exportable...[OK]

myswitch(config)#exit
```

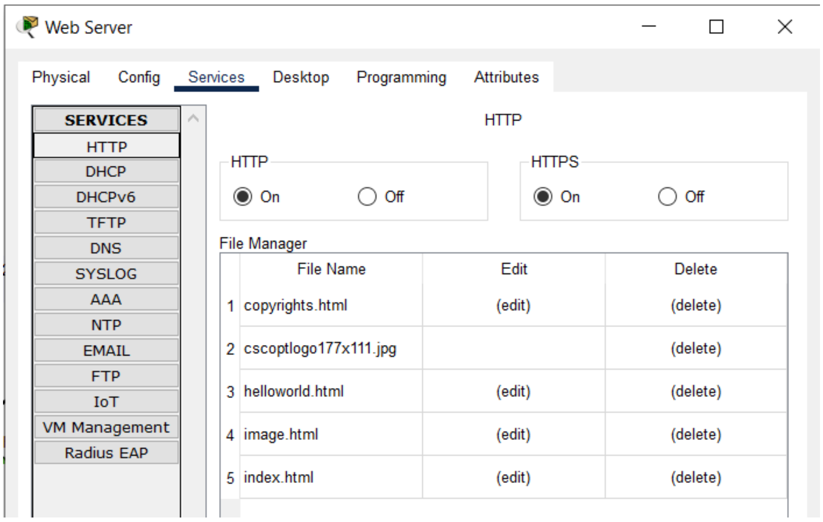
Now test SSH connection

```
C:\>ssh -l myname1 10.0.0.2
Password:

myswitch>
```

Task 11 a). Establish a webserver connection using the PCs browser.



Configure the webserver connection and DNS server



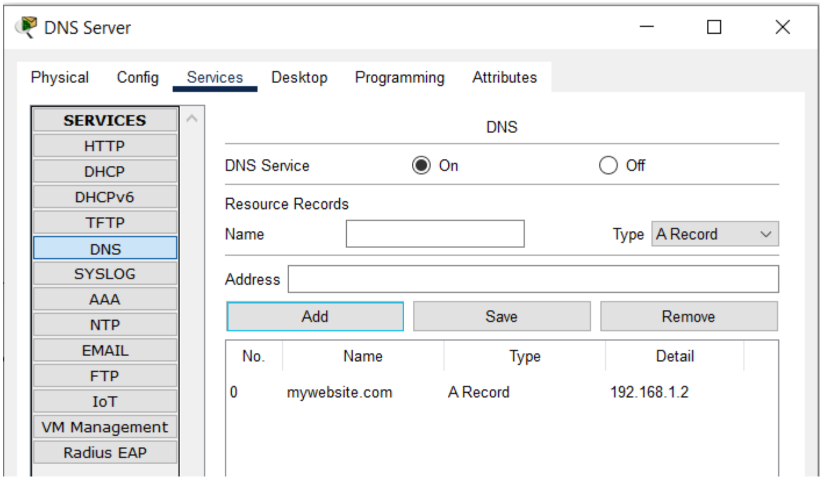
HTTP

HTTP ● On ○ Off

HTTPS ● On ○ Off

File Manager

	File Name	Edit	Delete
1	copyrights.html	(edit)	(delete)
2	cscoptlogo177x111.jpg		(delete)
3	helloworld.html	(edit)	(delete)
4	image.html	(edit)	(delete)
5	index.html	(edit)	(delete)



DNS

DNS Service ● On ○ Off

Resource Records

Name: Type: A Record

Address:

No.	Name	Type	Detail
0	mywebsite.com	A Record	192.168.1.2

11b) Install wireshark and view ii). Network Traffic and iii).Examine ethernet frames View Wired and Wireless NIC information.

Objectives:

a. Capture and analyse local ICMP data in wireshark
b. Capture and analyse Remote ICMP data in wireshark

Install wireshark latest version.

Step 1: Retrieve your PC interface addresses.

For this lab, you will need to retrieve your PC IP address and its network interface card (NIC) physical address, also called the MAC address.
a. Open a command window, type **ipconfig /all**, and then press Enter.
b. Note the IP address of your PC interface, its description, and its MAC (physical) address

Steps

1: Open wireshark and start capturing the packets. The data lines will appear in different colors based on protocol.

2. open command prompt and

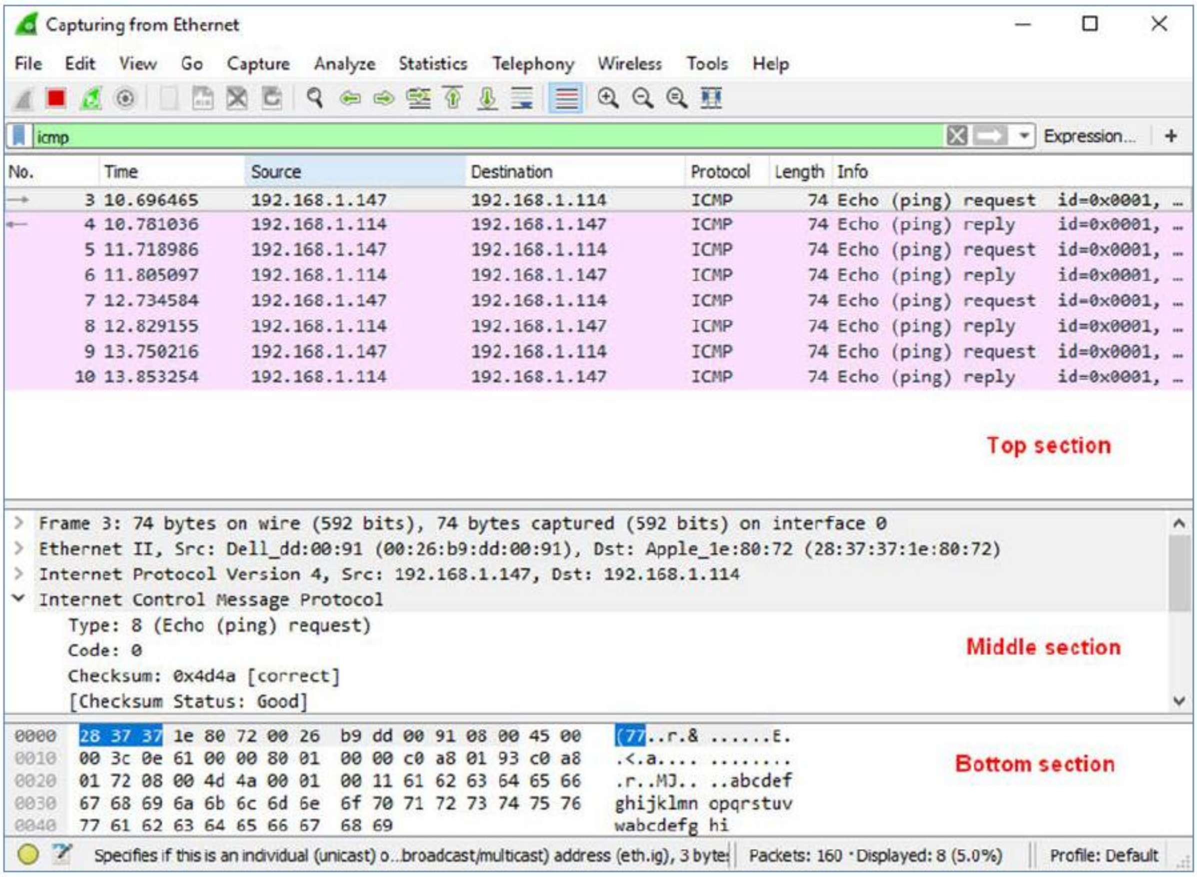
```
ping any url. Ex: ping google.com
ping sdc.in
ping yahoo.com
ping cisco.com
```

3. Stop capturing the packets.

Step 3: Examine the captured data.

Wireshark data is displayed in three sections: 1) The top section displays the list of PDU frames captured with a summary of the IP packet information listed; 2) the middle section lists PDU information for the frame selected in the top part of the screen and separates a captured PDU frame by its protocol layers; and 3) the bottom section displays the raw data of each layer. The raw data is displayed in both hexadecimal and decimal form.

Click the first ICMP request PDU frames in the top section of Wireshark. Notice that the Source column has your PC IP address, and the Destination column contains the IP address you pinged.



```
icmp
```

```
No.  Time       Source         Destination    Protocol Length Info
   3 10.696465  192.168.1.147  192.168.1.114  ICMP        74 Echo (ping) request  id=0x0001, …
   4 10.781036  192.168.1.114  192.168.1.147  ICMP        74 Echo (ping) reply    id=0x0001, …
   5 11.718986  192.168.1.147  192.168.1.114  ICMP        74 Echo (ping) request  id=0x0001, …
   6 11.805097  192.168.1.114  192.168.1.147  ICMP        74 Echo (ping) reply    id=0x0001, …
   7 12.734584  192.168.1.147  192.168.1.114  ICMP        74 Echo (ping) request  id=0x0001, …
   8 12.829155  192.168.1.114  192.168.1.147  ICMP        74 Echo (ping) reply    id=0x0001, …
   9 13.750216  192.168.1.147  192.168.1.114  ICMP        74 Echo (ping) request  id=0x0001, …
  10 13.853254  192.168.1.114  192.168.1.147  ICMP        74 Echo (ping) reply    id=0x0001, …
```

```
> Frame 3: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
> Ethernet II, Src: Dell_dd:00:91 (00:26:b9:dd:00:91), Dst: Apple_1e:80:72 (28:37:37:1e:80:72)
> Internet Protocol Version 4, Src: 192.168.1.147, Dst: 192.168.1.114
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x4d4a [correct]
    [Checksum Status: Good]
```

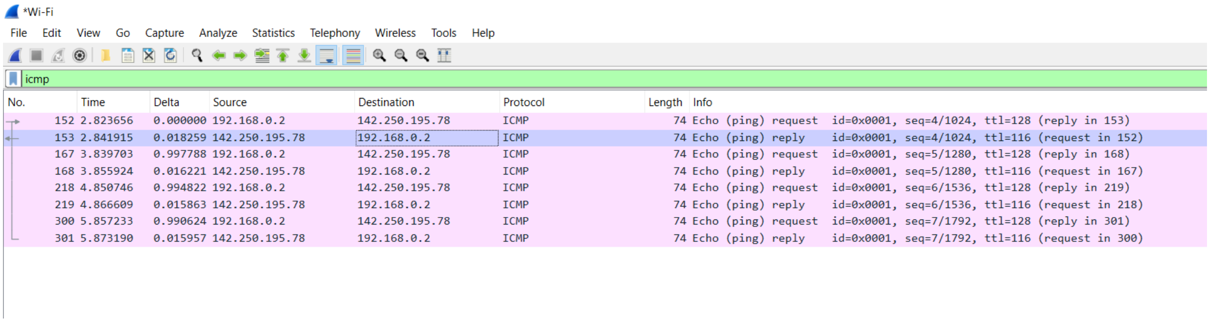
```
0000  28 37 37 1e 80 72 00 26  b9 dd 00 91 08 00 45 00   (77..r.& ......E.
0010  00 3c 0e 61 00 00 80 01  00 00 c0 a8 01 93 c0 a8   .<.a.... ........
0020  01 72 08 00 4d 4a 00 01  00 11 61 62 63 64 65 66   .r..MJ.. ..abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi
```

Specifies if this is an individual (unicast) o...broadcast/multicast) address (eth.ig), 3 byte: | Packets: 160 · Displayed: 8 (5.0%) | Profile: Default

4. Go to the filter bar and type and check the following protocols

ICMP enter.
DNS
TCP
UDP
ARP

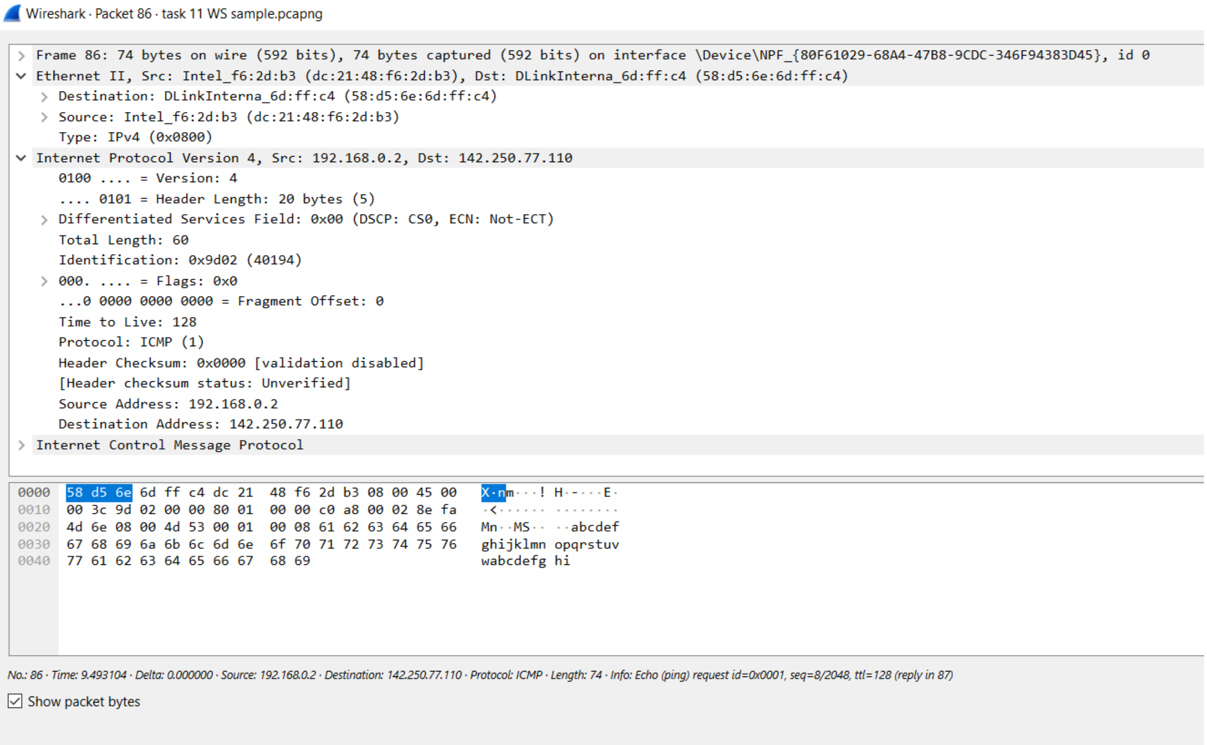
And observe the packets.


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
icmp
```

```
No.  Time      Delta    Source          Destination     Protocol  Length Info
 152 2.823656  0.000000 192.168.0.2     142.250.195.78  ICMP      74 Echo (ping) request  id=0x0001, seq=4/1024, ttl=128 (reply in 153)
 153 2.841915  0.018259 142.250.195.78  192.168.0.2     ICMP      74 Echo (ping) reply    id=0x0001, seq=4/1024, ttl=116 (request in 152)
 167 3.839703  0.997788 192.168.0.2     142.250.195.78  ICMP      74 Echo (ping) request  id=0x0001, seq=5/1280, ttl=128 (reply in 168)
 168 3.855924  0.016221 142.250.195.78  192.168.0.2     ICMP      74 Echo (ping) reply    id=0x0001, seq=5/1280, ttl=116 (request in 167)
 218 4.850746  0.994822 192.168.0.2     142.250.195.78  ICMP      74 Echo (ping) request  id=0x0001, seq=6/1536, ttl=128 (reply in 219)
 219 4.866609  0.015863 142.250.195.78  192.168.0.2     ICMP      74 Echo (ping) reply    id=0x0001, seq=6/1536, ttl=116 (request in 218)
 300 5.857233  0.990624 192.168.0.2     142.250.195.78  ICMP      74 Echo (ping) request  id=0x0001, seq=7/1792, ttl=128 (reply in 301)
 301 5.873190  0.015957 142.250.195.78  192.168.0.2     ICMP      74 Echo (ping) reply    id=0x0001, seq=7/1792, ttl=116 (request in 300)
```

Examine the Ethernet frame fields in the middle section:



Wireshark · Packet 86 · task 11 WS sample.pcapng

```
> Frame 86: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{80F61029-68A4-47B8-9CDC-346F94383D45}, id 0
v Ethernet II, Src: Intel_f6:2d:b3 (dc:21:48:f6:2d:b3), Dst: DLinkInterna_6d:ff:c4 (58:d5:6e:6d:ff:c4)
  > Destination: DLinkInterna_6d:ff:c4 (58:d5:6e:6d:ff:c4)
  > Source: Intel_f6:2d:b3 (dc:21:48:f6:2d:b3)
    Type: IPv4 (0x0800)
v Internet Protocol Version 4, Src: 192.168.0.2, Dst: 142.250.77.110
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x9d02 (40194)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: ICMP (1)
    Header Checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.0.2
    Destination Address: 142.250.77.110
> Internet Control Message Protocol

0000  58 d5 6e 6d ff c4 dc 21  48 f6 2d b3 08 00 45 00   X·nm···! H·-···E·
0010  00 3c 9d 02 00 00 80 01  00 00 c0 a8 00 02 8e fa   ·<······ ········
0020  4d 6e 08 00 4d 53 00 01  00 08 61 62 63 64 65 66   Mn··MS·· ··abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi
```

No.: 86 · Time: 9.493104 · Delta: 0.000000 · Source: 192.168.0.2 · Destination: 142.250.77.110 · Protocol: ICMP · Length: 74 · Info: Echo (ping) request id=0x0001, seq=8/2048, ttl=128 (reply in 87)

☑ Show packet bytes

View Wired and Wireless NIC information:

Step 1: Use the Network and Sharing Center.

a. Open the Network and Sharing Center by clicking the Windows Start button > Control Panel > View network status and tasks under Network and Internet heading in the Category View.
b. In the left pane, click the Change adapter settings link.
c. The Network Connections window displays, which provides the list of NICs available on this PC. Look for your Local Area Connection and Wireless Network Connection adapters in this window.

Or

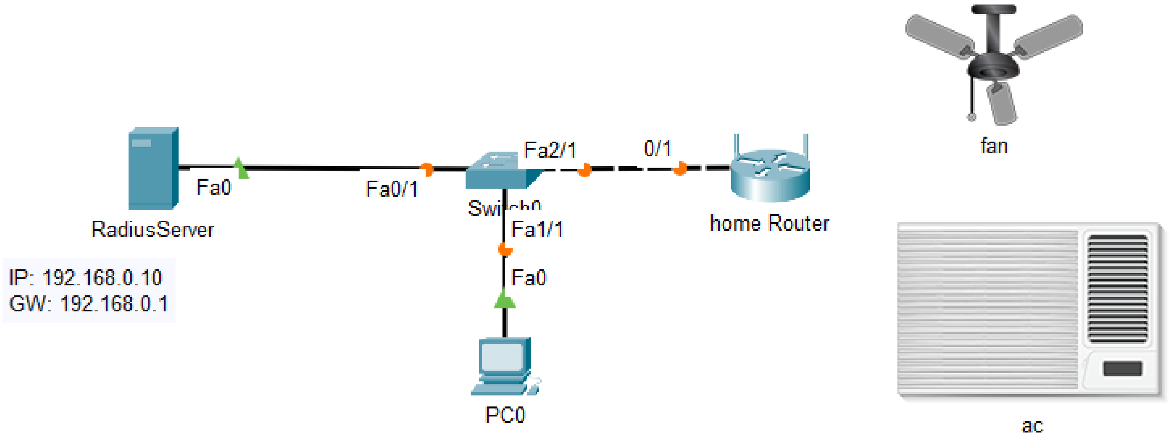
1. Right click on start (windows button)
Settings – status – properties

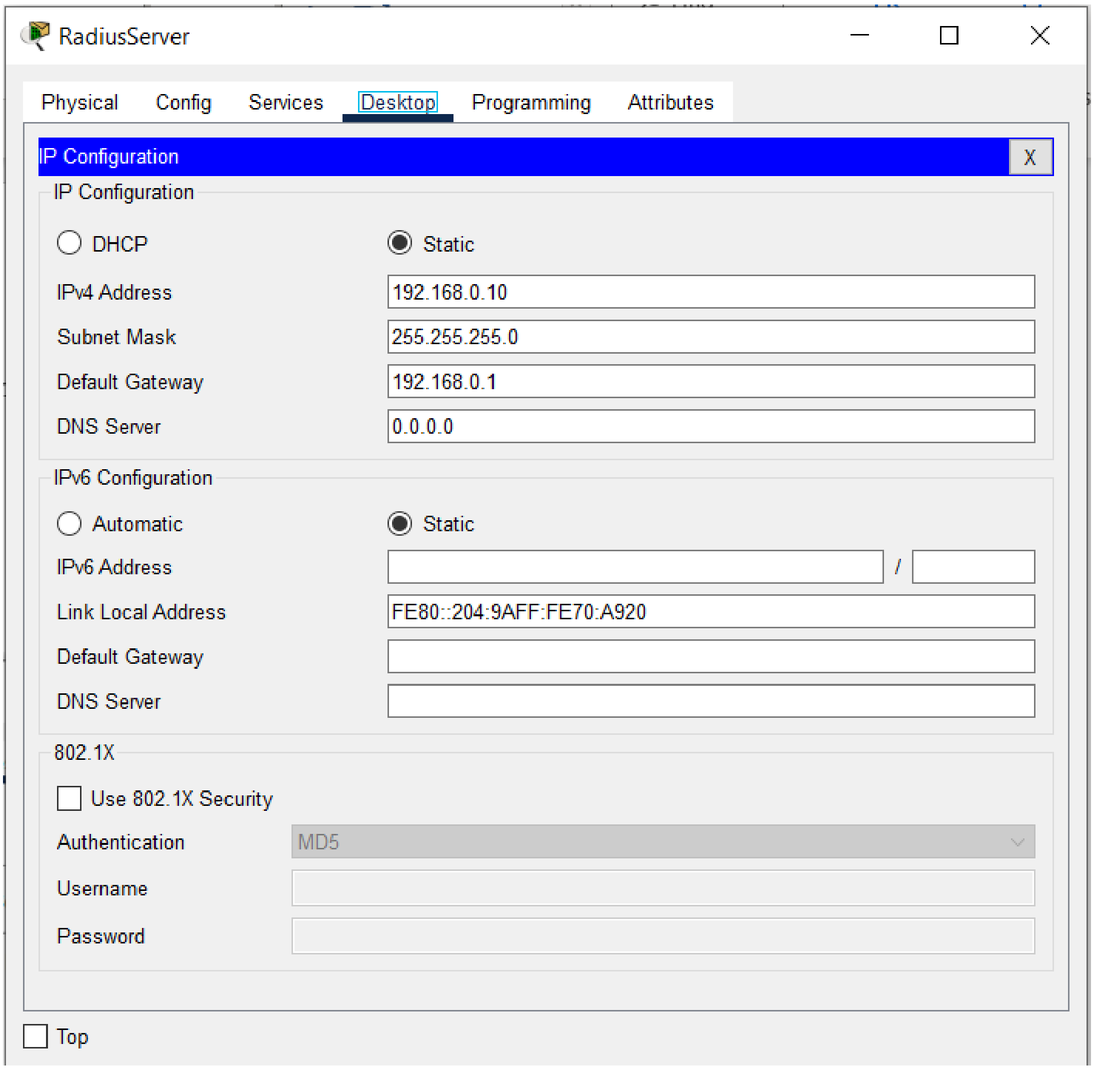
Compare with:

Open a command window prompt and type **ipconfig /all**
And observe the above addresses, both must be same.

Task-12

a). Adding IoT devices to Smart Homes using Packet Tracer
b). Connect and Monitor IoT Devices using Packet Tracer





IP Configuration

IP Configuration

○ DHCP ● Static

IPv4 Address	192.168.0.10
Subnet Mask	255.255.255.0
Default Gateway	192.168.0.1
DNS Server	0.0.0.0

IPv6 Configuration

○ Automatic ● Static

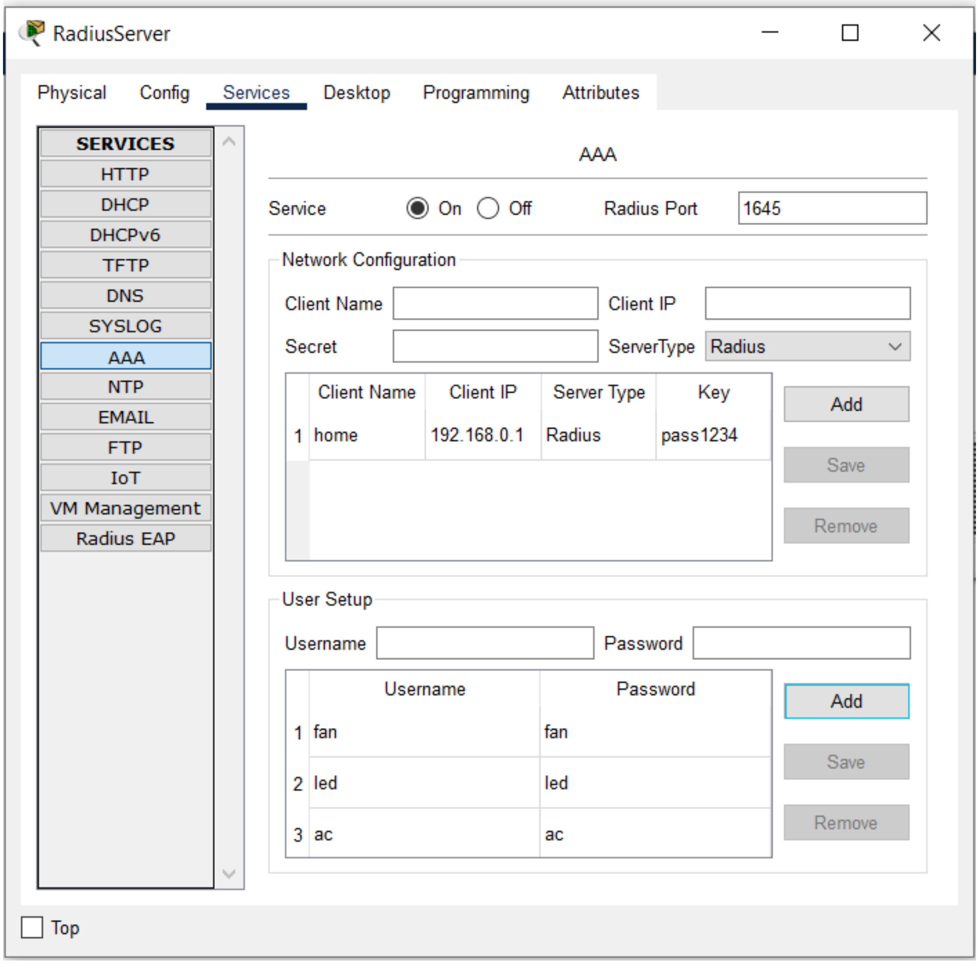
IPv6 Address	/
Link Local Address	FE80::204:9AFF:FE70:A920
Default Gateway	
DNS Server	

802.1X

□ Use 802.1X Security

Authentication	MD5
Username	
Password	

□ Top



RadiusServer

AAA

Service ● On ○ Off Radius Port 1645

Network Configuration

Client Name | Client IP

Secret | ServerType Radius

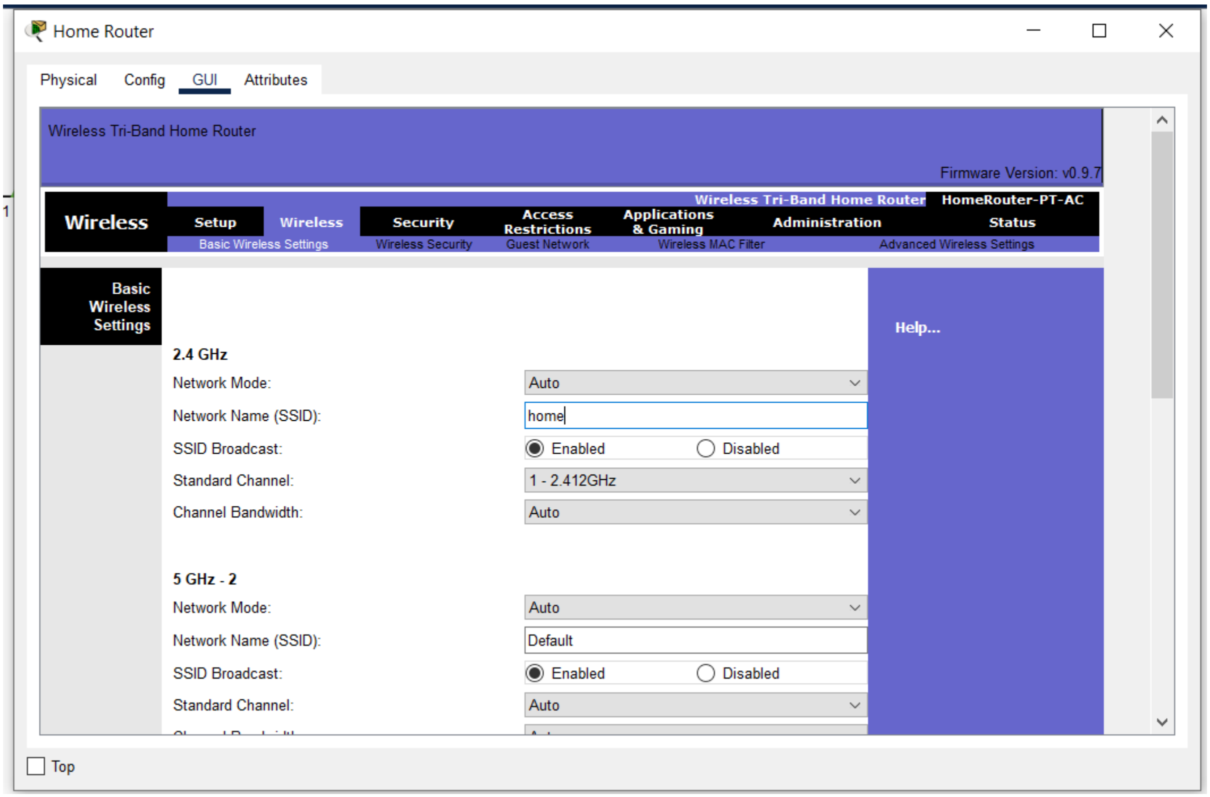
	Client Name	Client IP	Server Type	Key
1	home	192.168.0.1	Radius	pass1234

User Setup

Username | Password

	Username	Password
1	fan	fan
2	led	led
3	ac	ac

□ Top



Home Router

2.4 GHz

Network Mode: Auto

Network Name (SSID): home

SSID Broadcast: ● Enabled ○ Disabled

Standard Channel: 1 - 2.412GHz

Channel Bandwidth: Auto

5 GHz - 2

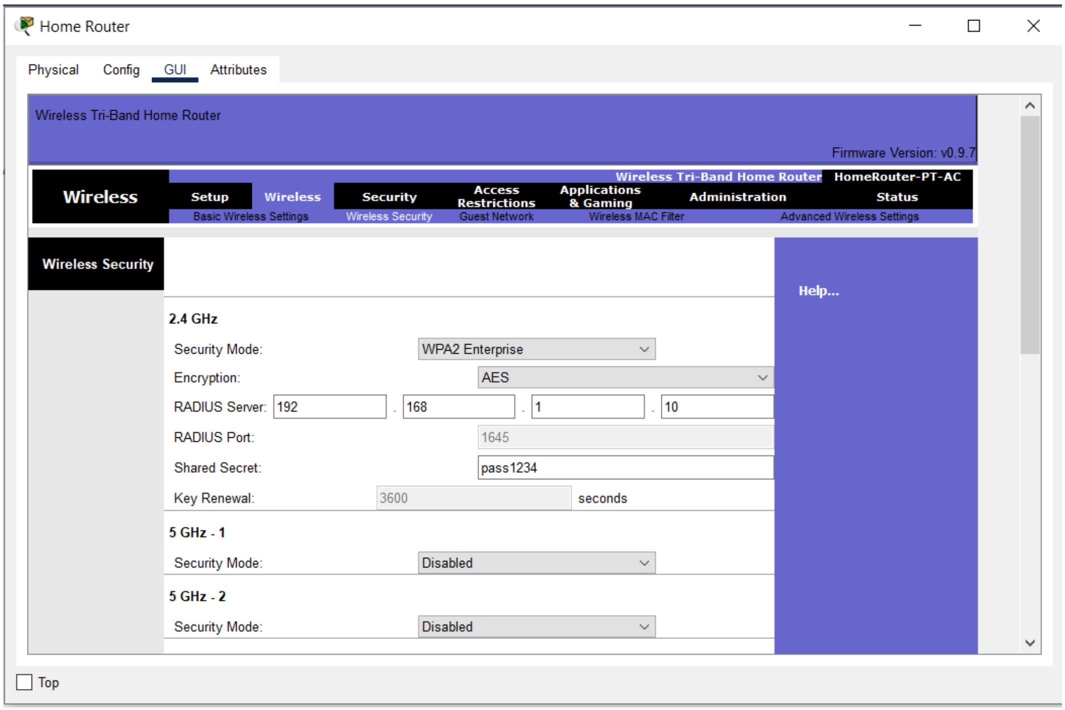
Network Mode: Auto

Network Name (SSID): Default

SSID Broadcast: ● Enabled ○ Disabled

Standard Channel: Auto

□ Top



2.4 GHz

Security Mode: WPA2 Enterprise

Encryption: AES

RADIUS Server: 192 . 168 . 1 . 10

RADIUS Port: 1645

Shared Secret: pass1234

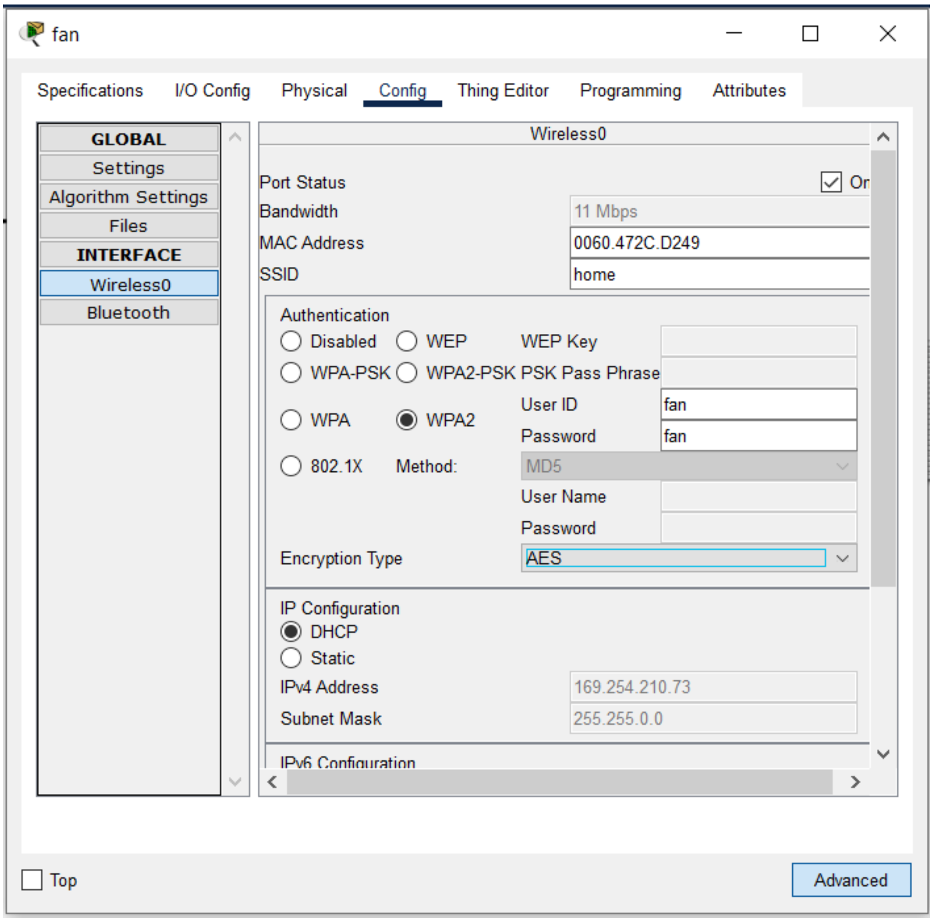
Key Renewal: 3600 seconds

5 GHz - 1

Security Mode: Disabled

5 GHz - 2

Security Mode: Disabled



Wireless0

Port Status ☑ On

Bandwidth 11 Mbps

MAC Address 0060.472C.D249

SSID home

Authentication

○ Disabled ○ WEP WEP Key

○ WPA-PSK ○ WPA2-PSK PSK Pass Phrase

○ WPA ● WPA2 User ID fan

Password fan

○ 802.1X Method: MD5

User Name

Password

Encryption Type AES

IP Configuration

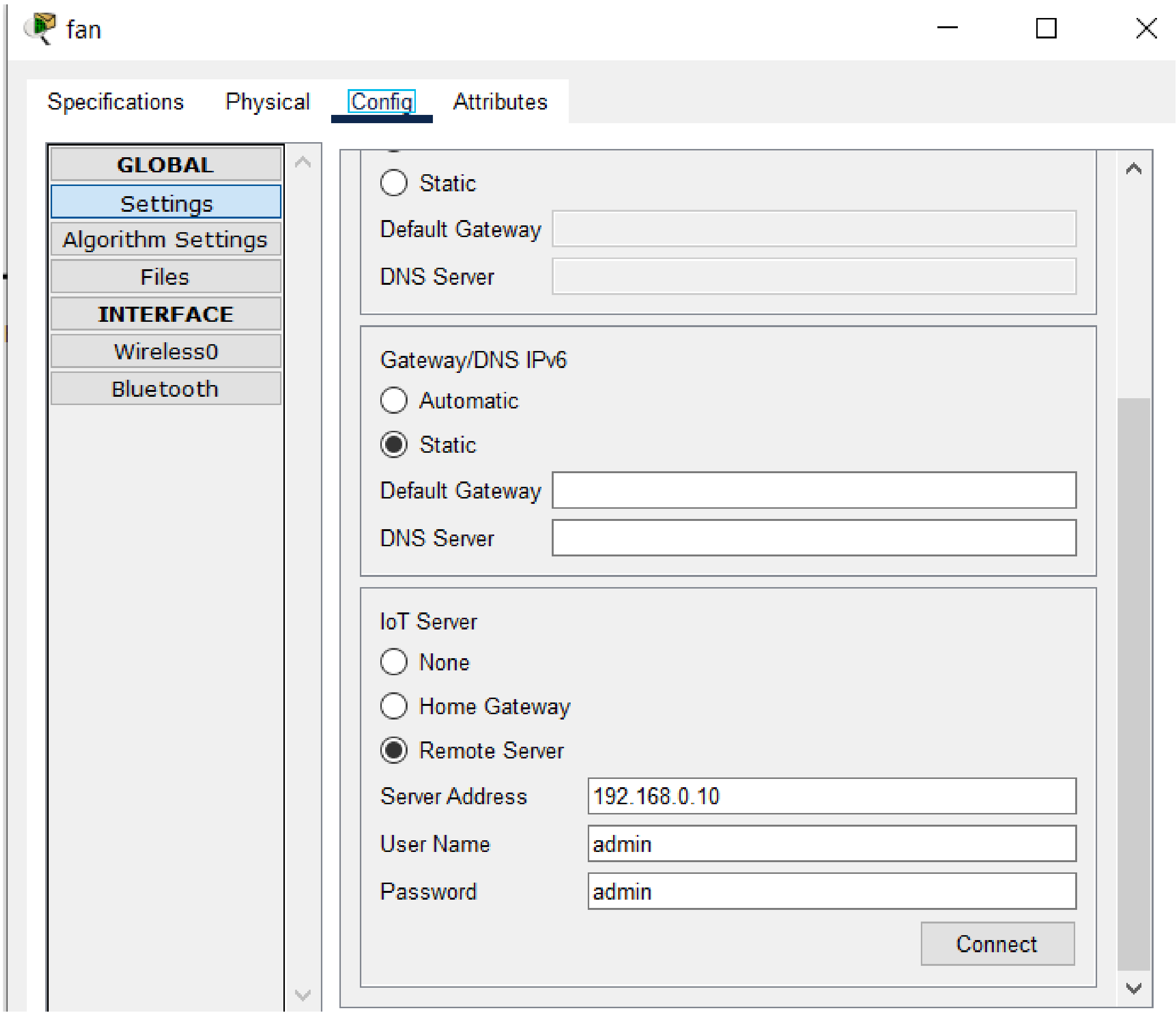
● DHCP

○ Static

IPv4 Address 169.254.210.73

Subnet Mask 255.255.0.0

IPv6 Configuration



fan

○ Static

Default Gateway

DNS Server

Gateway/DNS IPv6

○ Automatic

● Static

Default Gateway

DNS Server

IoT Server

○ None

○ Home Gateway

● Remote Server

Server Address: 192.168.0.10

User Name: admin

Password: admin

Add more IoT devices and monitor their functioning like On and Off.