DNS lookup Program :

```
import  java.net.*;
import  java.util.*;
public class Main {

   public static void main(String[] args) {
      int choice=1; while(choice!=0) {
         String HOST;
         Scanner ch = new Scanner(System.in);
         System.out.print(" 1 --> Enter THe host\n 2 --> Enter the IP address\n 3 --> Exit\nCHOICE : ");
         choice = ch.nextInt();
         Scanner inp = new Scanner(System.in); switch(choice)
         {
            case 1 : System.out.print(" Enter Host Name : " );
               HOST = inp.nextLine();
               try {
                  InetAddress address = InetAddress.getByName(HOST);
                  System.out.println("IP ADDRESS: " + address.getHostAddress());
                  System.out.println("Host Name : " + address.getHostName());
               } catch (UnknownHostException ex) {
                  System.out.print(" NOT Found " + HOST);
               }
               break;

            case 2: System.out.print(" Enter Host IP ADDRESS : " );
               HOST= inp.nextLine(); try{
                  InetAddress address= InetAddress.getByName(HOST);
                  System.out.println("IP ADDRESS: "+ address.getHostAddress());
                  System.out.println("Host Name : "+ address.getHostName());
               }
               catch(UnknownHostException ex){
                  System.out.print(" NOT Found "+ HOST);
               }
               break;

            case 3:
               choice=0;
               break;

            default:
               System.out.println("Invalid Choice, Try Again"); break;
         }
      }
   }
}
```

Output :

```
1 --> Enter THe host
2 --> Enter the IP address
3 --> Exit
```

CHOICE : 1

Enter Host Name : www.facebook.com
IP ADDRESS: 31.13.79.35
Host Name : www.facebook.com
1 --> Enter THe host
2 --> Enter the IP address
3 --> Exit
CHOICE : 2
Enter Host IP ADDRESS : 31.13.79.35
IP ADDRESS: 31.13.79.35
Host Name : edge-star-mini-shv-02-bom1.facebook.com
1 --> Enter THe host
2 --> Enter the IP address
3 --> Exit
CHOICE : 3

Program :

```
import java.util.Scanner;

class subnet { public static void
  main(String args[]) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter the ip address: ");
    String ip = sc.nextLine();
    String split_ip[] = ip.split("\\."); // SPlit the string after every .
    String split_bip[] = new String[4]; // split binary ip
    String bip = "";
    for (int i = 0; i < 4; i++) { split_bip[i] =
      appendZeros(Integer.toBinaryString(Integer.parseInt(split_ip[i]))); // "18" => 18 =>
10010 =>
      // 00010010
      bip += split_bip[i];
    }
    System.out.println("IP in binary is " + bip);
    System.out.print("Enter the number of addresses: ");
    int n = sc.nextInt();

    // Calculation of mask
    int bits = (int) Math.ceil(Math.log(n) / Math.log(2));
    System.out.println("Number of bits required for address = " + bits); int
    mask = 32 - bits;
    System.out.println("The subnet mask is = " + mask);

    // Calculation of first address and last address int fbip[] = new int[32];
    for (int i = 0; i < 32; i++) fbip[i] = (int) bip.charAt(i) - 48; // convert
    cahracter 0,1 to integer 0,1
    for (int i = 31; i > 31 - bits; i--)// Get first address by ANDing last n bits with 0
      fbip[i] &= 0;
    String fip[] = { "", "", "", "" }; for (int i = 0; i
    < 32; i++) fip[i / 8] = new String(fip[i / 8] +
    fbip[i]); System.out.print("Subnet address
    is = "); for (int i = 0; i < 4; i++) {
      System.out.print(Integer.parseInt(fip[i],   2));
      if (i != 3)
        System.out.print(".");
    }
    System.out.println();

    int lbip[] = new int[32]; for (int i = 0; i < 32; i++) lbip[i] = (int)
    bip.charAt(i) - 48; // convert cahracter 0,1 to integer 0,1
    for (int i = 31; i > 31 - bits; i--)// Get last address by ORing last n bits with 1
      lbip[i] |= 1;
    String lip[] = { "", "", "", "" }; for (int i = 0; i <
    32; i++) lip[i / 8] = new String(lip[i / 8] +
    lbip[i]);          System.out.print("Broadcast
    address is = "); for (int i = 0; i < 4; i++) {
      System.out.print(Integer.parseInt(lip[i],   2));
      if (i != 3)
```

```
            System.out.print(".");
        }
        System.out.println();
    }

    static String appendZeros(String s) {
        String temp = new String("00000000");
        return temp.substring(s.length()) + s;
    }
}
```

Output:

Enter the ip address: 10.10.10.222

IP in binary is 00001010000010100000101011011110

Enter the number of addresses: 7

Number of bits required for address = 3

The subnet mask is = 29

Subnet address is = 10.10.10.216

Broadcast address is = 10.10.10.223

Selective Repeat:

Server :

```
import    java.net.*;
import      java.io.*;
public class server {
public static void main(String[] args) throws IOException {
      ServerSocket ss = new ServerSocket(3333);
      Socket s = ss.accept();
      DataInputStream din = new DataInputStream(s.getInputStream());
      DataOutputStream dout = new DataOutputStream(s.getOutputStream());
      int[] arr = { 10, 20, 30, 40, 50, 60, 70 }; int k = arr.length; dout.write(k);
      for (int i = 0; i < arr.length; i++) {
         System.out.println("sending frame:" + arr[i]); dout.write(arr[i]);
      }
      int rd = din.read();
      System.out.println("ERROR frame recieved aft ack" + (rd +
      1)); dout.write(arr[rd]); din.close(); s.close();
      ss.close();
   }
}
```

Client :

```
import    java.io.DataInputStream;
import java.io.DataOutputStream;
import            java.io.IOException;
import    java.net.Socket;    public
class client {
public static void main(String[] args) throws IOException {
      Socket s = new Socket("localhost", 3333);
      DataInputStream din = new DataInputStream(s.getInputStream());

      DataOutputStream dout = new DataOutputStream(s.getOutputStream());
      int y = din.read(); System.out.println(); int[] arr = new int[y]; for (int i = 0;
      i < y; i++) { arr[i] = din.read();
         System.out.println("recieving frame: " + arr[i]);
      }
      arr[4] = -1; int
      temp = 0;
      for (int i = 0; i < y; i++) {
         if (arr[i] == -1) { temp
         = i;
            System.out.println("eror in frame: " + arr[i]);
         } else {
            System.out.println("reciving frame:" + arr[i]);
         }
      }
```

```
      dout.write(temp); arr[temp]
      = din.read();
      System.out.println("resent frame:" + temp);
      System.out.println("final frame");
      System.out.println(arr[temp]);
      din.close(); s.close();
   }
}
```

output:

* server:
* sending frame:10
* sending frame:20
* sending frame:30
* sending frame:40
* sending frame:50
* sending frame:60
* sending frame:70
* ERROR frame recieved aft ack5
*
* client:
* recieving frame: 10
* recieving frame: 20
* recieving frame: 30

* recieving frame: 40
* recieving frame: 50
* recieving frame: 60
* recieving frame: 70
* reciving frame:10
* reciving frame:20
* reciving frame:30
* reciving frame:40
* eror in frame: -1
* reciving frame:60
* reciving frame:70
* resent frame:4
* final frame
* 50

GO back N:

server

```
import    java.net.*;
import      java.io.*;
public class server {
public static void main(String[] args) throws IOException {
      ServerSocket ss = new ServerSocket(3000);
      Socket s = ss.accept();
      DataInputStream din = new DataInputStream(s.getInputStream());
      DataOutputStream dout = new DataOutputStream(s.getOutputStream());
      int[] arr = { 10, 20, 30, 40, 50, 60, 70 }; int k = arr.length; dout.write(k);
      for (int i = 0; i < arr.length; i++) {
```

```
         System.out.println("sending frame:" + arr[i]); dout.write(arr[i]);
      }
      int rd = din.read();
      System.out.println("ERROR frame recieved aft ack" + (rd + 1));
      for (int i = rd; i < arr.length; i++) {
         System.out.println("sending frame:" + arr[i]); dout.write(arr[i]);
      }
      din.close(); s.close();
      ss.close();
   }
}
```

client

```
import    java.io.DataInputStream;
import java.io.DataOutputStream;
import            java.io.IOException;
import    java.net.Socket;    public
class client {
public static void main(String[] args) throws IOException {
      Socket s = new Socket("localhost", 3000);
      DataInputStream din = new DataInputStream(s.getInputStream());
      DataOutputStream dout = new DataOutputStream(s.getOutputStream());
      int y = din.read(); System.out.println(); int[] arr = new int[y]; for (int i = 0;
      i < y; i++) { arr[i] = din.read();
         System.out.println("recieving frame: " + arr[i]);
      }
      arr[4] = -1; int
      temp = 4;
      for (int i = 0; i < y; i++) {
         if (arr[i] == -1) { temp
         = i;
            System.out.println("eror in frame: " + arr[i]);
         } else {
            System.out.println("reciving frame:" + arr[i]);
         }
      }
      dout.write(temp);
      for (int i = temp; i < arr.length; i++) { arr[i]
         = din.read();
         System.out.println("resent frame:" + i);
      }
      System.out.println("final  frame");
      System.out.println(arr[temp + 2]);
      din.close(); s.close();
   }
}
```

Output:

server sending
frame:10 sending
frame:20 sending
frame:30 sending
frame:40 sending
frame:50 sending
frame:60 sending
frame:70
ERROR frame recieved aft ack5
sending frame:50 sending frame:60 sending frame:70
client recieving frame: 10 recieving frame: 20 recieving frame: 30 recieving frame: 40 recieving frame: 50 recieving frame: 60 recieving frame: 70
reciving frame:10 reciving frame:20 reciving frame:30 reciving frame:40 eror in frame: -1 reciving frame:60 reciving frame:70 resent frame:4 resent frame:5 resent frame:6 final frame
70

Program:

server :

```
import    java.net.*;
import      java.io.*;
public class server {
public static void main(String[] args) throws IOException {
      ServerSocket ss = new ServerSocket(3333);
      Socket s = ss.accept();
      DataInputStream din = new DataInputStream(s.getInputStream());
      DataOutputStream dout = new DataOutputStream(s.getOutputStream());
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      String str = "", str1 = "";
      while  (str  !=  "Bye")  {
      str1 = din.readUTF();
         System.out.println("client: " + str1);
         System.out.println("server: "); str =
         br.readLine();
         dout.writeUTF(str);
      }
      s.close();
      ss.close();
   }
}
```

client :

```
import     java.io.*;
import  java.net.*;
public class client {
public static void main(String[] args) throws IOException {
      Socket s = new Socket("localhost", 3333);
      DataInputStream din = new DataInputStream(s.getInputStream());
      DataOutputStream dout = new DataOutputStream(s.getOutputStream());
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      String str = "", str1 = ""; while
      (str != "Bye") {
         System.out.println("client: ");
         str = br.readLine();
         dout.writeUTF(str); dout.flush();
         str1 = din.readUTF();

         System.out.println("server: " + str1);
      }
      s.close();
   }
}
```

Output:

client:
hi server: hello
client: how are
you? server: I
am fine client:
Bye
//server client: hi server: hello client: how are you? server: I am fine client: Bye server: Bye

Program :

```
import java.util.*;
 public class Dijkstra
{
public int distance[] = new int[10]; public int cost[][] = new int[10][10]; public void calc(int n,int s)
{
int flag[] = new int[n+1];
int i,minpos=1,k,c,minimum; for(i=1;i<=n;i++)
{
flag[i]=0; this.distance[i]=this.cost[s][i];
}
c=2; while(c<=n)
{
minimum=99; for(k=1;k<=n;k++)
{
if(this.distance[k]<minimum && flag[k]!=1)
{
minimum=this.distance[i]; minpos=k;
}
}
flag[minpos]=1; c++;
for(k=1;k<=n;k++)
{
if(this.distance[minpos]+this.cost[minpos][k] < this.distance[k] && flag[k]!=1 )
this.distance[k]=this.distance[minpos]+this.cost[minpos][k];
}
}
}
public static void main(String args[])
{
int nodes,source,i,j;
Scanner in = new Scanner(System.in); System.out.println("Enter the Number of Nodes \n"); nodes =
in.nextInt();
Dijkstra d = new Dijkstra();
System.out.println("Enter the Cost Matrix Weights: \n"); for(i=1;i<=nodes;i++)
for(j=1;j<=nodes;j++)
{
d.cost[i][j]=in.nextInt();
if(d.cost[i][j]==0) d.cost[i][j]=999;
}
System.out.println("Enter the Source Vertex :\n"); source=in.nextInt();
d.calc(nodes,source);
System.out.println("The Shortest Path from Source \t"+source+"\t to all other vertices are :\n");
for(i=1;i<=nodes;i++) if(i!=source)
System.out.println("source :"+source+"\t destination :"+i+"\t MinCost is:"+d.distance[i]+"\t");
}
}
```

Output :

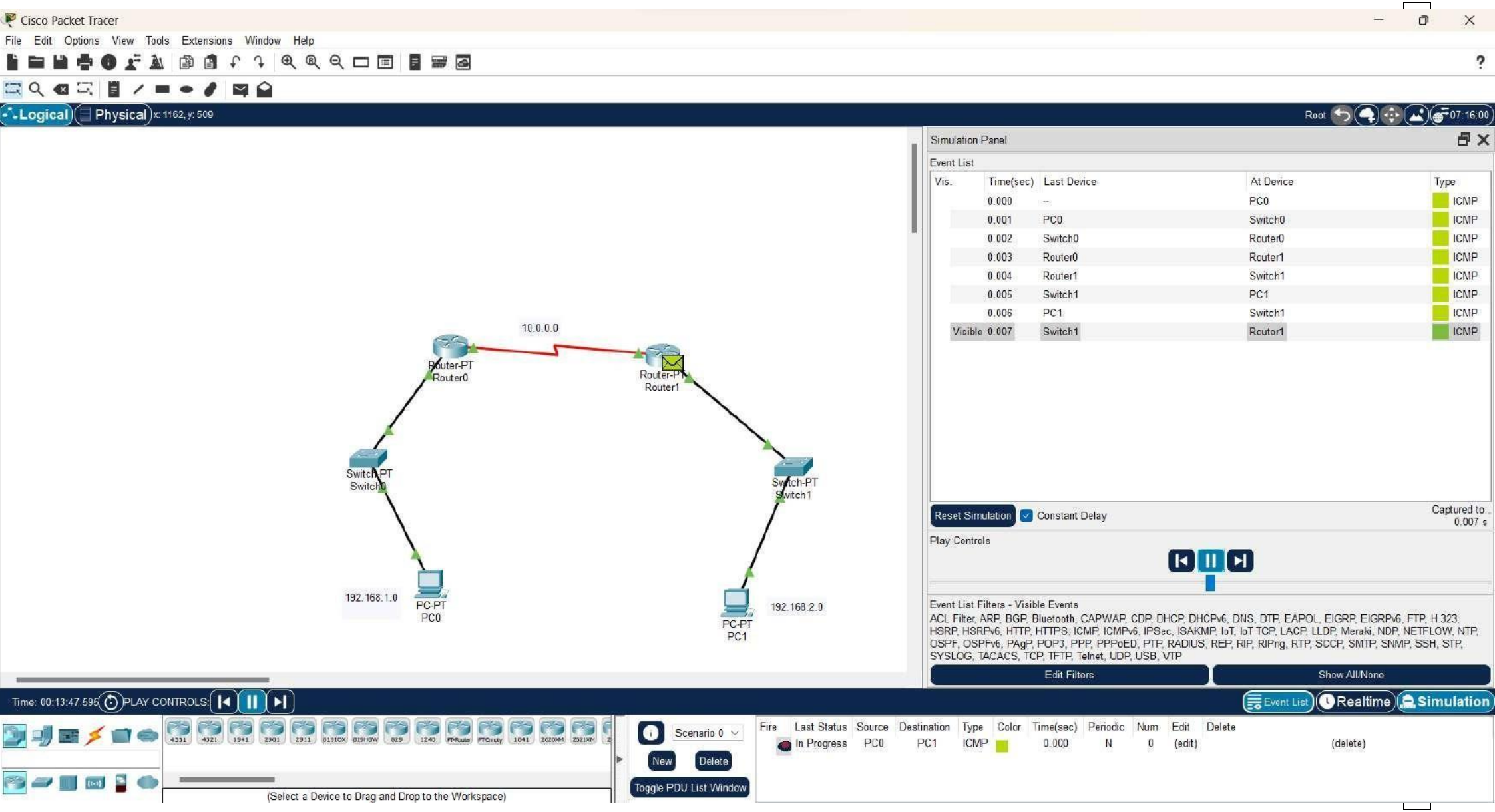
Enter the Number of Nodes

5
Enter the Cost Matrix Weights:

0 1 0 2 0
1 0 1 0 4
0 1 0 0 3
2 0 0 0 3
0 4 3 3 0
Enter the Source Vertex :

1
The Shortest Path from Source 1 to all other vertices are :

source :1 destination :2 MinCost is:1 source
:2 destination :3 MinCost is:2 source :3
destination :4 MinCost is:2 source :4
destination :5 MinCost is:5


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical Physical x: 1162, y: 509
Root 07:16:00
10.0.0.0
Router-PT Router0
Router-PT Router1
Switch-PT Switch0
Switch-PT Switch1
192.168.1.0
PC-PT PC0
PC-PT PC1
192.168.2.0
Simulation Panel
Event List
Vis. Time(sec) Last Device At Device Type
0.000 -- PC0 ICMP
0.001 PC0 Switch0 ICMP
0.002 Switch0 Router0 ICMP
0.003 Router0 Router1 ICMP
0.004 Router1 Switch1 ICMP
0.005 Switch1 PC1 ICMP
0.006 PC1 Switch1 ICMP
Visible 0.007 Switch1 Router1 ICMP
Reset Simulation
Constant Delay
Captured to: 0.007 s
Play Controls
Event List Filters - Visible Events
ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP
Edit Filters
Show All/None
Time: 00:13:47.595 PLAY CONTROLS:
Event List Realtime Simulation
(Select a Device to Drag and Drop to the Workspace)
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
In Progress PC0 PC1 ICMP 0.000 N 0 (edit) (delete)