21BDS0340

Abhinav Dinesh Srivatsa

Computer Networks Lab

Assignment – V

**Question 1**

Aim:

Develop a program to find and display the class of IPv4 (Classes A-E) from set of input addresses

Code:

import java.util.Scanner;

public class question1 {

public static String integerToBits(int n) {

String bin = "";

while (n != 0) {

bin = Integer.toString(n % 2) + bin;

n /= 2;

}

return bin;

}

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

String ip = s.nextLine().trim();

int netId = Integer.parseInt(ip.split("\\.")[0]);

String bits = integerToBits(netId);

char ipClass = '\0';

if (bits.startsWith("0"))

ipClass = 'A';

else if (bits.startsWith("10"))

ipClass = 'B';

else if (bits.startsWith("110"))

ipClass = 'C';

else if (bits.startsWith("1110"))

ipClass = 'C';

else if (bits.startsWith("1111"))

ipClass = 'E';

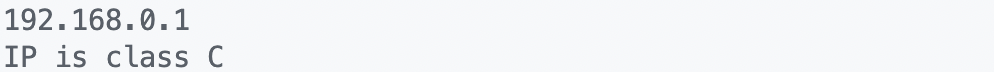
System.out.println("IP is class " + ipClass);

s.close();

}

}

Output:



**Question 2**

Aim:

Write a program that converts a 32-bit IP address to its equivalent dotted decimal number format

Code:

import java.util.Scanner;

// 11111111111111110000000010101001

public class question2 {

public static int bitsToInteger(String bits) {

int sum = 0;

for (int x = 0; x < bits.length(); x++)

sum += (bits.charAt(x) - '0') \* Math.pow(2, bits.length() - x - 1);

return sum;

}

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

String ip = s.nextLine().trim();

String bytes[] = ip.split("(?<=\\G.{8})");

int c = 0;

for (String str : bytes)

if (c++ == 0)

System.out.print(bitsToInteger(str));

else

System.out.print("." + bitsToInteger(str));

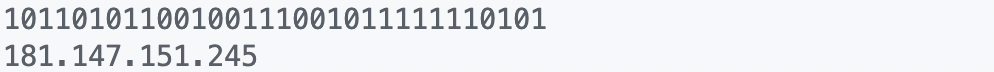
System.out.println("");

s.close();

}

}

Output:

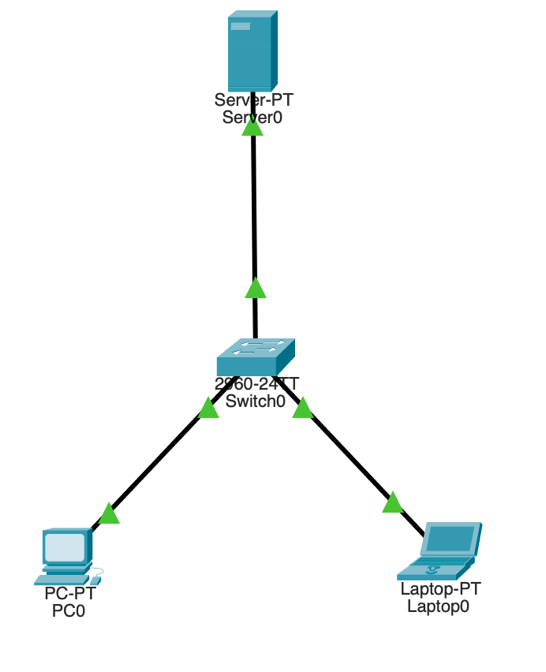


**Question 3**

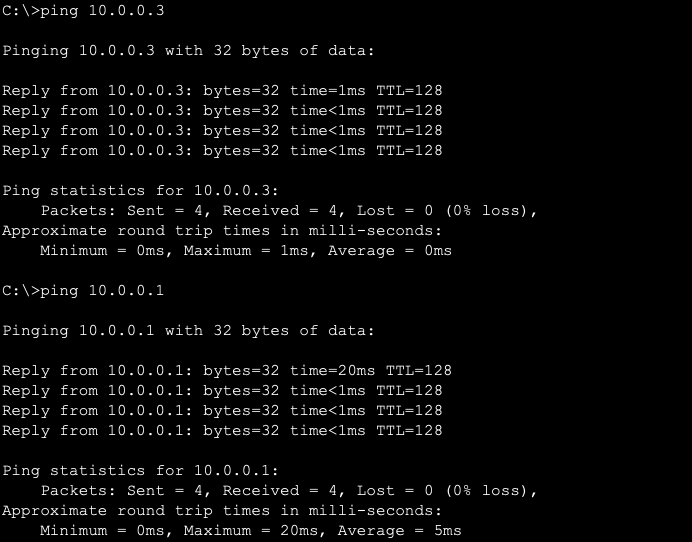
Aim:

Implement the NAT in static dynamic and port type using the packet tracer for topology using 1 PC, 1 laptop, 1 switch and 1 web server

Topology:



Testing Connections:

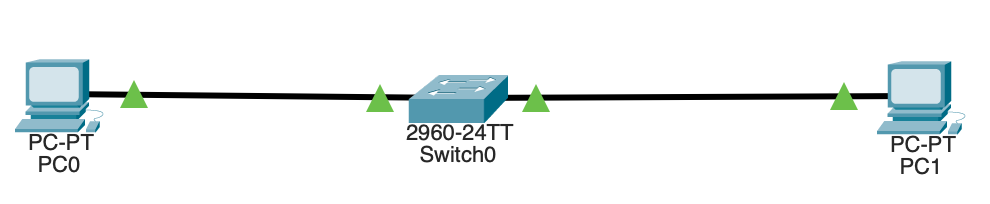


**Question 4**

Aim:

Simulate a IPv6 web traffic using simple network topology implemented in packet tracer. Test the connections using PDU’s

Topology:

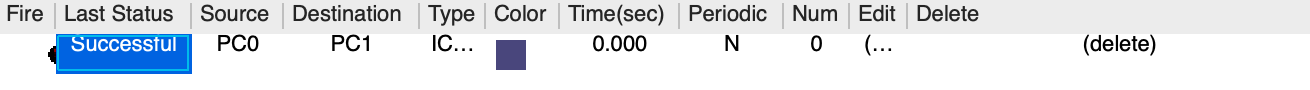


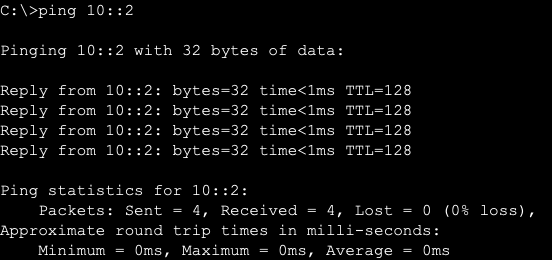
IPv6 Assignment:

A screenshot of a computer

Description automatically generated with low confidence

Testing Connection:





**Question 5**

Aim:

Create a socket with the same machine for client and server program. Client can initiate a conversation to which a server can respond and keep listening for new messages. Client program will terminate if user enters “bye” or “exit”.

Code:

Server:

import socket

def server\_program():

host = socket.gethostname()

port = 4500

server\_socket = socket.socket()

server\_socket.bind((host, port))

server\_socket.listen(1)

print("Server started. Waiting for connections...")

client\_socket, address = server\_socket.accept()

print("Connection from:", address)

while True:

data = client\_socket.recv(1024).decode('utf-8')

print("Client:", data)

if data.lower() == "bye" or data.lower() == "exit":

break

message = input("Server: ")

client\_socket.send(message.encode('utf-8'))

client\_socket.close()

server\_socket.close()

if \_\_name\_\_ == '\_\_main\_\_':

server\_program()

Client:

import socket

def client\_program():

host = socket.gethostname()

port = 4500

client\_socket = socket.socket()

client\_socket.connect((host, port))

while True:

message = input("Client: ")

client\_socket.send(message.encode('utf-8'))

data = client\_socket.recv(1024).decode('utf-8')

print("Server:", data)

if data.lower() == "bye" or data.lower() == "exit":

break

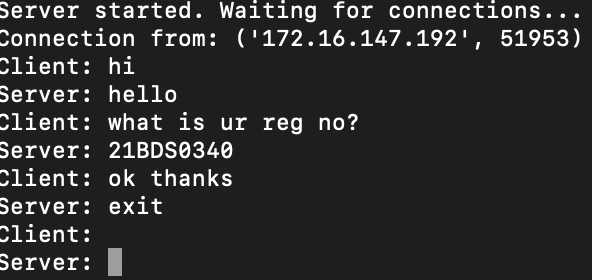
client\_socket.close()

if \_\_name\_\_ == '\_\_main\_\_':

client\_program()

Output:

Server:



Client:

