**Practical - 03**

**Title: - Program to assign IPv4 Addresses to nodes.**

**Aim: -** To assign IPv4 Addresses to nodes.

**Lab Objectives: -**

To get familiarize with a Network Simulation Tool

**Description: -**

**IP Addresses**

The network layer is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links). The network layer adds a header that includes the logical addresses of the sender and receiver to the packet coming from the upper layer. One of the functions of the network layer is to provide a routing mechanism.

Logical addresses are necessary for universal communications that are independent of underlying physical networks. Physical addresses are not adequate in an internetwork environment where different networks can have different address formats. A universal addressing system is needed in which each host can be identified uniquely, regardless of the underlying physical network. The logical addresses are designed for this purpose.

A logical address on the Internet is currently a 32-bit address that can uniquely define a host connected to the Internet. No two publicly addressed and visible hosts on the Internet can have the same IP address. For example, 172.18.4.10

**Types of IP Addresses**

* IP v4
* IP v6

**IP v4 Addresses**

An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a device. They are unique in the sense that each address defines one, and only one, connection to the Internet. Two devices on the Internet can never have the same address at the same time.

The IPv4 addresses are universal in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet. An address space is the total number of addresses used by the protocol.

If a protocol uses N bits to define an address, the address space is 2N because each bit can have two different values (0 or 1) and N bits can have 2N values. IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4 billion)

There are two prevalent notations to show an IPv4 address:

**Binary Notation**

The following is an example of an IPv4 address in binary notation:

01110101 10010101 00011101 00000010

**Dotted-decimal Notation.**

The following is the dotted-decimal notation of the above address - 117.149.29.2

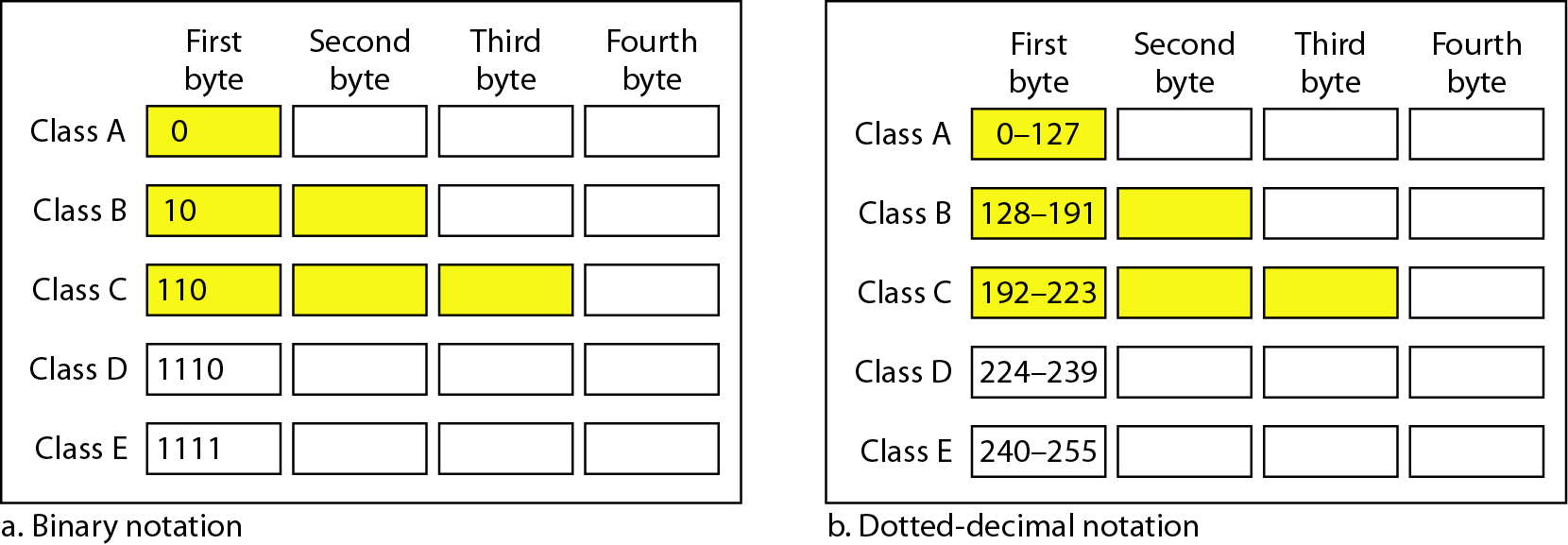
**IPv4 Addresses: Classful Addressing**

In classful addressing, the address space is divided into five classes: A, B, C, D and E.

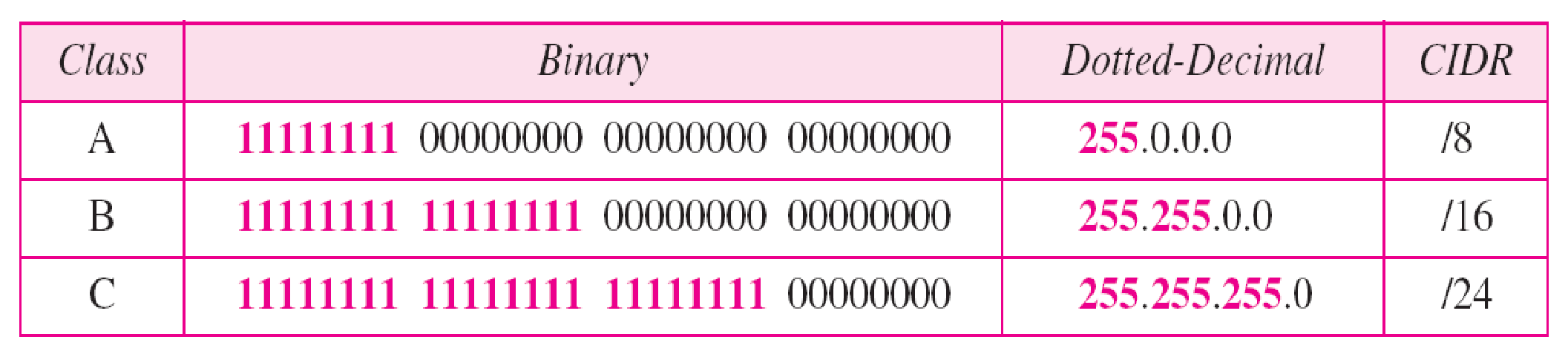
Each class occupies some part of the address space.

In classful addressing, an IP address in class A, B, or C is divided into netid and hostid.

These parts are of varying lengths, depending on the class of the address.



**Mask** - a 32-bit number made of contiguous 1 followed by contiguous 0. The mask can help us to find the netid and the hostid.



**IPv4 Addresses: Classless Addressing**

To overcome address depletion and give more organizations access to the Internet, classless addressing was designed and implemented.

In this scheme, there are no classes, but the addresses are still granted in blocks.

**Mask** -a mask in classless addressing is a 32-bit number in which the n leftmost bits are 1s and the 32 - n rightmost bits are 0s. In classless addressing the mask for a block can take any value from 0 to 32.

**Network Addresses**

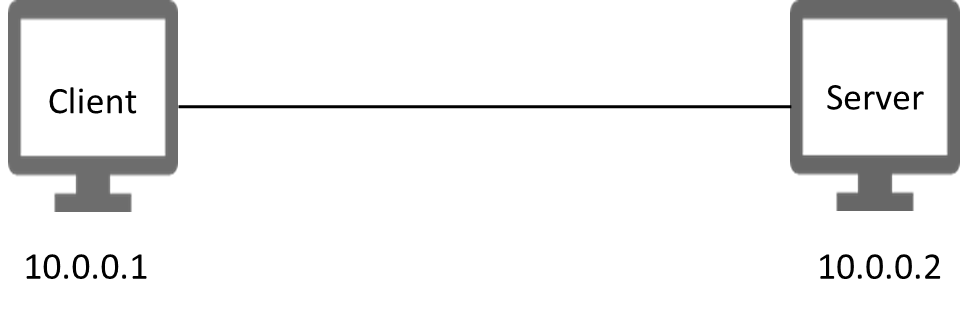
When an organization is given a block of addresses, the organization is free to allocate the addresses to the devices that need to be connected to the Internet.

The first address in the class is normally treated as a special address called as the network address and defines the organization network.

It defines the organization itself to the rest of the world.

**Exercise**

1. Implement the following topology and assign class B addresses to the nodes.



**Code :**  **Created Onkar Malawade A-23-0075 A2-35**

/-- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; --/

// add required header files

#include "ns3/core-module.h"

#include "ns3/network-module.h"

#include "ns3/internet-module.h"

#include "ns3/point-to-point-module.h"

#include "ns3/applications-module.h"

#include "ns3/netanim-module.h"

#include "ns3/csma-module.h"

#include "ns3/ipv4-global-routing-helper.h"

// Adding namespace declaration

using namespace ns3;

//Define log component where log msgs will be saved

NS\_LOG\_COMPONENT\_DEFINE("p2pExercise");

// Main function

int main(int argc, char \*argv[]){

// declare number of nodes in bus Topology

// read the command line arguments

CommandLine cmd(\_FILE\_);

// Process the command line arguments

cmd.Parse(argc, argv);

// Set time Resolution to 1 nano second

Time::SetResolution(Time::NS);

// Logging

LogComponentEnable("UdpEchoClientApplication",LOG\_LEVEL\_INFO);

LogComponentEnable("UdpEchoServerApplication",LOG\_LEVEL\_INFO);

// Create NodeContainer object to store our nodes

NodeContainer nodes;

// Create 2 nodes

nodes.Create(2);

// create object of the point-to-point helper object class to configure net device and the channels

PointToPointHelper pointToPoint;

// Configure the net Device

pointToPoint.SetDeviceAttribute("DataRate", StringValue("5Mbps")); // Set Data Rate

// Configure the Channel

pointToPoint.SetChannelAttribute("Delay", StringValue("2ms")); // Set Delay Attribute

// Install net devices on nodes

NetDeviceContainer devices;

devices=pointToPoint.Install(nodes);// install netdevices on node and connect with the Channels

// Configure and Install protocol suits on nodes

InternetStackHelper stack;

stack.Install (nodes);

// configure network IP address and subnet mask for network

Ipv4AddressHelper address;

// set data

address.SetBase("40.120.80.0","255.255.240.0");

// Assign IP addresses to the interfaces of netDevices

Ipv4InterfaceContainer interfaces = address.Assign(devices);

// Configure our Applications

// Configure UDPEchoServerApplication

UdpEchoServerHelper echoServer(9); // Setting port number of server application

// Application Container create object to store server application and install on node(1)

ApplicationContainer serverApp = echoServer.Install(nodes.Get(1)); // indexed 1 server

// Configure start and stop time of server Application

serverApp.Start(Seconds(1.0)); // server app should start first

serverApp.Stop(Seconds(10.0)); // server app should stop

// Configure UdpEchoClientApplication

UdpEchoClientHelper echoClient(interfaces.GetAddress(1),9);

// Configure the attribute of client Application

echoClient.SetAttribute("MaxPackets", UintegerValue (1));

echoClient.SetAttribute("Interval", TimeValue (Seconds(1.0)));

echoClient.SetAttribute("PacketSize", UintegerValue (1024));

// Install Client Application on Node 0

ApplicationContainer clientApp = echoClient.Install(nodes.Get(0));

// Configure Start and Stop Time

clientApp.Start(Seconds(2.0));

clientApp.Stop(Seconds(10.0));

// Enables Routing IP Address:- "40.120.80.0"

Ipv4GlobalRoutingHelper::PopulateRoutingTables();

// for Running the code 4056

AnimationInterface anim("p2pAniExcer.xml");

anim.SetConstantPosition(nodes.Get(0),20.0,30.0);

anim.SetConstantPosition(nodes.Get(1),40.0,30.0);

// Simulation on Run and start

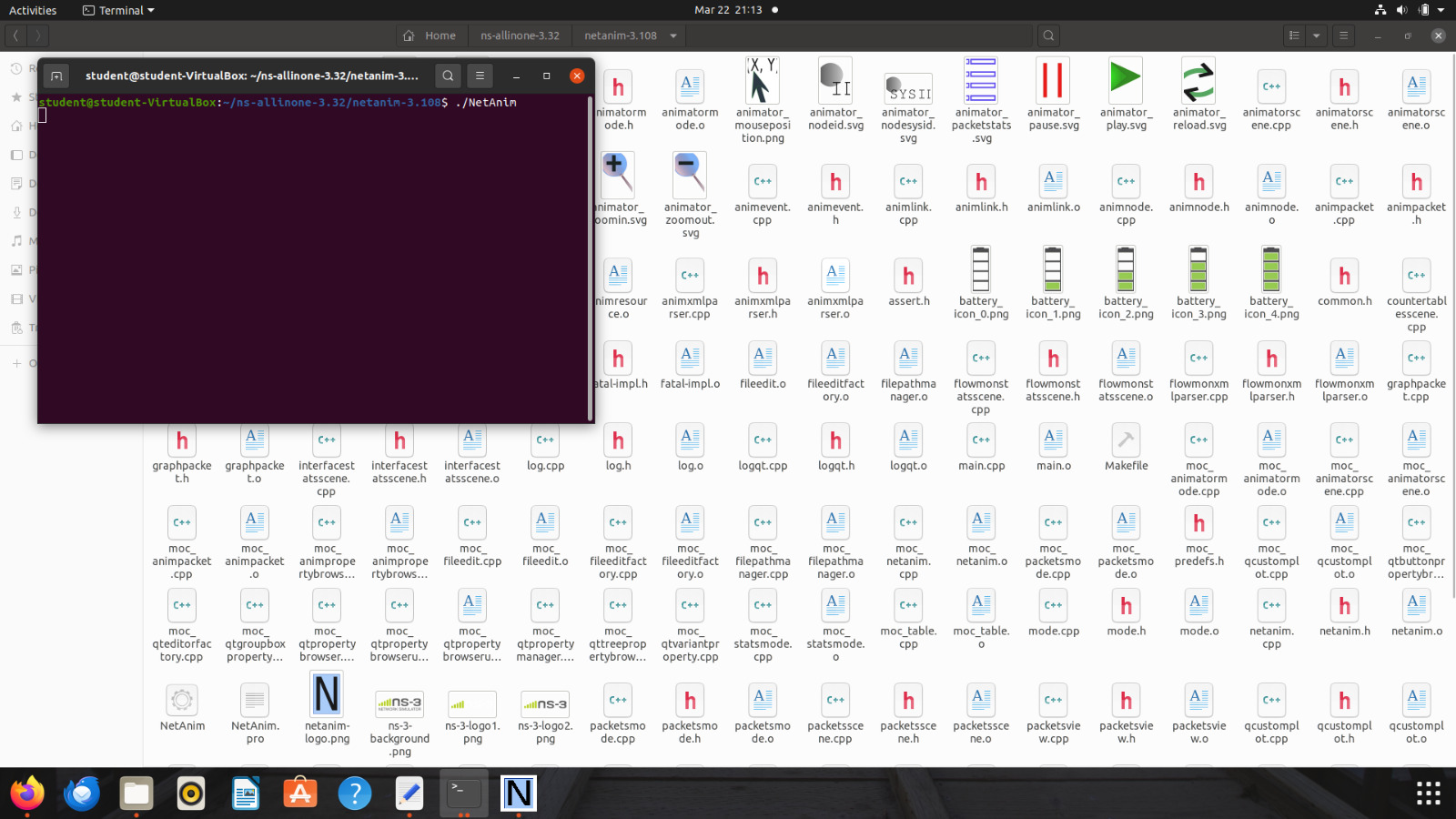
Simulator::Run();

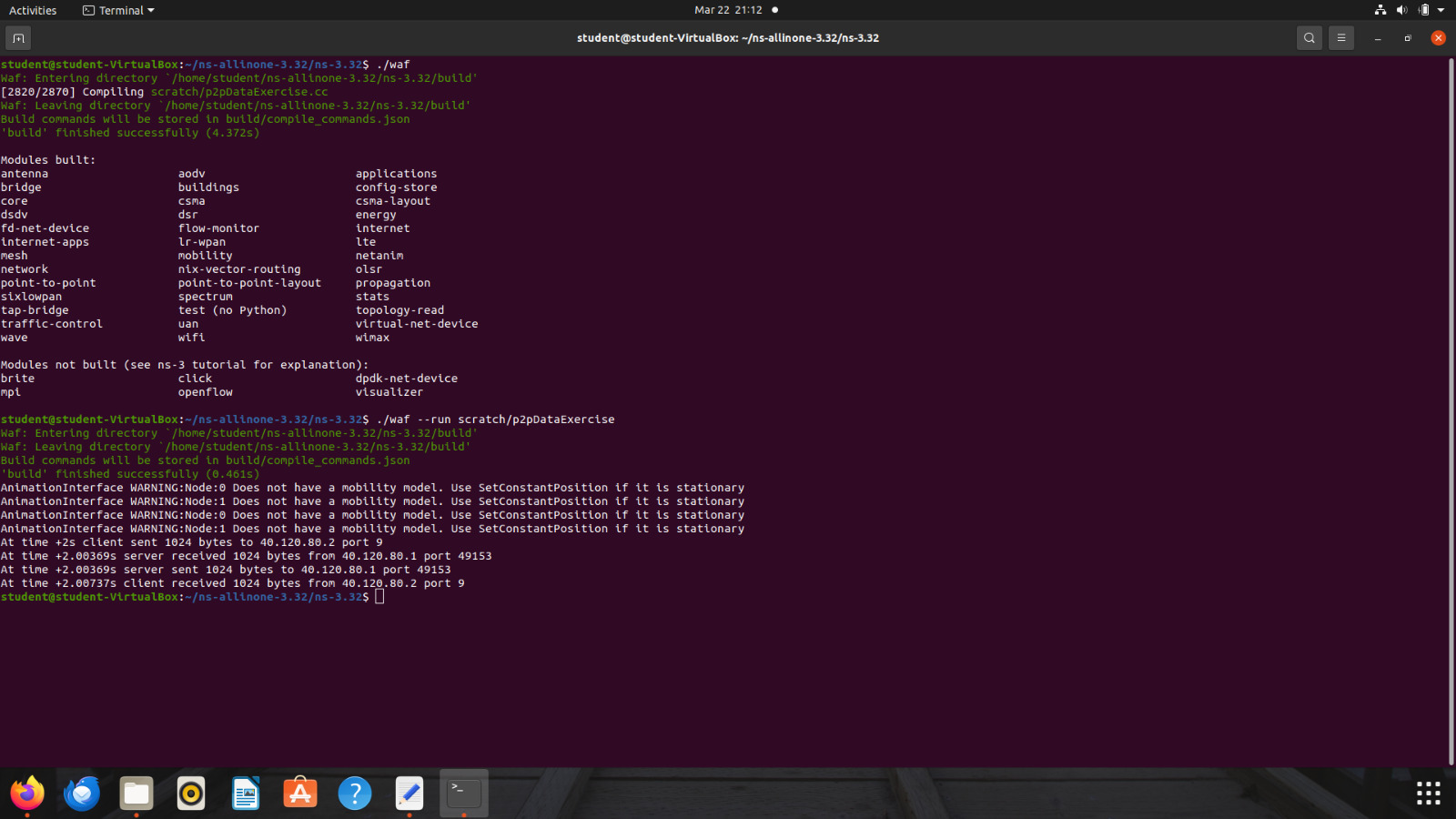
// Destory this Resourses

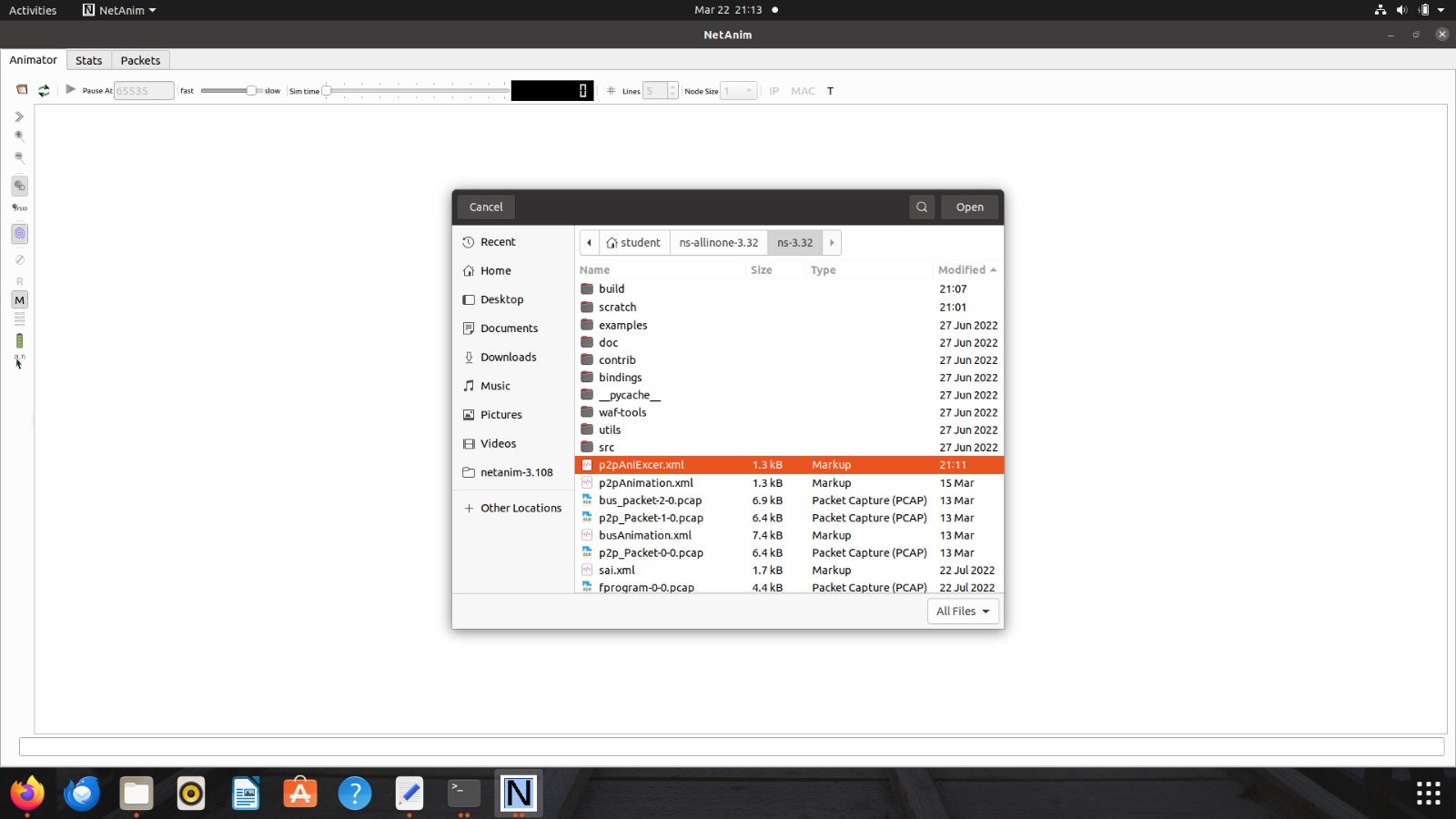
Simulator::Destroy();

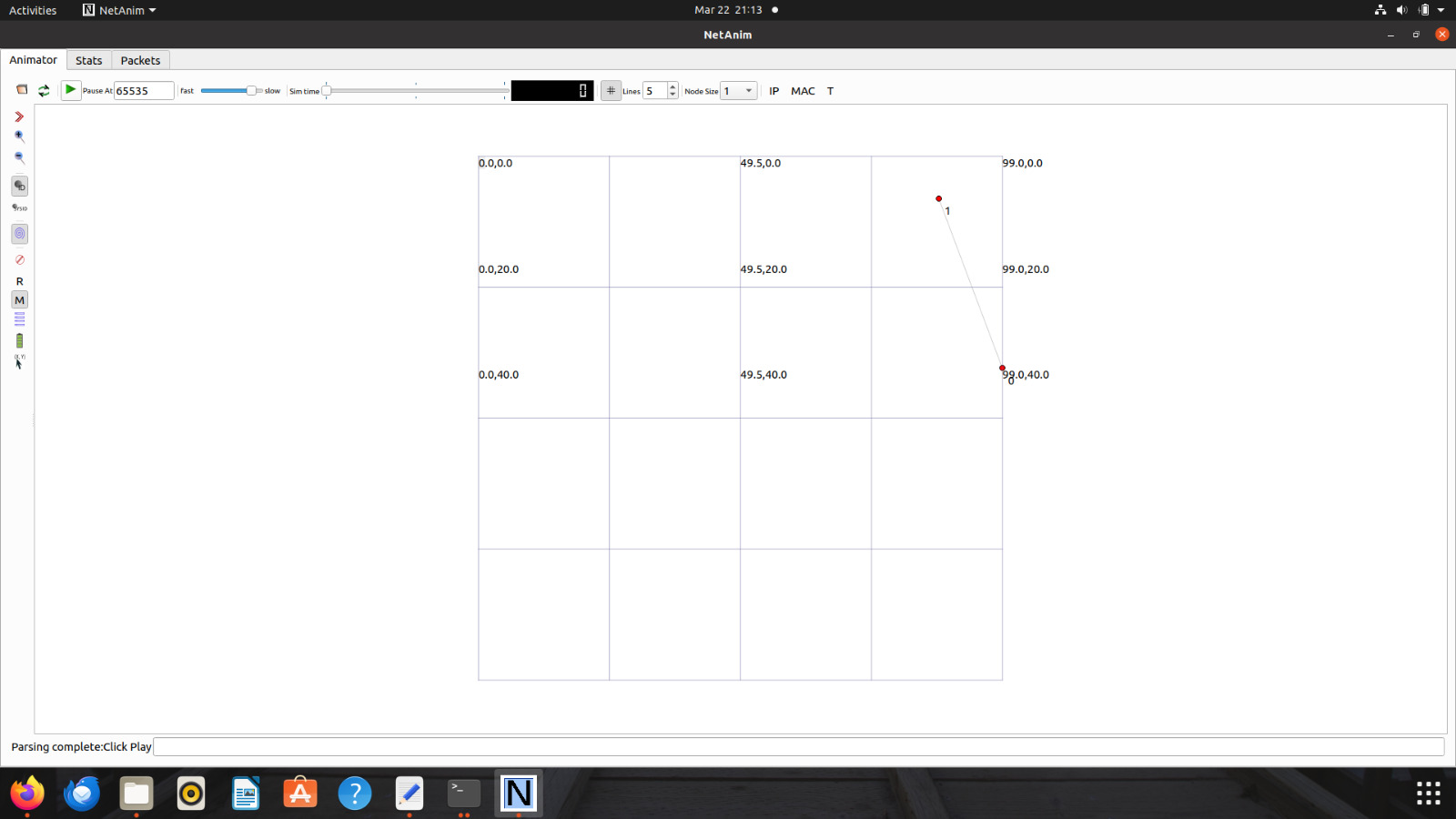
return 0;

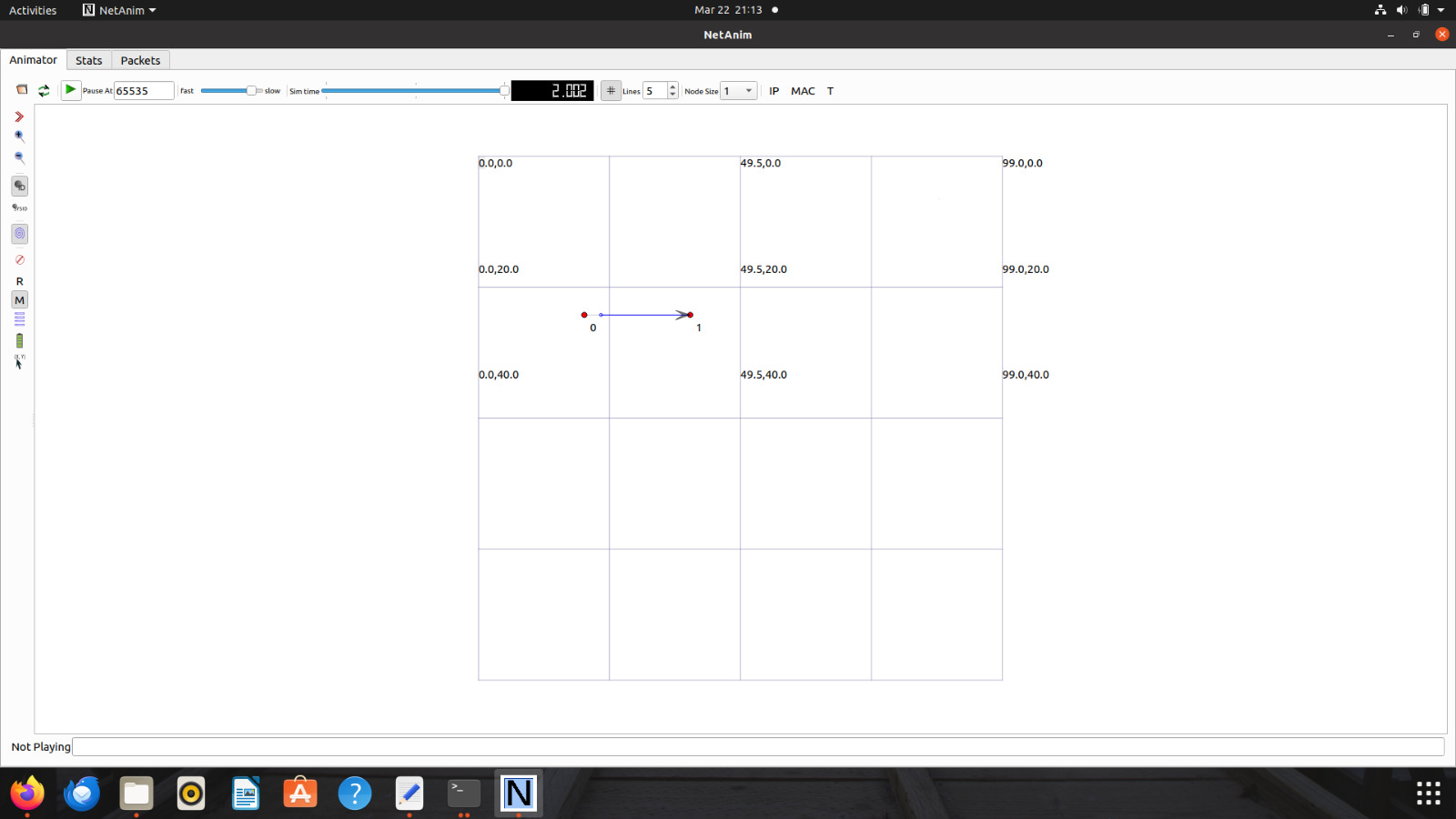
}

**Output**









**Conclusion:** Implemented UDP Client server topology, assigned IP v4 addresses to the nodes and simulated traffic between in UDP client server.

**After performing this Practical/lab, students are expected to answer the following questions.**

1. What is the size of IPv4 address?
2. What is the range of addresses in Class B? What is the default mask of Class B Addresses in CIDR notation and dotted decimal notation.
3. In a block of addresses, we know the IP address of one host is 182.44.82.16/26. What is the first address (network address) , the last address and total number of addresses in this block?

**References**