Q1. Point-To-Point Network Simulation Using TCP

```
#include "ns3/applications-module.h"
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/network-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/netanim-module.h"

// Default Network Topology
//
//       10.1.1.0
// n0 -------------- n1
//    point-to-point
//

using namespace ns3;

NS_LOG_COMPONENT_DEFINE("FirstScriptExample");

int
main(int argc, char* argv[])
{
    CommandLine cmd(__FILE__);
    cmd.Parse(argc, argv);

    Time::SetResolution(Time::NS);
    LogComponentEnable("UdpEchoClientApplication", LOG_LEVEL_INFO);
    LogComponentEnable("UdpEchoServerApplication", LOG_LEVEL_INFO);

    NodeContainer nodes;
    nodes.Create(2);

    PointToPointHelper pointToPoint;
    pointToPoint.SetDeviceAttribute("DataRate", StringValue("50Mbps"));
    pointToPoint.SetChannelAttribute("Delay", StringValue("5ms"));

    NetDeviceContainer devices;
    devices = pointToPoint.Install(nodes);

    InternetStackHelper stack;
    stack.Install(nodes);

    Ipv4AddressHelper address;
    address.SetBase("10.1.1.0", "255.255.255.0");

    Ipv4InterfaceContainer interfaces = address.Assign(devices);

    UdpEchoServerHelper echoServer(9);

    ApplicationContainer serverApps = echoServer.Install(nodes.Get(0));
    serverApps.Start(Seconds(1.0));
    serverApps.Stop(Seconds(20.0));

    UdpEchoClientHelper echoClient(interfaces.GetAddress(1), 9);
    echoClient.SetAttribute("MaxPackets", UintegerValue(10));
    echoClient.SetAttribute("Interval", TimeValue(Seconds(1.0)));
    echoClient.SetAttribute("PacketSize", UintegerValue(1024));

    ApplicationContainer clientApps = echoClient.Install(nodes.Get(1));
    clientApps.Start(Seconds(2.0));
    clientApps.Stop(Seconds(20.0));

    Simulator::Run();
    Simulator::Destroy();
    return 0;
}
```

Output:-

```
[0/21 Re-checking globbed directories...
ninja: no work to do.
At time +2s client sent 1024 bytes to 10.1.2.4 port 9
At time +2.0118s server received 1024 bytes from 10.1.1.1 port 49153
At time +2.0118s server sent 1024 bytes to 10.1.1.1 port 49153
received 1024 bytes
At time +2.02161s client received 1024 bytes from 10.1.2.4 port 9
```

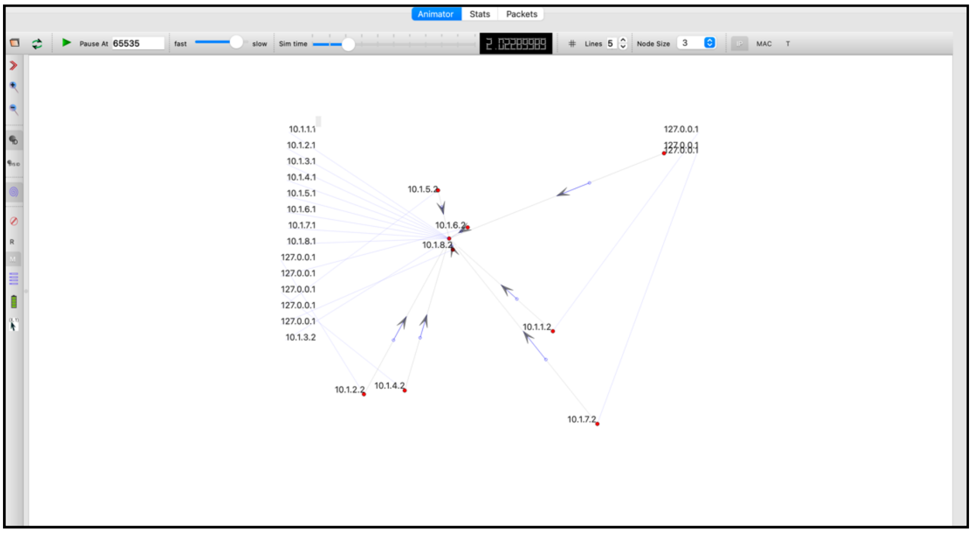
Q2. Star Topology Simulation Using TCP

```
#include "ns3/applications-module.h"
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/netanim-module.h"
#include "ns3/network-module.h"
#include "ns3/point-to-point-layout-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/netanim-module.h"

// Network topology (default)
//
//        n2 n3 n4              .
//         \ | /                .
//          \|/                 .
//     n1--- n0---n5            .
//          /|\                 .
//         / | \                .
//        n8 n7 n6              .
//

using namespace ns3;

NS_LOG_COMPONENT_DEFINE("Star");

int
main(int argc, char* argv[])
{
    //
    // Set up some default values for the simulation.
    //
    Config::SetDefault("ns3::OnOffApplication::PacketSize", UintegerValue(137));

    // ??? try and stick 15kb/s into the data rate
    Config::SetDefault("ns3::OnOffApplication::DataRate", StringValue("14kb/s"));

    //
    // Default number of nodes in the star.  Overridable by command line argument.
    //
    uint32_t nSpokes = 8;

    CommandLine cmd(__FILE__);
    cmd.AddValue("nSpokes", "Number of nodes to place in the star", nSpokes);
    cmd.Parse(argc, argv);

    NS_LOG_INFO("Build star topology.");
    PointToPointHelper pointToPoint;
    pointToPoint.SetDeviceAttribute("DataRate", StringValue("5Mbps"));
    pointToPoint.SetChannelAttribute("Delay", StringValue("2ms"));
    PointToPointStarHelper star(nSpokes, pointToPoint);
    NS_LOG_INFO("Install internet stack on all nodes.");
    InternetStackHelper internet;
    star.InstallStack(internet);

    NS_LOG_INFO("Assign IP Addresses.");
    star.AssignIpv4Addresses(Ipv4AddressHelper("10.1.1.0", "255.255.255.0"));

    NS_LOG_INFO("Create applications.");
    //
    // Create a packet sink on the star "hub" to receive packets.
    //
    uint16_t port = 50000;
    Address hubLocalAddress(InetSocketAddress(Ipv4Address::GetAny(), port));
    PacketSinkHelper packetSinkHelper("ns3::TcpSocketFactory", hubLocalAddress);
    ApplicationContainer hubApp = packetSinkHelper.Install(star.GetHub());
    hubApp.Start(Seconds(1.0));
    hubApp.Stop(Seconds(10.0));

    //
    // Create OnOff applications to send TCP to the hub, one on each spoke node.
    //
    OnOffHelper onOffHelper("ns3::TcpSocketFactory", Address());
    onOffHelper.SetAttribute("OnTime",
StringValue("ns3::ConstantRandomVariable[Constant=1]"));
    onOffHelper.SetAttribute("OffTime",
StringValue("ns3::ConstantRandomVariable[Constant=0]"));

    ApplicationContainer spokeApps;

    for (uint32_t i = 0; i < star.SpokeCount(); ++i)
    {
        AddressValue remoteAddress(InetSocketAddress(star.GetHubIpv4Address(i),
port));
        onOffHelper.SetAttribute("Remote", remoteAddress);
        spokeApps.Add(onOffHelper.Install(star.GetSpokeNode(i)));
    }
    spokeApps.Start(Seconds(1.0));
    spokeApps.Stop(Seconds(10.0));

    NS_LOG_INFO("Enable static global routing.");
    //
    // Turn on global static routing so we can actually be routed across the star.
    //
    Ipv4GlobalRoutingHelper::PopulateRoutingTables();

    NS_LOG_INFO("Enable pcap tracing.");
    //
    // Do pcap tracing on all point-to-point devices on all nodes.
    //
    pointToPoint.EnablePcapAll("star");
    AnimationInterface anim("star.xml");

    NS_LOG_INFO("Run Simulation.");
    Simulator::Run();
    Simulator::Destroy();
    NS_LOG_INFO("Done.");

    return 0;
}
```

Output:-

```
10/21 Re-checking globbed directories...
At time +2s client sent 1024 bytes to 10.1.1.2 port 9
At time +3s client sent 1024 bytes to 10.1.1.2 port 9
At time +4s client sent 1024 bytes to 10.1.1.2 port 9
At time +5s client sent 1024 bytes to 10.1.1.2 port 9
At time +6s client sent 1024 bytes to 10.1.1.2 port 9
At time +7s client sent 1024 bytes to 10.1.1.2 port 9
At time +8s client sent 1024 bytes to 10.1.1.2 port 9
At time +9s client sent 1024 bytes to 10.1.1.2 port 9
At time +10s client sent 1024 bytes to 10.1.1.2 port 9
At time +11s client sent 1024 bytes to 10.1.1.2 port 9
```

Simulation:-



Q3. Ping messages/trace routes over a network of 6 nodes and find the number packets dropped due to congestion.

```
#include "ns3/core-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-apps-module.h"
#include "ns3/internet-module.h"

#include <fstream>

using namespace ns3;

NS_LOG_COMPONENT_DEFINE("Ping6Example");

int
main(int argc, char** argv)
{
    bool verbose = false;
    bool allNodes = false;

    CommandLine cmd(__FILE__);
    cmd.AddValue("verbose", "turn on log components", verbose);
    cmd.AddValue("allNodes", "Ping all the nodes (true) or just one neighbor
(false)", allNodes);
    cmd.Parse(argc, argv);

    if (verbose)
    {
        LogComponentEnable("Ping6Example", LOG_LEVEL_INFO);
        LogComponentEnable("Ipv6EndPointDemux", LOG_LEVEL_ALL);
        LogComponentEnable("Ipv6L3Protocol", LOG_LEVEL_ALL);
        LogComponentEnable("Ipv6StaticRouting", LOG_LEVEL_ALL);
        LogComponentEnable("Ipv6ListRouting", LOG_LEVEL_ALL);
        LogComponentEnable("Ipv6Interface", LOG_LEVEL_ALL);
        LogComponentEnable("Icmpv6L4Protocol", LOG_LEVEL_ALL);
        LogComponentEnable("Ping", LOG_LEVEL_ALL);
        LogComponentEnable("NdiscCache", LOG_LEVEL_ALL);
    }

    NS_LOG_INFO("Create nodes.");
    NodeContainer n;
    n.Create(4);

    /* Install IPv4/IPv6 stack */
    InternetStackHelper internetv6;
    internetv6.SetIpv4StackInstall(false);
    internetv6.Install(n);
    NS_LOG_INFO("Create channels.");
    CsmaHelper csma;
    csma.SetChannelAttribute("DataRate", DataRateValue(5000));
    csma.SetChannelAttribute("Delay", TimeValue(MilliSeconds(2)));
    NetDeviceContainer d = csma.Install(n);

    Ipv6AddressHelper ipv6;
    NS_LOG_INFO("Assign IPv6 Addresses.");
    Ipv6InterfaceContainer i = ipv6.Assign(d);

    NS_LOG_INFO("Create Applications.");

    // Create a Ping application to send ICMPv6 echo request from node zero
    uint32_t packetSize = 1024;
    uint32_t maxPacketCount = 7;

    Ipv6Address destination = allNodes ? Ipv6Address::GetAllNodesMulticast() :
i.GetAddress(1, 0);
    PingHelper ping(destination);

    ping.SetAttribute("Count", UintegerValue(maxPacketCount));
    ping.SetAttribute("Size", UintegerValue(packetSize));
    ping.SetAttribute("InterfaceAddress", AddressValue(i.GetAddress(0, 0)));

    ApplicationContainer apps = ping.Install(n.Get(0));
    apps.Start(Seconds(2.0));
    apps.Stop(Seconds(10.0));

    AsciiTraceHelper ascii;
    csma.EnableAsciiAll(ascii.CreateFileStream("ping6.tr"));
    csma.EnablePcapAll(std::string("ping6"), true);

    NS_LOG_INFO("Run Simulation.");
    Simulator::Run();
    Simulator::Destroy();
    NS_LOG_INFO("Done.");

    return 0;
}
```

Output:-

```
10/21 Re-checking globbed directories...
PING fe80::200:ff: fe00:2 - 1024 bytes of data; 1072 bytes including IMP and IPv6 headers.
1032 bytes from (fe80::200:ff:fe00:2): imp_seq=0 ttl=64 time=4219.74 ms - fe80::200:ff:fe00:2 ping
statistics - 7 packets transmitted, 1 received, 85% packet loss, time 8000ms
rtt min/avg/max/mdev = 4219/4219/4219/0 ms
```