Problem 1

'traceroute' is a computer network diagnostic tool for displaying the route (path) and measuring transit delays of packets across an Internet Protocol (IP) network. In this problem, you will use the 'traceroute' command to understand how packets route to a destination.

1. Run *traceroute* command to find a route to 'ucla.edu'. How many hops are there in between your local host to the destination? Copy and paste the result on your console in the answer box. (If you are using Windows Command Prompt, then use '*tracert*' command instead.)

2. Run *traceroute* command to find a route to 'columbia.edu'. Copy and paste the result into the answer box.

3. Compare two results in terms of the number of hops and the delays.

1. traceroute ucla.edu
traceroute to ucla.edu (128.97.27.37), 64 hops max, 52 byte packets
1 wifi-131-179-34-2 (131.179.34.2) 19.077 ms 23.616 ms 28.306 ms
2 169.232.8.153 (169.232.8.153) 10.408 ms 3.762 ms 6.395 ms
3 cr00f2.csb1–sr02f2.csb1.ucla.net (169.232.8.7) 4.406 ms 6.125 ms 2.443 ms
4 128.97.27.37 (128.97.27.37) 3.877 ms !Z 3.150 ms !Z 2.922 ms !Z

2. traceroute columbia.edu
traceroute to columbia.edu (128.59.105.24), 64 hops max, 52 byte packets
1 192.168.0.1 (192.168.0.1) 1.918 ms 1.269 ms 3.888 ms
2 142.254.236.89 (142.254.236.89) 11.771 ms 14.149 ms 10.154 ms
3 agg57.snmncaby02h.socal.rr.com (76.167.30.5) 12.787 ms 12.506 ms 12.904 ms
4 agg20.lamrcadq02r.socal.rr.com (72.129.10.130) 17.546 ms 18.491 ms 20.633 ms
5 agg28.tustcaft01r.socal.rr.com (72.129.9.2) 15.655 ms 16.379 ms 15.727 ms
6 bu-ether26.tustca4200w-bcr00.tbone.rr.com (66.109.3.232) 20.543 ms 20.326 ms 15.965 ms
7 bu-ether14.lsancarc0yw-bcr00.tbone.rr.com (66.109.6.4) 18.607 ms be4.clmkohpe01r.midwest.rr.com (107.14.19.37) 18.848 ms 21.107 ms
8 66.109.5.123 (66.109.5.123) 15.894 ms 14.603 ms 15.061 ms
9 be5341.ccr41.lax04.atlas.cogentco.com (38.142.237.33) 14.400 ms 14.633 ms 16.071 ms
10 be3360.ccr42.lax01.atlas.cogentco.com (154.54.25.149) 18.316 ms
be3271.ccr41.lax01.atlas.cogentco.com (154.54.42.101) 16.436 ms 16.115 ms
11 be2932.ccr32.phx01.atlas.cogentco.com (154.54.45.161) 29.490 ms 27.842 ms
be2931.ccr31.phx01.atlas.cogentco.com (154.54.44.85) 28.326 ms
12 be2930.ccr21.elp01.atlas.cogentco.com (154.54.42.78) 68.749 ms
be2929.ccr21.elp01.atlas.cogentco.com (154.54.42.66) 38.497 ms 36.803 ms
13 be2928.ccr42.iah01.atlas.cogentco.com (154.54.30.161) 51.490 ms 52.323 ms 51.827 ms
14 be2690.ccr42.atl01.atlas.cogentco.com (154.54.28.129) 66.612 ms 64.834 ms 66.018 ms
15 be2113.ccr42.dca01.atlas.cogentco.com (154.54.24.221) 76.490 ms 76.683 ms 75.626 ms
16 be2807.ccr42.jfk02.atlas.cogentco.com (154.54.40.109) 82.181 ms 83.560 ms 81.781 ms
17 be2897.rcr24.jfk01.atlas.cogentco.com (154.54.84.214) 85.117 ms 83.047 ms 83.262 ms
18 38.122.8.210 (38.122.8.210) 84.652 ms 83.747 ms 83.077 ms
19 cc-core-1-x-nyser32-gw-1.net.columbia.edu (128.59.255.5) 91.146 ms 97.885 ms 84.604 ms
20 cc-conc-1-x-cc-core-1.net.columbia.edu (128.59.255.210) 82.474 ms 83.668 ms 84.287 ms
21 childpolicyintl.org (128.59.105.24) 82.140 ms 87.638 ms 84.382 ms

3. For ucla.edu, since it's close to us, it has 4 hops and delay of 2ms, and columbia.edu is far from us, and it has 21 hops and delay of 87 ms.

Problem 2

Host A is sending real-time voice over a packet-switched network. Host A converts analog voice to a digital 128 kbps bit stream on the fly. Host A then groups 1,600 bytes into a packet. Assume that the 1,600 bytes packet already includes all headers. There is one link between Hosts A and B; its transmission rate is 3 Mbps and its propagation delay is 20 msec. As soon as Host A gathers a packet, it sends it to Host B. As soon as Host B receives an entire packet, it converts the packets bits to an analog signal. How much time elapses from the time a bit is created (from the original analog signal at Host A) until the bit is decoded (as part of the analog signal at Host B)? In this problem, do not consider acknowledgement (response) from Host B.

Host A converts analog to digital bit stream 128 kbps
Host A grouping bits 1600-byte packets
Transmission rate 3mbps
Propagation delay 20 msec
Queuing delay: 1600 * 8/ 128 = 100 ms
Transmission delay: 1600 * 8 / 3000000 s = 4.267 ms
Time elapsed = Queuing delay + transmission delay + propagation delay = 100 + 4.267 + 20 = 124.267 ms

Problem 3

Tow hosts, A and B are separated by 20,000 kilometers and are connected by a direct link of $R = 2Mbps$. Suppose the propagation speed over the link is $2.5 * 10^8 meters/sec$.

1. Consider sending a file of 800,000 bits from Host A to Host B. Suppose the file is sent continuously as one large message. What is the maximum number of bits that will be in the link at any given time?

2. How long does it take to send the file, assuming it is sent continuously?

3. Suppose now the file is broken up into 20 packets with each packet containing 40,000 bits. Suppose that each packet is acknowledged by the receiver and the transmission time of an acknowledgment packet is negligible. Finally, assume that the sender cannot send a packet until the preceding one is acknowledged. How long does it take to send the file?

1. Distance between A and B: 20 000 000 m
Transmission Rate R = 2Mbps = 2000 000 bps
Propagation Delay = distance / propagation speed = 20 000 000/250 000 000 = 0.08 sec
Max of bits = Propagation Delay * R = 0.08 * 2000 000 = 16 0000 bits

2. Transmission Delay = 800 000 / 2000 000 = 0.4 sec
total time = Propagation Delay + Transmission Delay = 0.08 + 0.004 = 0.48 sec

3. Since the sender cannot send a packet until the preceding one is acknowledged,
The total time is 20 times the time for one packet
new Transmission Delay = 40 000 / 2000 000 = 0.02 sec
total time = 20 * (Propagation Delay + new Transmission Delay)
total time = 20 * (0.08 + 0.02) = 2 sec

Problem 4

Alice and Bob are working remotely on a course project and are using `git` as the version control software.

1. Is it true that one must have GitHub/GitLab account to use git?
2. What is(are) the command(s) to initialize a local git repository?
3. Do Alice and Bob both must initialize local git repository? If no, what are the alternative?
4. Let's consider that Alice modified the file `server.cpp`:
   (a) What git commands Alice needs to save modifications in the local git repository
   (b) What git commands Alice needs to upload saved modifications to GitHub
   (c) What git commands Bob needs to get Alice's changes and apply them to the local repository
5. Let's consider that both Alice and Bob modified the file `server.cpp` and Alice was first to successfully upload saved modifications (commit) to GitHub
   (a) Can Bob upload his changes without any additional actions? If no, why?
   (b) If actions needed, list git commands that Bob will need to use to share his modifications with Alice.

1. No. We just use 'git' command in terminal.

2. git init
git clone /path/to/repository

3. No. They can work remotely:
git clone username@host:/path/to/repository

4. (a) git add 'filename'
git commit -m 'Commit message'
(b) git push origin master
If we have already cloned a new repository
(c) git fetch
git merge

5. (a) No. He has to resolve the changes manually. Since the part he modified might be deleted by Alice in her latest change. Then use 'git add' to commit the changes.
(b) git diff 'source branch' 'target branch'
Use the 'diff' command to create a file showing the differences so Alice can know the modifications.

Problem 5

You will learn some basic usages of `Vagrant` in your projects.

1. What is Vagrant mainly used for?
2. What is VirtualBox used for?
3. What is *Vagrantfile*?
4. List at least five commands you can use with Vagrant.

1. Vagrant is a tool for building and managing virtual machine environments in a single workflow.

2. It sets up a virtual environment, that allows you to emulate an operating system on your own PC and use it like it's running on real hardware.

3. Vagrantfile is a Ruby file used to configure Vagrant on a per-project basis. The main function of the Vagrantfile is to described the virtual machines required for a project as well as how to configure and provision these machines.

4.vagrant init
vagrant up
vagrant ssh
vagrant halt
vagrant destroy