Task1

1. What is the output of “nodes” and “net”

```
*** Starting CLI:
mininet> nodes
available nodes are:
h1 h2 h3 h4 h5 h6 h7 h8 s1 s2 s3 s4 s5 s6 s7
```

Fig1 nodes

```
mininet> net
h1 h1-eth0:s3-eth2
h2 h2-eth0:s3-eth3
h3 h3-eth0:s4-eth2
h4 h4-eth0:s4-eth3
h5 h5-eth0:s6-eth2
h6 h6-eth0:s6-eth3
h7 h7-eth0:s7-eth2
h8 h8-eth0:s7-eth3
s1 lo:  s1-eth1:s2-eth1 s1-eth2:s5-eth1
s2 lo:  s2-eth1:s1-eth1 s2-eth2:s3-eth1 s2-eth3:s4-eth1
s3 lo:  s3-eth1:s2-eth2 s3-eth2:h1-eth0 s3-eth3:h2-eth0
s4 lo:  s4-eth1:s2-eth3 s4-eth2:h3-eth0 s4-eth3:h4-eth0
s5 lo:  s5-eth1:s1-eth2 s5-eth2:s6-eth1 s5-eth3:s7-eth1
s6 lo:  s6-eth1:s5-eth2 s6-eth2:h5-eth0 s6-eth3:h6-eth0
s7 lo:  s7-eth1:s5-eth3 s7-eth2:h7-eth0 s7-eth3:h8-eth0
mininet>
```

Fig2 net

2. What is the output of “h7 ifconfig”

```
mininet> h7 ifconfig
h7-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.7  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::d430:ddff:fe5e:e8bd  prefixlen 64  scopeid 0x20<link>
        ether d6:30:dd:5e:e8:bd  txqueuelen 1000  (Ethernet)
        RX packets 250  bytes 34898 (34.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10  bytes 796 (796.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

mininet>
```

Fig3 h7 ifconfig

Task2

1. Draw the function call graph of this controller. For example, once a packet comes to the controller, which function is the first to be called, which one is the second, and so forth?

2. Have h1 ping h2, and h1 ping h8 for 100 times (e.g., h1 ping -c100 p2).

```
mininet> h1 ping -c100 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.522 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.173 ms
```

Fig4 h1 ping -c100 h2

```
mininet> h1 ping -c100 h8
PING 10.0.0.8 (10.0.0.8) 56(84) bytes of data.
64 bytes from 10.0.0.8: icmp_seq=1 ttl=64 time=1.50 ms
64 bytes from 10.0.0.8: icmp_seq=2 ttl=64 time=0.239 ms
```

Fig5 h1 ping -c100 h8

a. How long does it take (on average) to ping for each case?
   h1 ping -c100 h2: 0.168ms
   h1 ping -c100 h8: 0.232ms
b. What is the minimum and maximum ping you have observed?
   h1 ping -c100 h2: min = 0.071ms, max = 0.522ms
   h1 ping -c100 h8: min = 0.064ms, max = 1.496ms
c. What is the difference, and why?
   h1 ping h8 is longer than h2, in that packets between h1 and h2 only need to go through s3, but h1 between h8 have to go through s3, s2, s1, s5, s7.

3. Run “iperf h1 h2” and “iperf h1 h8”

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['110 Gbits/sec', '110 Gbits/sec']
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['87.7 Gbits/sec', '87.8 Gbits/sec']
```

Fig6 iperf

a. What is “iperf” used for?
   Test TCP bandwidth between hosts.
b. What is the throughput for each case?

h1-h2: 110 Gbits/sec, h2-h1: 110 Gbits/sec
h1-h8: 87.7 Gbits/sec, h8-h1: 87.8 Gbits/sec

c. What is the difference, and explain the reasons for the difference.
Throughput between h1 and h2 is higher than h1 and h8, because the switches that the packets need to go through between h1, h2 are less than h1, h8.

4. Which of the switches observe traffic? Please describe your way for observing such traffic on switches (e.g., adding some functions in the “of_tutorial” controller).
All the switches can observe traffic.

Task3

1. Describe how the above code works, such as how the "MAC to Port" map is established. You could use a ‘ping’ example to describe the establishment process (e.g., h1 ping h2).
“MAC to Port” map is empty at first, when the packets go through, the program will check whether the source port and mac are stored in the map, then decide to add the key-value pairs to the map. After that, the program will find out whether the destination mac has stored port in the map. If exists, send the packets to the port; if not, send the packets to each port.

2. (Comment out all prints before doing this experiment) Have h1 ping h2, and h1 ping h8 for 100 times (e.g., h1 ping -c100 p2).
a. How long did it take (on average) to ping for each case?
h1 ping h2: 1.114ms
h1 ping h8: 4.526ms
b. What is the minimum and maximum ping you have observed?
h1 ping h2: min = 1.021ms, max = 2.003ms
h1 ping h8: min = 3.327ms, max = 6.097ms
c. Any difference from Task 2 and why do you think there is a change if there is?
Time is longer than Task2, maybe in that it takes time to check mac-to-port map.

3. Run “iperf h1 h2” and “iperf h1 h8”.
a. What is the throughput for each case?
h1-h2: 117 Gbits/sec, h2-h1: 118 Gbits/sec
h1-h8: 89.7 Gbits/sec, h8-h1: 89.8 Gbits/sec

b. What is the difference from Task 2 and why do you think there is a change if there is?
The throughput is higher than Task2, maybe because the destination ports are already known.