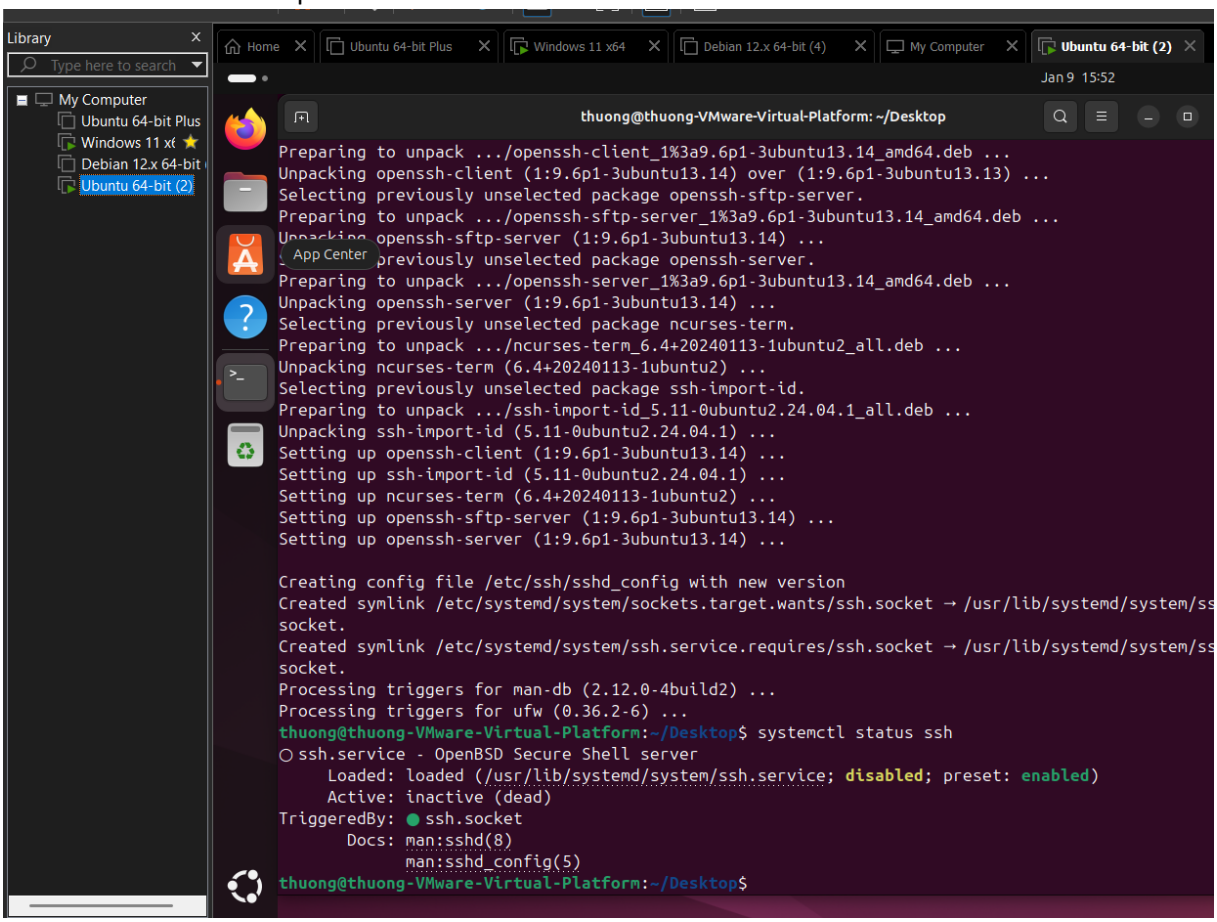
Week 6 – Networking

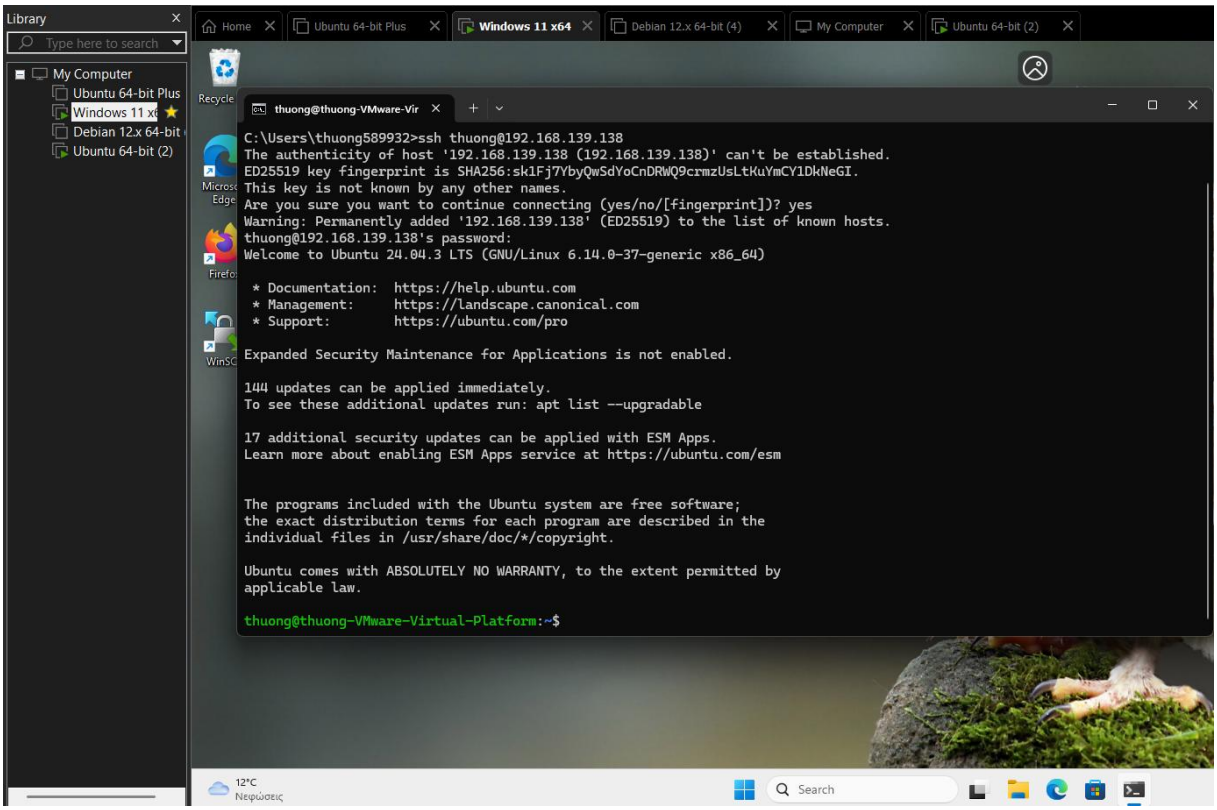
Student number: 589932

Assignment 6.1: Working from home

Screenshot installation openssh-server:

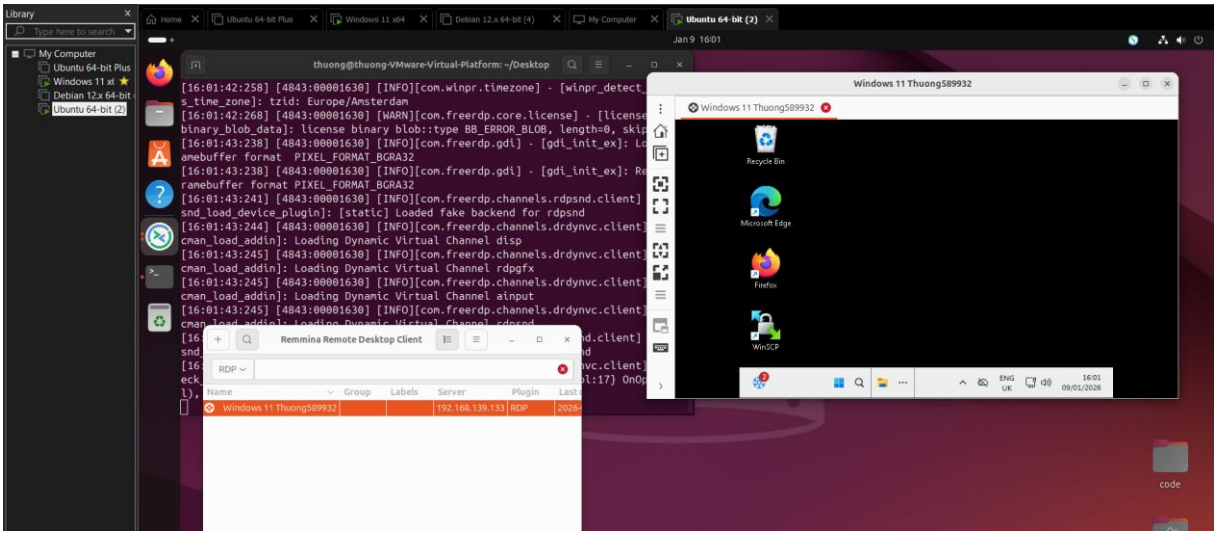


Screenshot successful SSH command execution:



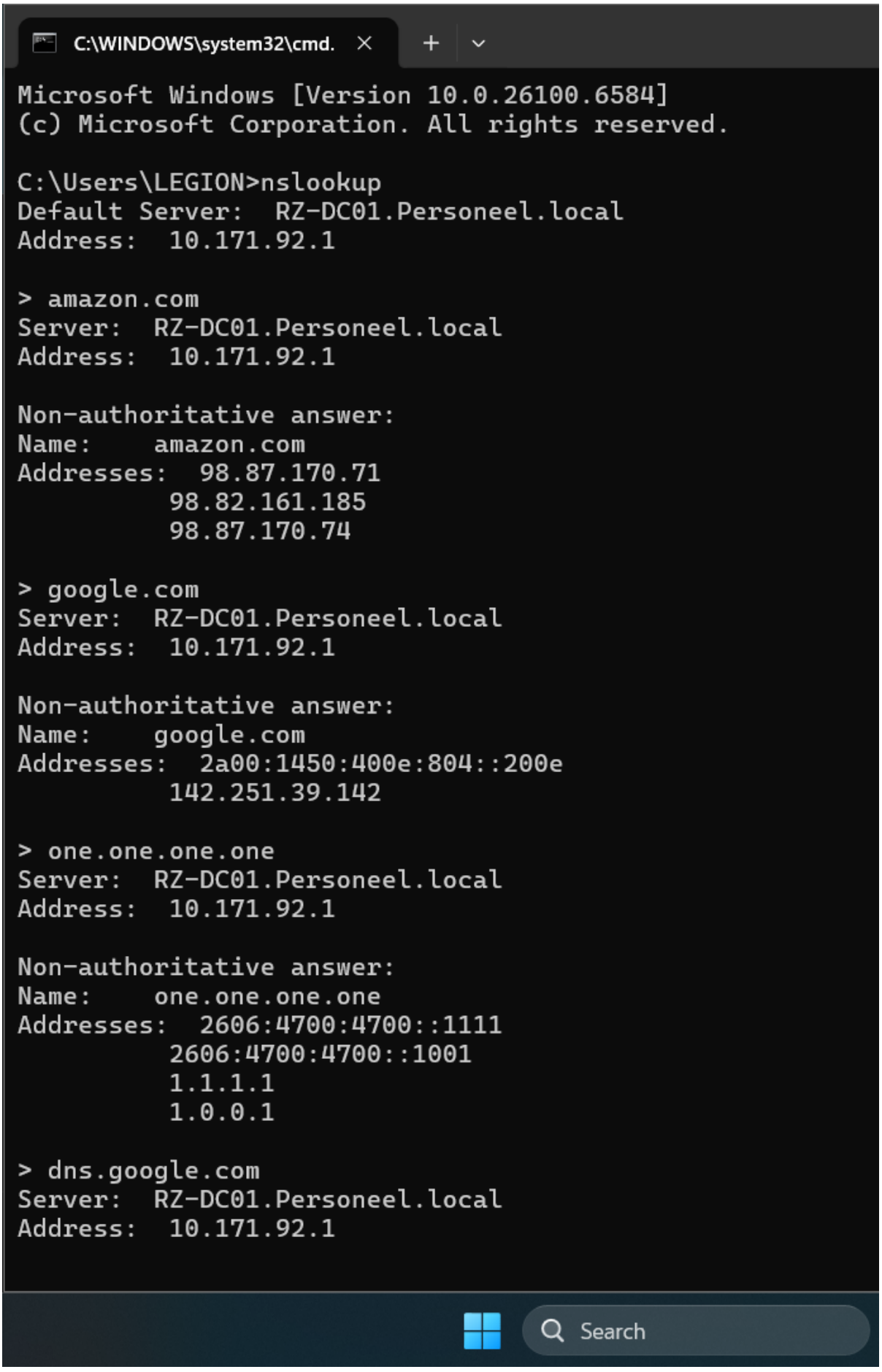
Screenshot successful execution SCP command:

Screenshot remmina:

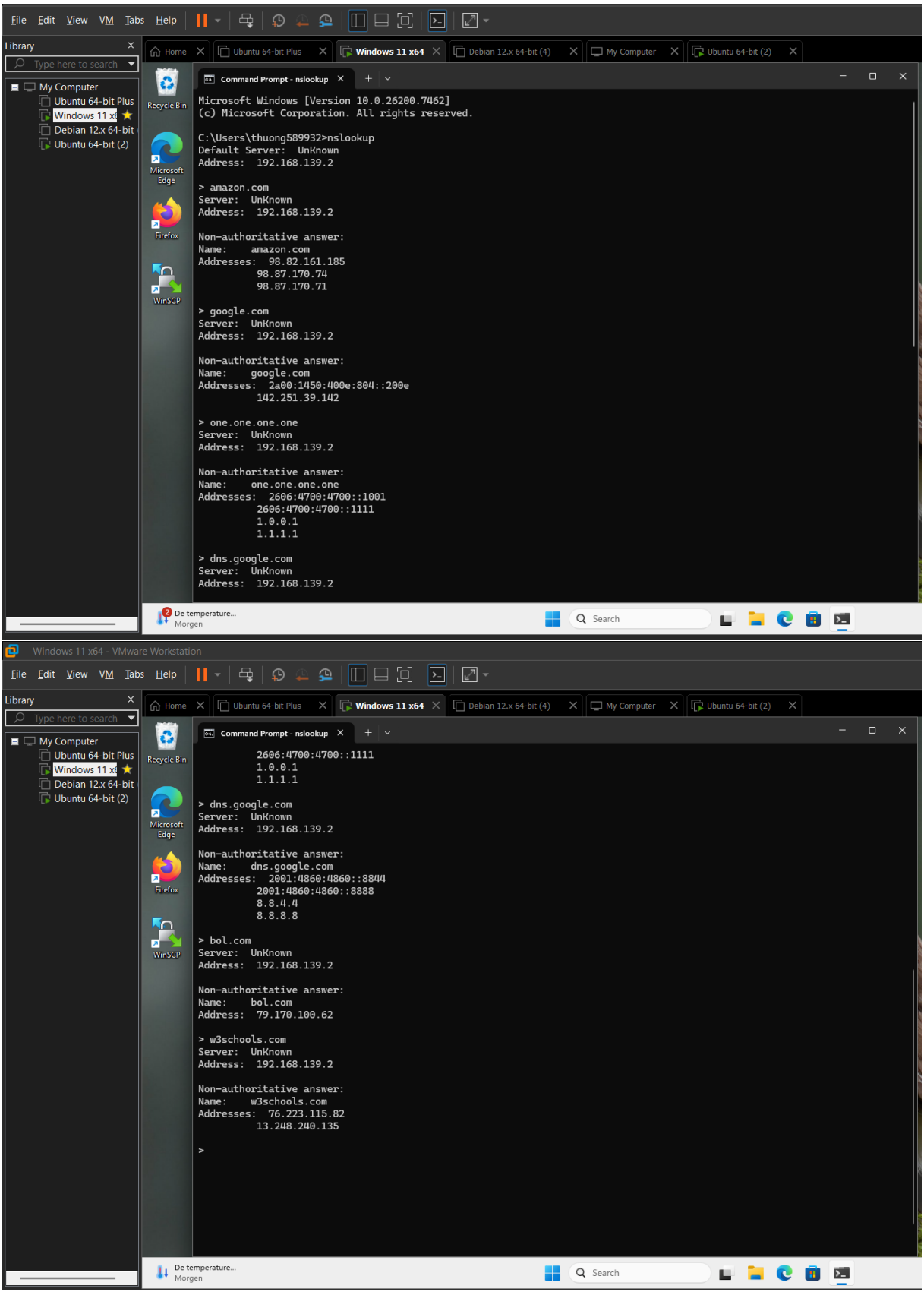


Assignment 6.2: IP addresses websites

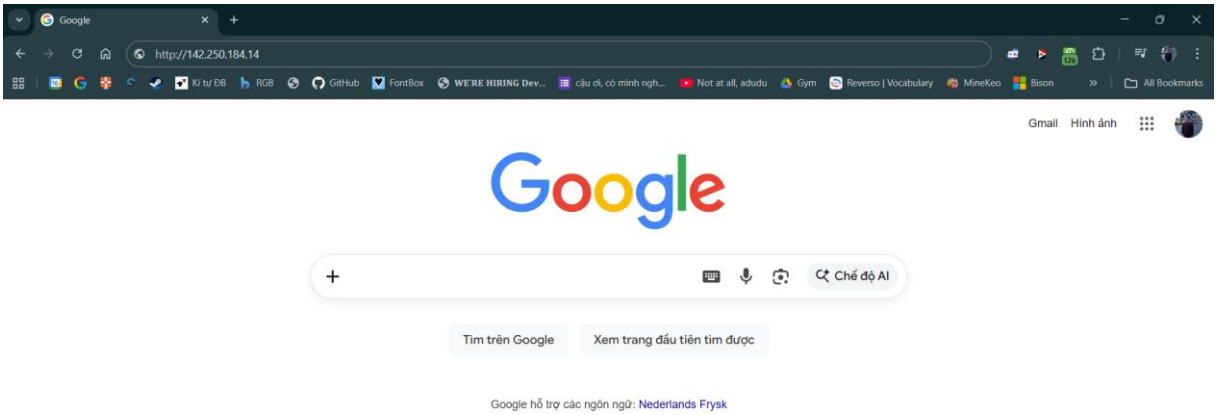
Relevant screenshots nslookup command:

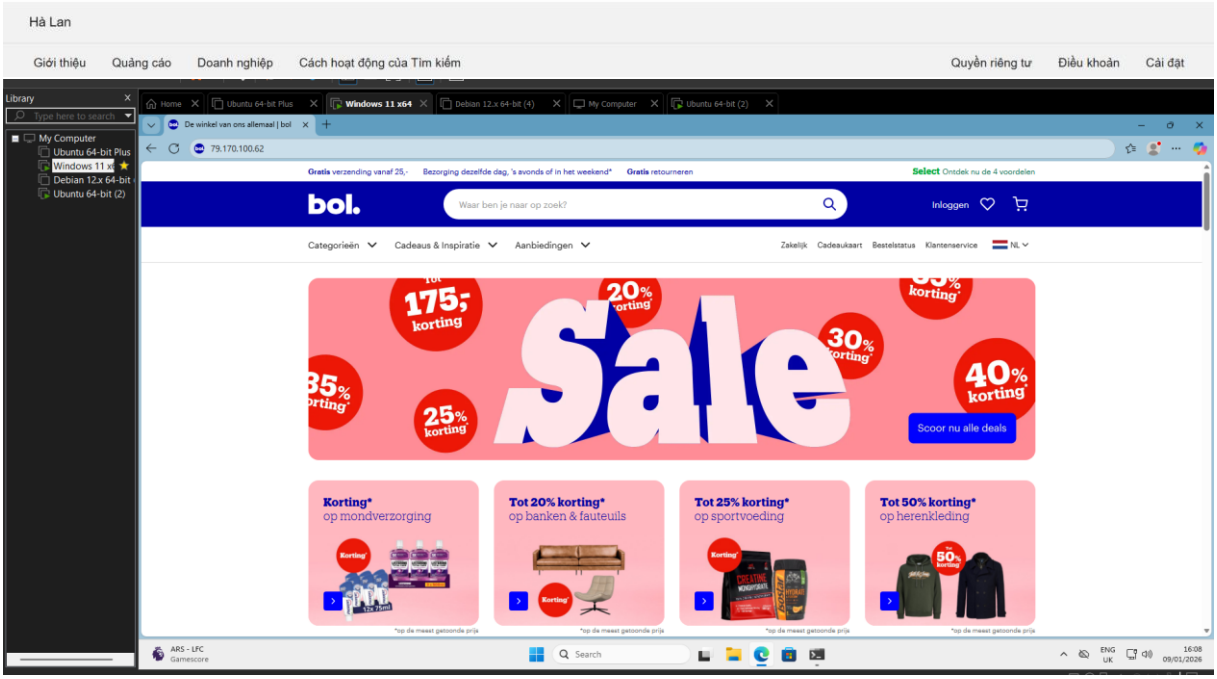

C:\WINDOWS\system32\cmd.
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.
C:\Users\LEGION>nslookup
Default Server: RZ-DC01.Personeel.local
Address: 10.171.92.1
> amazon.com
Server: RZ-DC01.Personeel.local
Address: 10.171.92.1
Non-authoritative answer:
Name: amazon.com
Addresses: 98.87.170.71
98.82.161.185
98.87.170.74
> google.com
Server: RZ-DC01.Personeel.local
Address: 10.171.92.1
Non-authoritative answer:
Name: google.com
Addresses: 2a00:1450:400e:804::200e
142.251.39.142
> one.one.one.one
Server: RZ-DC01.Personeel.local
Address: 10.171.92.1
Non-authoritative answer:
Name: one.one.one.one
Addresses: 2606:4700:4700::1111
2606:4700:4700::1001
1.1.1.1
1.0.0.1
> dns.google.com
Server: RZ-DC01.Personeel.local
Address: 10.171.92.1
Search


C:\WINDOWS\system32\cmd.
> dns.google.com
Server: RZ-DC01.Personeel.local
Address: 10.171.92.1
Non-authoritative answer:
Name: dns.google.com
Addresses: 2001:4860:4860::8888
2001:4860:4860::8844
8.8.8.8
8.8.4.4
> bol.com
Server: RZ-DC01.Personeel.local
Address: 10.171.92.1
Non-authoritative answer:
Name: bol.com
Address: 79.170.100.62
> w3schools.com
Server: RZ-DC01.Personeel.local
Address: 10.171.92.1
Non-authoritative answer:
Name: w3schools.com
Addresses: 76.223.115.82
13.248.240.135
>
Search


File Edit View VM Tabs Help
Library
Type here to search
My Computer
Ubuntu 64-bit Plus
Windows 11 x64
Debian 12.x 64-bit
Ubuntu 64-bit (2)
Home
Ubuntu 64-bit Plus
Windows 11 x64
Debian 12.x 64-bit (4)
My Computer
Ubuntu 64-bit (2)
Recycle Bin
Microsoft Edge
Firefox
WinSCP
Command Prompt - nslookup
Microsoft Windows [Version 10.0.26200.7462]
(c) Microsoft Corporation. All rights reserved.
C:\Users\thuong589932>nslookup
Default Server: UnKnown
Address: 192.168.139.2
> amazon.com
Server: UnKnown
Address: 192.168.139.2
Non-authoritative answer:
Name: amazon.com
Addresses: 98.82.161.185
98.87.170.74
98.87.170.71
> google.com
Server: UnKnown
Address: 192.168.139.2
Non-authoritative answer:
Name: google.com
Addresses: 2a00:1450:400e:804::200e
142.251.39.142
> one.one.one.one
Server: UnKnown
Address: 192.168.139.2
Non-authoritative answer:
Name: one.one.one.one
Addresses: 2606:4700:4700::1001
2606:4700:4700::1111
1.0.0.1
1.1.1.1
> dns.google.com
Server: UnKnown
Address: 192.168.139.2
De temperature...
Morgen
Search
Windows 11 x64 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to search
My Computer
Ubuntu 64-bit Plus
Windows 11 x64
Debian 12.x 64-bit
Ubuntu 64-bit (2)
Home
Ubuntu 64-bit Plus
Windows 11 x64
Debian 12.x 64-bit (4)
My Computer
Ubuntu 64-bit (2)
Recycle Bin
Microsoft Edge
Firefox
WinSCP
Command Prompt - nslookup
2606:4700:4700::1111
1.0.0.1
1.1.1.1
> dns.google.com
Server: UnKnown
Address: 192.168.139.2
Non-authoritative answer:
Name: dns.google.com
Addresses: 2001:4860:4860::8844
2001:4860:4860::8888
8.8.4.4
8.8.8.8
> bol.com
Server: UnKnown
Address: 192.168.139.2
Non-authoritative answer:
Name: bol.com
Address: 79.170.100.62
> w3schools.com
Server: UnKnown
Address: 192.168.139.2
Non-authoritative answer:
Name: w3schools.com
Addresses: 76.223.115.82
13.248.240.135
>
De temperature...
Morgen
Search

Screenshot website visit via IP address:





Assignment 6.3: subnetting

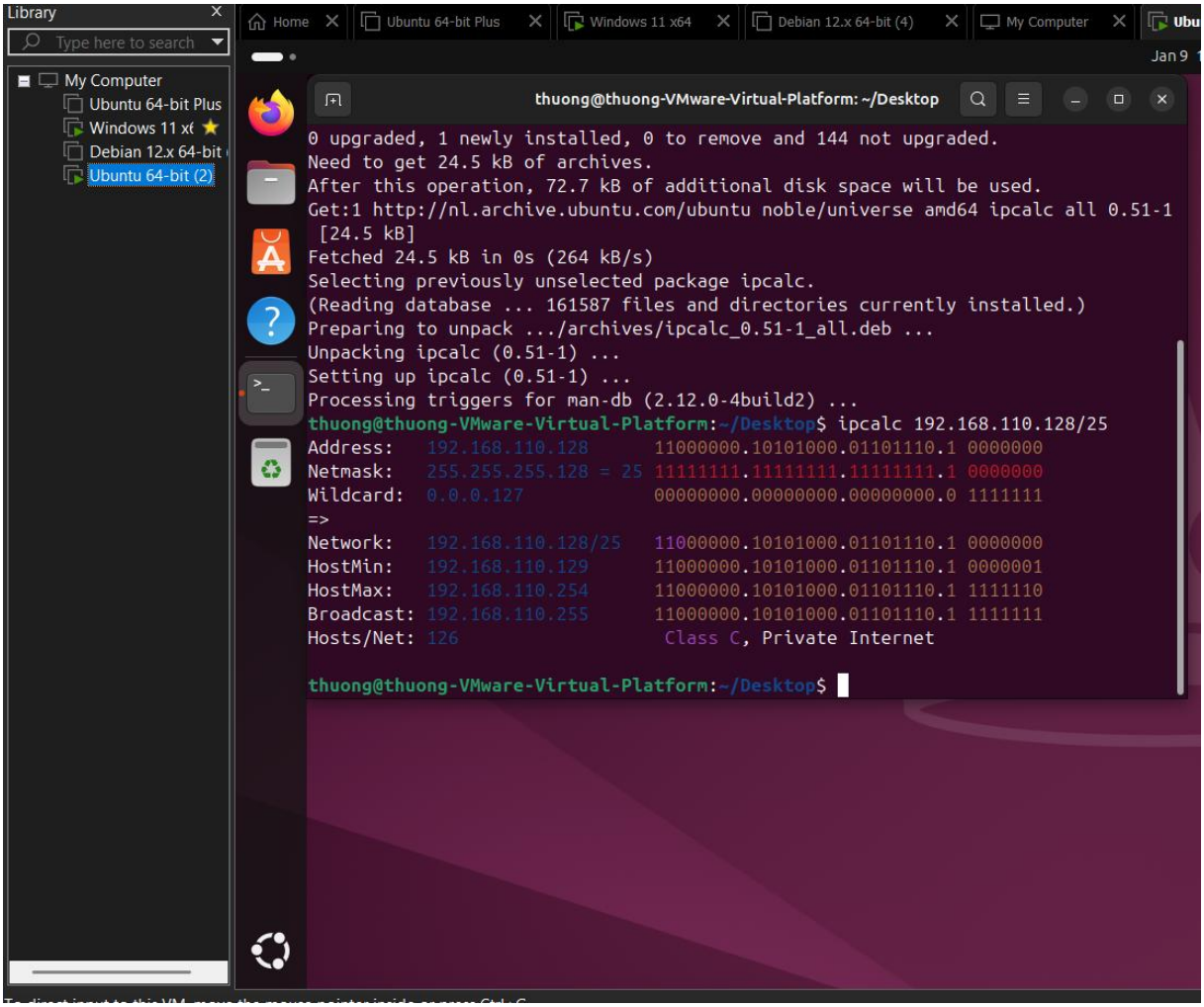
How many IP addresses are in this network configuration 192.168.110.128/25?

There are **128 IP addresses**

What is the usable IP range to hand out to the connected computers?

From **192.168.110.129 to 192.168.110.254**

Check your two previous answers with this Linux command: `ipcalc 192.168.110.128/25`



Explain the above calculation in your own words.

A /25 subnet means that 25 bits are used for the network part of the IP address, leaving 7 bits for host addresses.

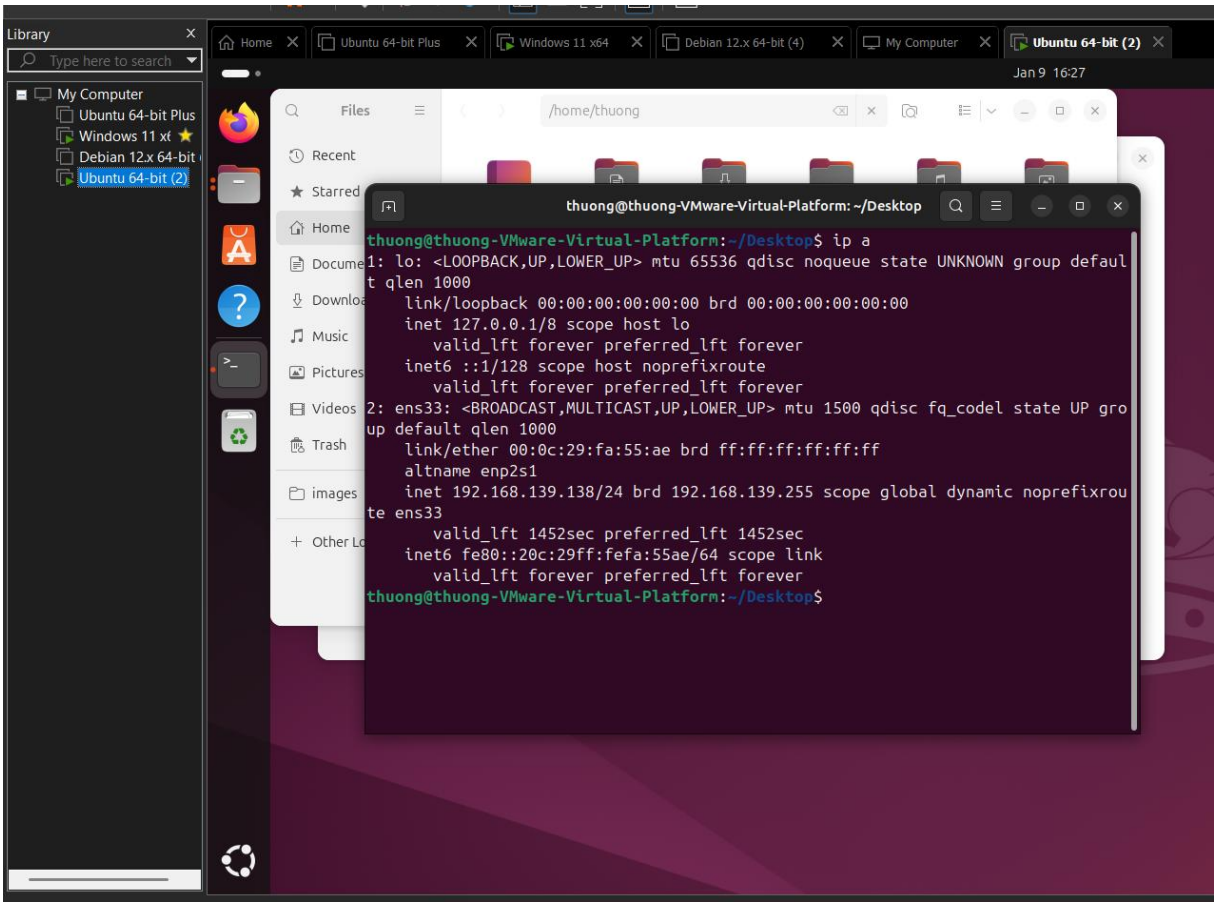
With 7 host bits, the total number of IP addresses is 2^7, which equals 128.

The first address of the subnet is reserved as the network address, and the last address is reserved as the

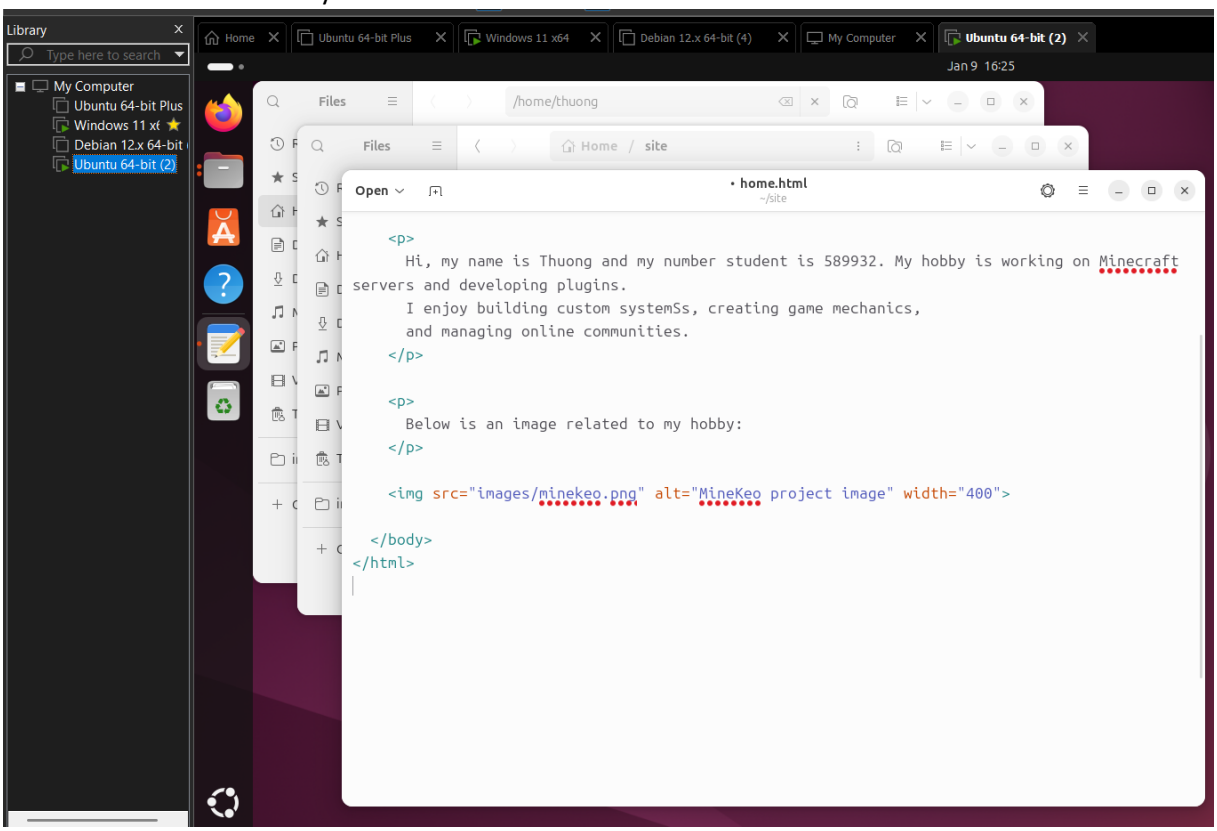
broadcast address.
Because these two addresses cannot be assigned to computers, only 126 IP addresses are usable.
Therefore, the usable IP range for connected computers is from 192.168.110.129 to 192.168.110.254.

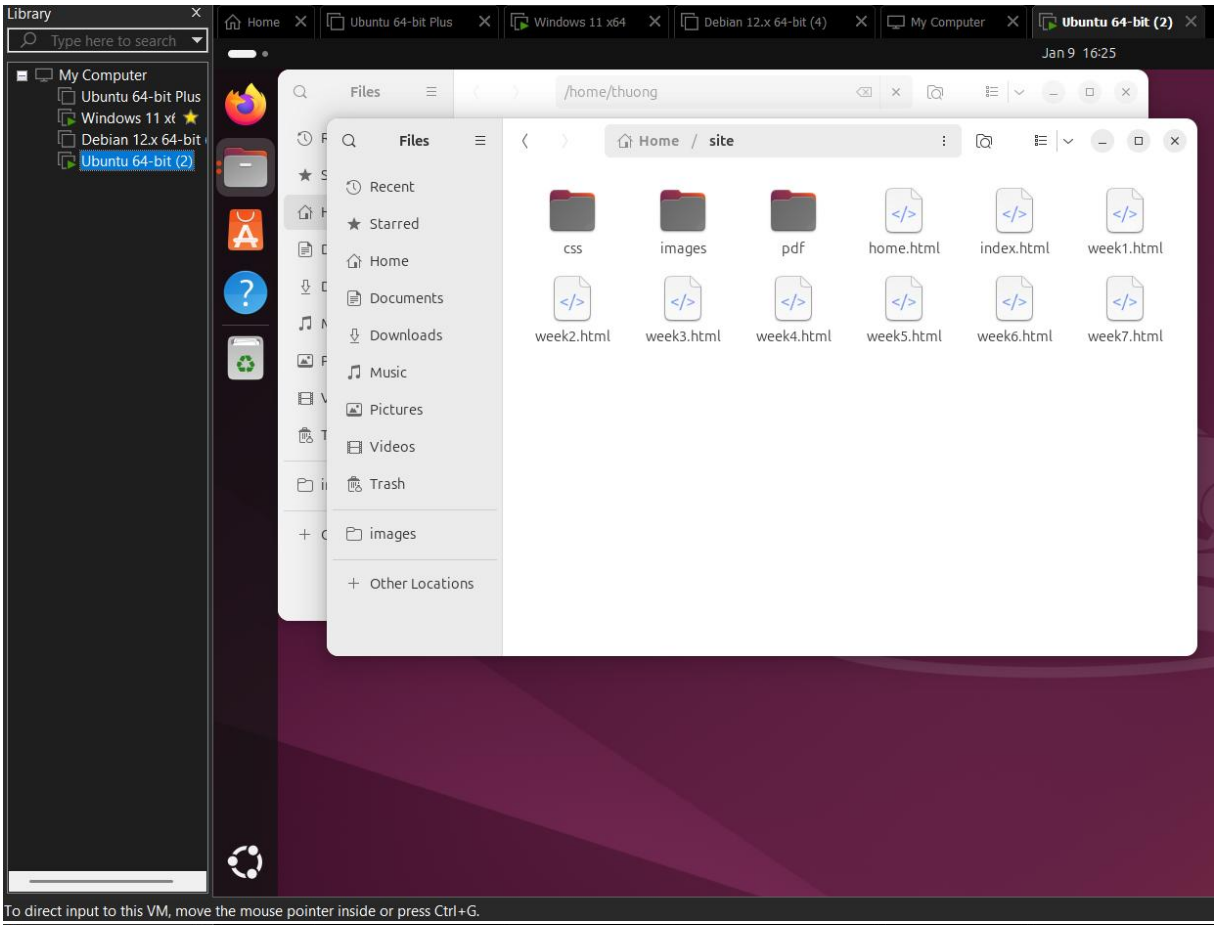
Assignment 6.4: HTML

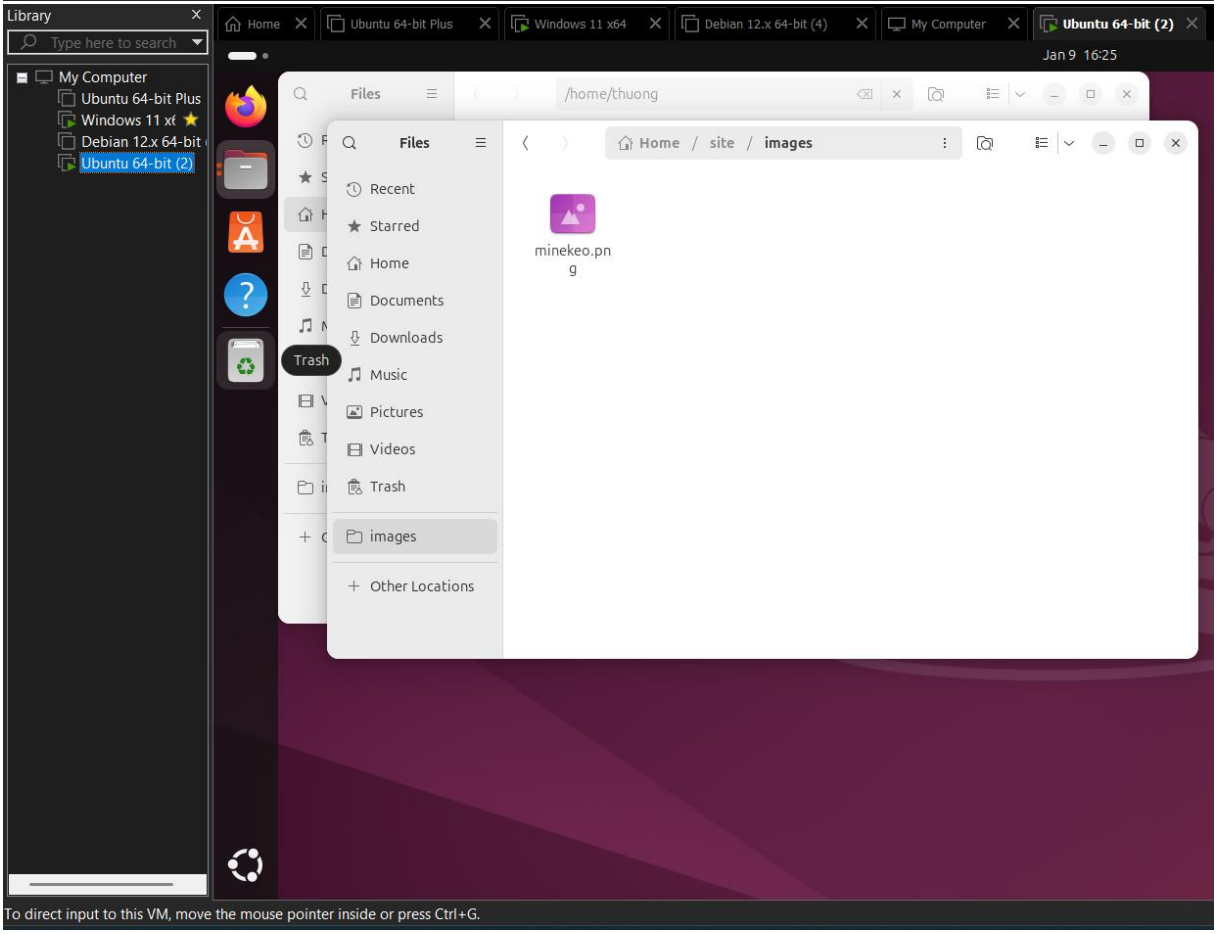
Screenshot IP address Ubuntu VM:



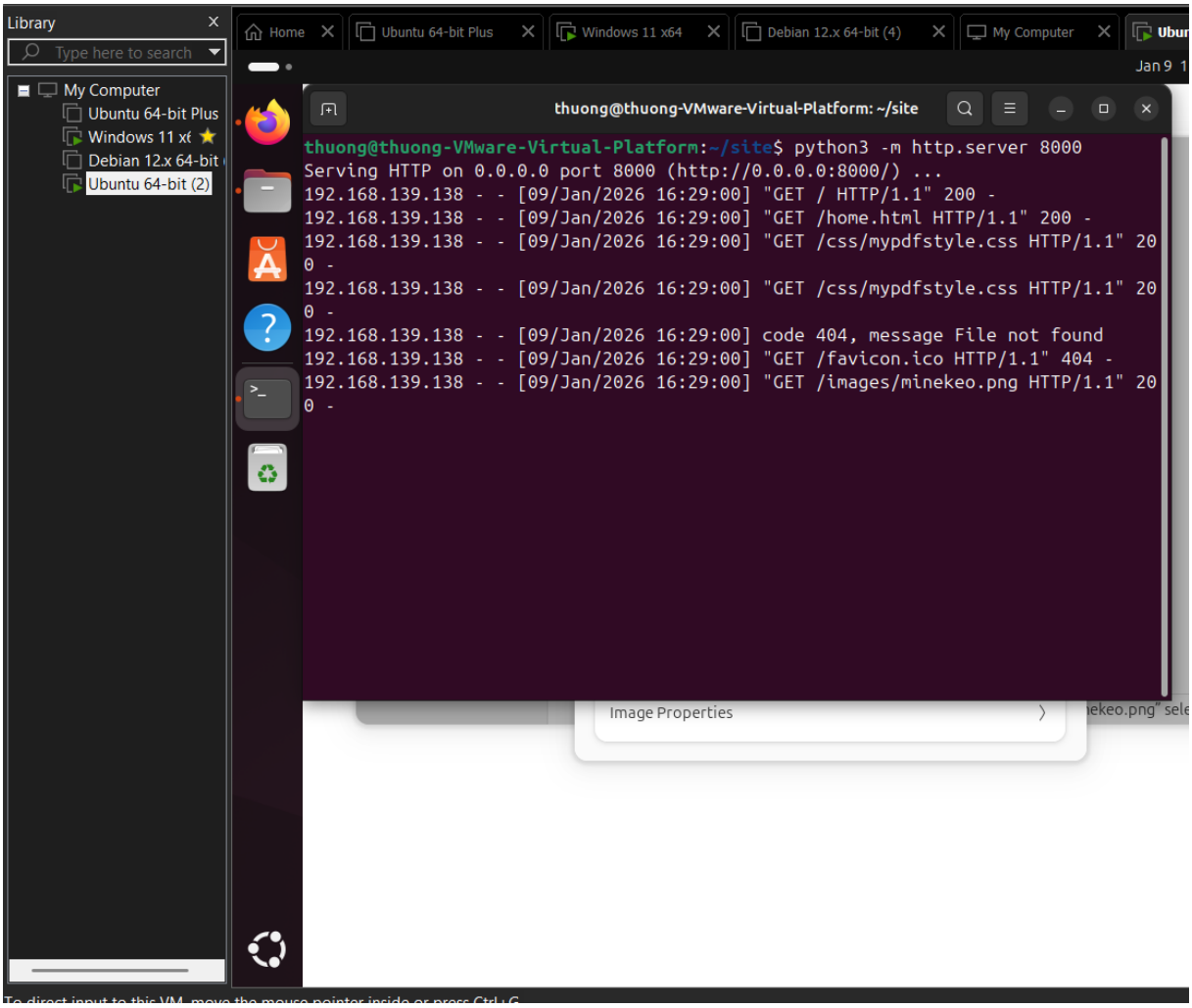
Screenshot of Site directory contents:



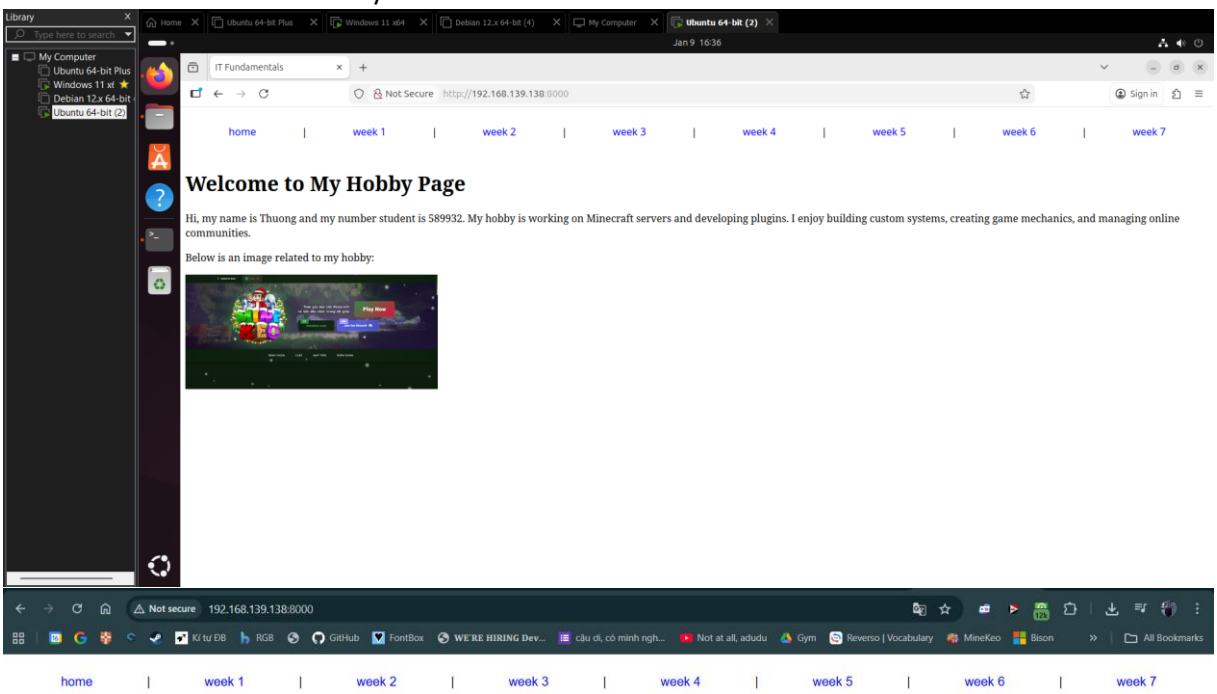

Library
Type here to search
My Computer
Ubuntu 64-bit Plus
Windows 11 x64
Debian 12.x 64-bit
Ubuntu 64-bit (2)
Home
Ubuntu 64-bit Plus
Windows 11 x64
Debian 12.x 64-bit (4)
My Computer
Ubuntu 64-bit (2)
Jan 9 16:25
Files
/home/thuong
Home / site
Recent
Starred
Home
Documents
Downloads
Music
Pictures
Videos
Trash
images
Other Locations
css
images
pdf
home.html
index.html
week1.html
week2.html
week3.html
week4.html
week5.html
week6.html
week7.html
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

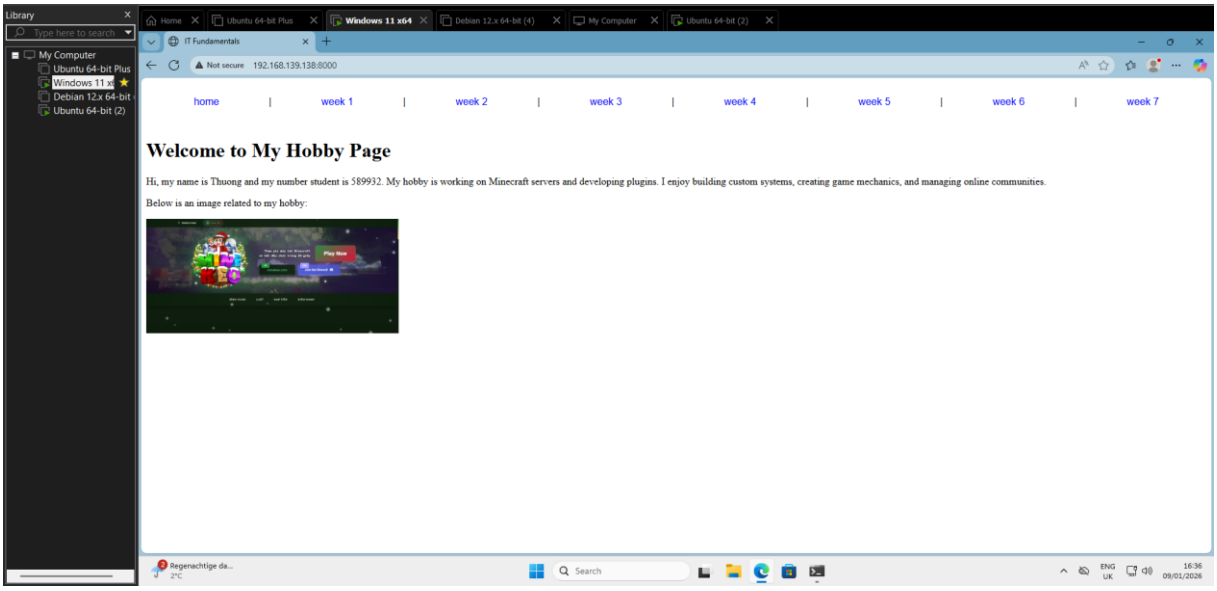

Library
Type here to search
My Computer
Ubuntu 64-bit Plus
Windows 11 x64
Debian 12.x 64-bit
Ubuntu 64-bit (2)
Home
Ubuntu 64-bit Plus
Windows 11 x64
Debian 12.x 64-bit (4)
My Computer
Ubuntu 64-bit (2)
Jan 9 16:25
Files
/home/thuong
Home / site / images
Recent
Starred
Home
Documents
Downloads
Music
Pictures
Videos
Trash
images
Other Locations
minekeo.png
Trash
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Screenshot python3 webserver command:



Screenshot web browser visits your site





Assignment 6.5: Network segment

Remember that bitwise java application you've made in week 2? Expand that application so that you can also calculate a network segment as explained in the PowerPoint slides of week 6. Use the bitwise & AND operator. You need to be able to input two Strings. An IP address and a subnet.

IP: 192.168.1.100 and subnet: 255.255.255.224 for /27

```
Example: 192.168.1.100/27
Calculate the network segment
IP Address:   11000000.10101000.00000001.01100100
Subnet Mask:  11111111.11111111.11111111.11100000
-----------------------------------------------
Network Addr: 11000000.10101000.00000001.01100000

This gives 192.168.1.96 in decimal as the network address.
For a /27 subnet, each segment (or subnet) has 32 IP addresses (2⁵).
The range of this network segment is from 192.168.1.96 to 192.168.1.127.
```

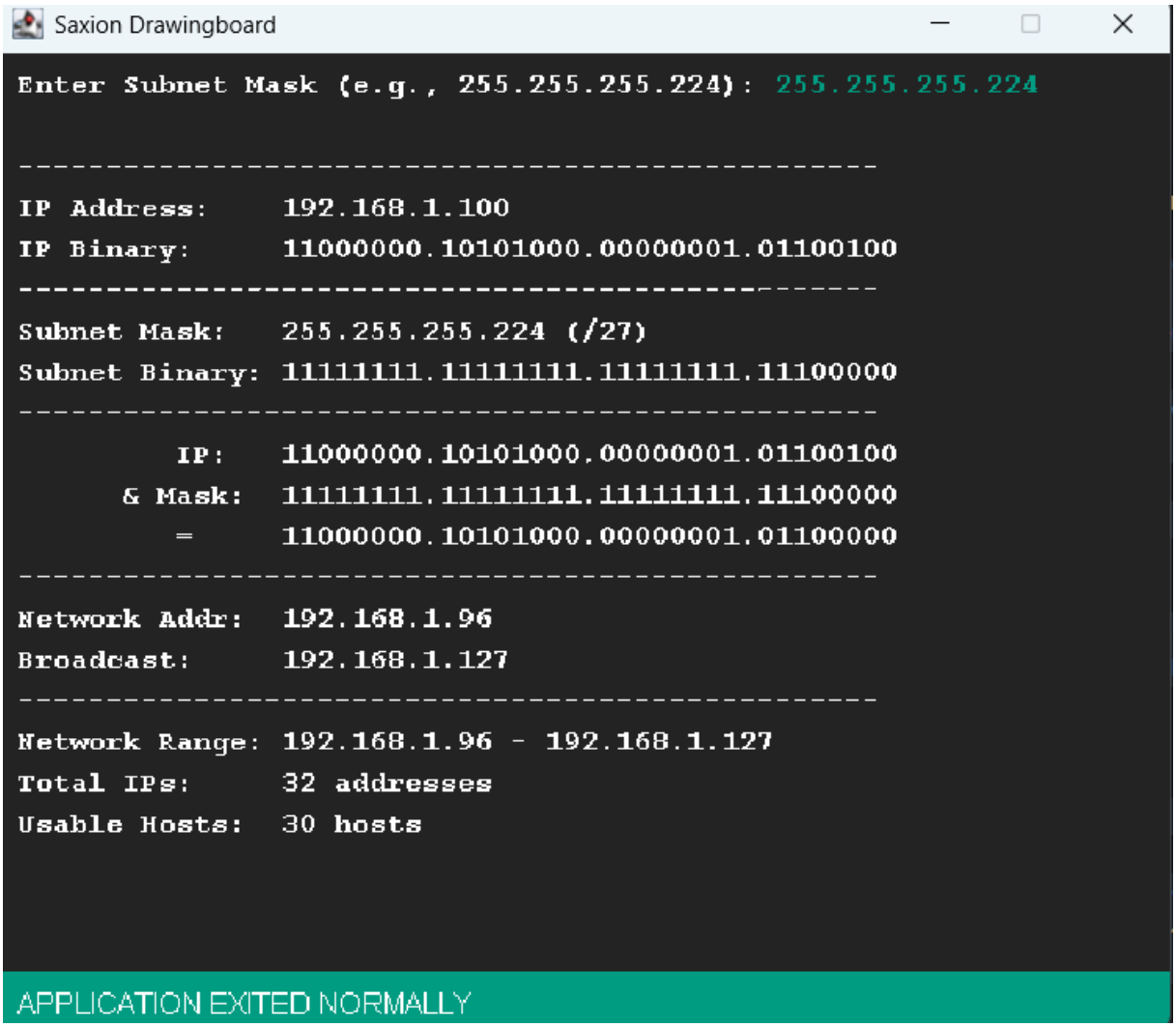
Paste source code here, with a screenshot of a working application.

```
import nl.saxion.app.SaxionApp;

public class Application implements Runnable {

    public static void main(String[] args) {
        SaxionApp.start(new Application(), 600, 500);
    }

    public void run() {
        SaxionApp.printLine("=== BITWISE CALCULATOR ===");
        SaxionApp.printLine("Choose an option:");
        SaxionApp.printLine("1. Is number odd?");
        SaxionApp.printLine("2. Is number a power of 2?");
        SaxionApp.printLine("3. Two's complement");
        SaxionApp.printLine("4. Calculate Network Segment");
        int choice = SaxionApp.readInt("Your choice: ");

        switch (choice) {
            case 1:
                SaxionApp.printLine("Enter a number: ");
                int num1 = SaxionApp.readInt();
                if (isOdd(num1)) SaxionApp.printLine("The number is odd.");
```

```
                else SaxionApp.printLine("The number is even.");
                break;

            case 2:
                SaxionApp.printLine("Enter a number: ");
                int num2 = SaxionApp.readInt();
                if (isPowerOfTwo(num2)) SaxionApp.printLine("The number IS
a power of 2.");
                else SaxionApp.printLine("The number is NOT a power of
2.");
                break;

            case 3:
                SaxionApp.printLine("Enter a number: ");
                int num3 = SaxionApp.readInt();
                int result = twosComplement(num3);
                SaxionApp.printLine("Two's complement: " + result);
                break;

            case 4:
                calculateNetworkSegment();
                break;

            default:
                SaxionApp.printLine("Invalid option.");
        }
    }

    boolean isOdd(int number) {
        return (number & 1) == 1;
    }

    boolean isPowerOfTwo(int number) {
        if (number <= 0) return false;
        return (number & (number - 1)) == 0;
    }

    int twosComplement(int number) {
        return (~number) + 1;
    }

    void calculateNetworkSegment() {
        SaxionApp.printLine("\n=== NETWORK SEGMENT CALCULATOR ===");

        SaxionApp.print("Enter IP Address (e.g., 192.168.1.100): ");
        String ipAddress = SaxionApp.readString();

        SaxionApp.print("Enter Subnet Mask (e.g., 255.255.255.224): ");
        String subnetMask = SaxionApp.readString();
```

```
        int[] ipParts = parseIP(ipAddress);
        int[] subnetParts = parseIP(subnetMask);

        int[] networkParts = new int[4];
        for (int i = 0; i < 4; i++) {
            networkParts[i] = ipParts[i] & subnetParts[i];
        }

        int cidr = calculateCIDR(subnetParts);

        int hostBits = 32 - cidr;
        int totalAddresses = (int) Math.pow(2, hostBits);

        int[] broadcastParts = new int[4];
        int[] invertedMask = new int[4];
        for (int i = 0; i < 4; i++) {
            invertedMask[i] = 255 - subnetParts[i];
            broadcastParts[i] = networkParts[i] | invertedMask[i];
        }

        SaxionApp.printLine("\n-----------------------------------------
-----");
        SaxionApp.printLine("IP Address:    " + ipAddress);
        SaxionApp.printLine("IP Binary:     " + toBinaryString(ipParts));
        SaxionApp.printLine("-----------------------------------------
---");
        SaxionApp.printLine("Subnet Mask:   " + subnetMask + " (/" + cidr +
")");
        SaxionApp.printLine("Subnet Binary: " +
toBinaryString(subnetParts));
        SaxionApp.printLine("-----------------------------------------
---");
        SaxionApp.printLine("         IP:   " + toBinaryString(ipParts));
        SaxionApp.printLine("      & Mask:  " +
toBinaryString(subnetParts));
        SaxionApp.printLine("         =     " +
toBinaryString(networkParts));
        SaxionApp.printLine("-----------------------------------------
---");
        SaxionApp.printLine("Network Addr:  " + formatIP(networkParts));
        SaxionApp.printLine("Broadcast:     " + formatIP(broadcastParts));
        SaxionApp.printLine("-----------------------------------------
---");
        SaxionApp.printLine("Network Range: " + formatIP(networkParts) + "
- " + formatIP(broadcastParts));
        SaxionApp.printLine("Total IPs:     " + totalAddresses + "
addresses");
        SaxionApp.printLine("Usable Hosts:  " + (totalAddresses - 2) + "
hosts");
    }
```

```
    int[] parseIP(String ip) {
        String[] parts = ip.split("\\.");
        int[] result = new int[4];
        for (int i = 0; i < 4; i++) {
            result[i] = Integer.parseInt(parts[i]);
        }
        return result;
    }

    String toBinaryString(int[] parts) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < 4; i++) {
            String binary = Integer.toBinaryString(parts[i]);
            while (binary.length() < 8) {
                binary = "0" + binary;
            }
            sb.append(binary);
            if (i < 3) sb.append(".");
        }
        return sb.toString();
    }

    String formatIP(int[] parts) {
        return parts[0] + "." + parts[1] + "." + parts[2] + "." + parts[3];
    }

    int calculateCIDR(int[] subnetParts) {
        int cidr = 0;
        for (int i = 0; i < 4; i++) {
            int octet = subnetParts[i];
            while (octet > 0) {
                cidr += (octet & 1);
                octet = octet >> 1;
            }
        }
        return cidr;
    }
}
```



Ready? Save this file and export it as a pdf file with the name: **week6.pdf**