

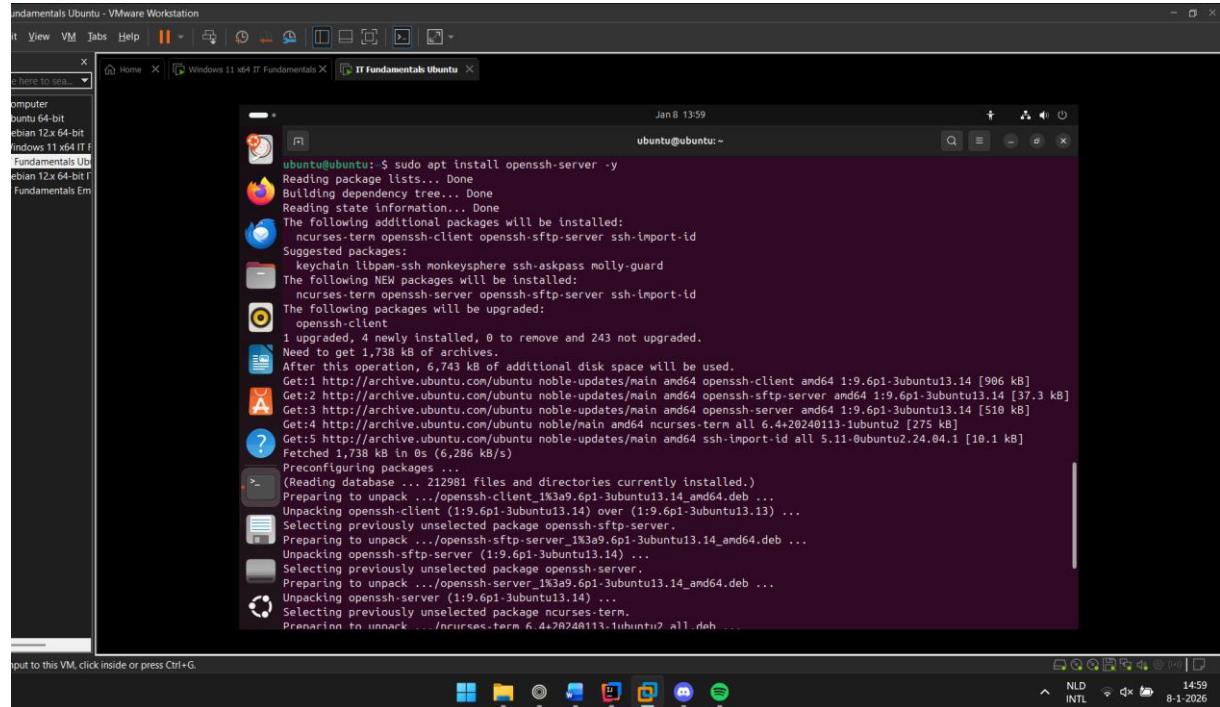
Template Week 6 – Networking

Student number: 579675

Assignment 6.1: Working from home

Screenshot installation openssh-server:

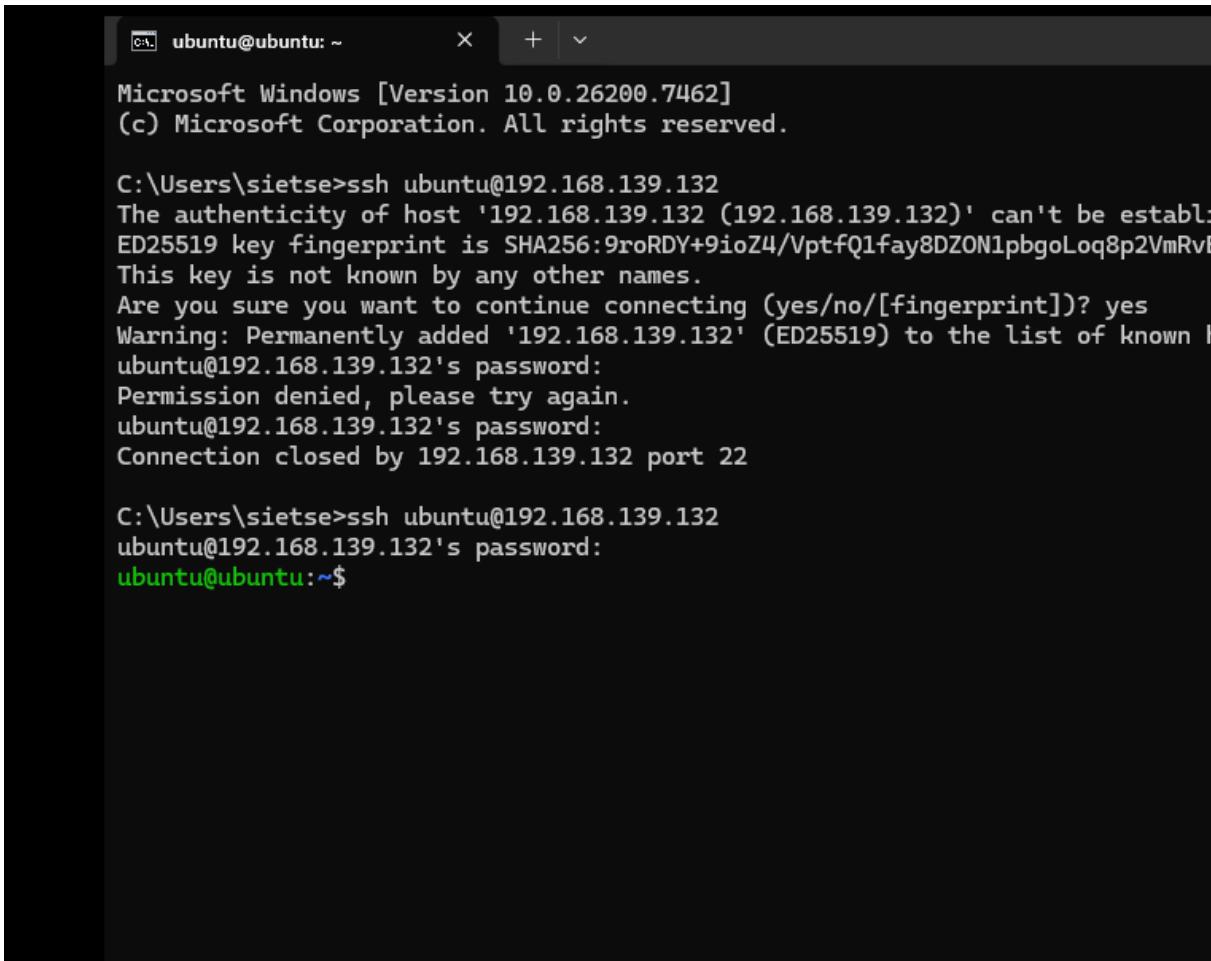
Ubuntu IP: 192.169.139.132



```
ubuntu@ubuntu:~$ sudo apt install openssh-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-client openssh-sftp-server ssh-import-id
  Suggested packages:
    keychain libpam-ssh monkeysphere ssh-askpass molly-guard
  The following NEW packages will be installed:
    ncurses-term openssh-server openssh-sftp-server ssh-import-id
  The following packages will be upgraded:
  openssh-client
  1 upgraded, 4 newly installed, 0 to remove and 243 not upgraded.
Need to get 1,738 kB of archives.
After this operation, 6,743 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-client amd64 1:9.6p1-3ubuntu13.14 [906 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-sftp-server amd64 1:9.6p1-3ubuntu13.14 [37.3 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-server amd64 1:9.6p1-3ubuntu13.14 [510 kB]
Get:4 http://archive.ubuntu.com/ubuntu/noble/main amd64 ncurses-term all 6.4+20240113-1ubuntu2 [275 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 ssh-import-id all 5.11-0ubuntu2.24.84.1 [10.1 kB]
Fetched 1,738 kB in 0s (6,286 kB/s)
Preconfiguring packages ...
(Reading database ... 212981 files and directories currently installed.)
Preparing to unpack .../openssh-client_1%3a9.6p1-3ubuntu13.14_amd64.deb ...
Unpacking openssh-client (1:9.6p1-3ubuntu13.14) over (1:9.6p1-3ubuntu13.13) ...
Selecting previously unselected package openssh-sftp-server.
Preparing to unpack .../openssh-sftp-server_1%3a9.6p1-3ubuntu13.14_amd64.deb ...
Unpacking openssh-sftp-server (1:9.6p1-3ubuntu13.14) ...
Selecting previously unselected package openssh-server.
Preparing to unpack .../openssh-server_1%3a9.6p1-3ubuntu13.14_amd64.deb ...
Unpacking openssh-server (1:9.6p1-3ubuntu13.14) ...
Selecting previously unselected package ncurses-term.
Preparing to unpack .../ncurses-term_6.4+20240113-1ubuntu2_all.deb ...

```

Screenshot successful SSH command execution:

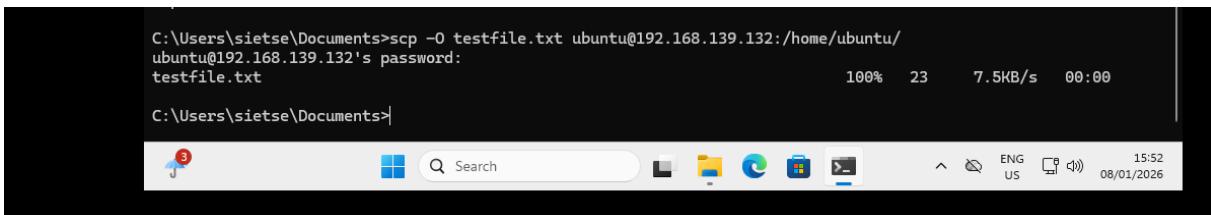


```
ubuntu@ubuntu: ~      X + v
Microsoft Windows [Version 10.0.26200.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sietse>ssh ubuntu@192.168.139.132
The authenticity of host '192.168.139.132 (192.168.139.132)' can't be established.
ED25519 key fingerprint is SHA256:9roRDY+9ioZ4/VptfQ1fay8DZON1pbgoLoq8p2VmRvE.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.139.132' (ED25519) to the list of known hosts.
ubuntu@192.168.139.132's password:
Permission denied, please try again.
ubuntu@192.168.139.132's password:
Connection closed by 192.168.139.132 port 22

C:\Users\sietse>ssh ubuntu@192.168.139.132
ubuntu@192.168.139.132's password:
ubuntu@ubuntu:~$
```

Screenshot successful execution SCP command:

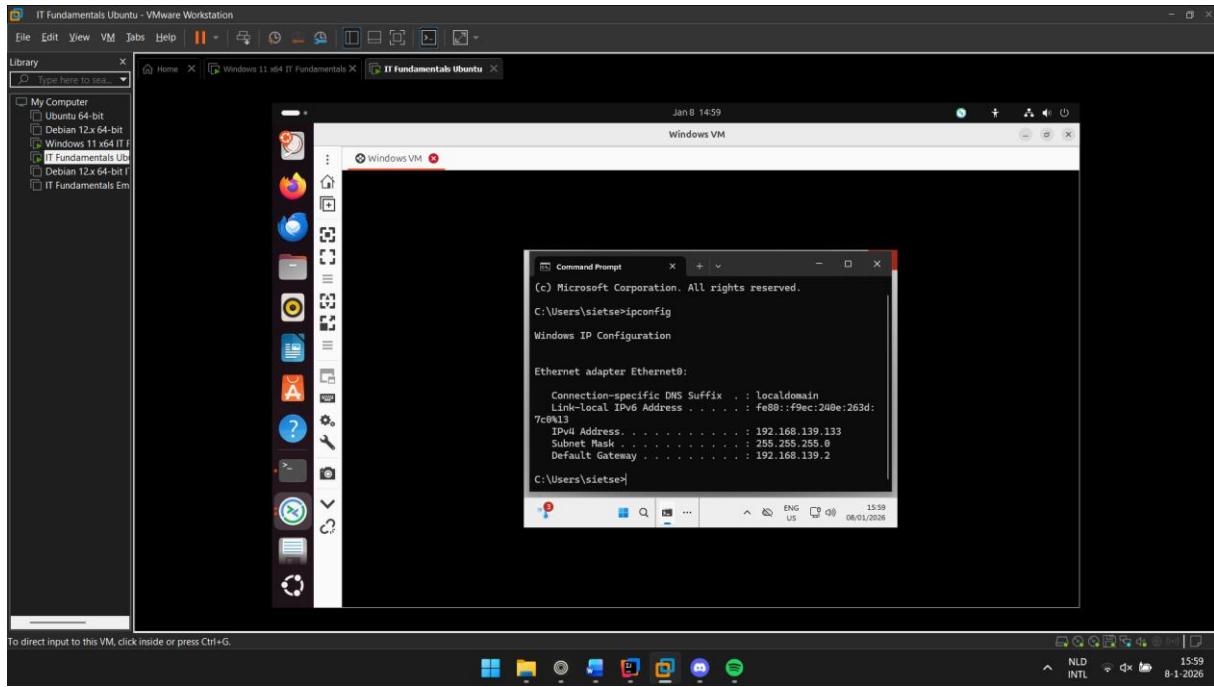


```
C:\Users\sietse\Documents>scp -O testfile.txt ubuntu@192.168.139.132:/home/ubuntu/
ubuntu@192.168.139.132's password:                                          100%   23    7.5KB/s  00:00
testfile.txt

C:\Users\sietse\Documents>
```

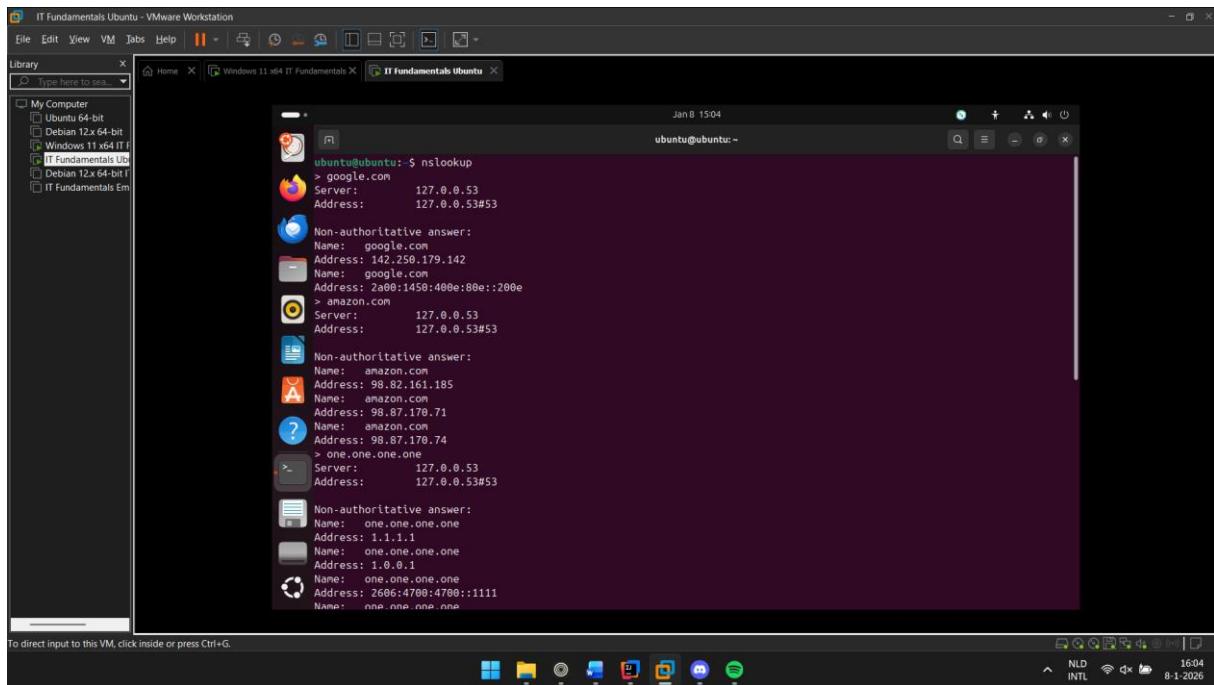
Screenshot remmina:

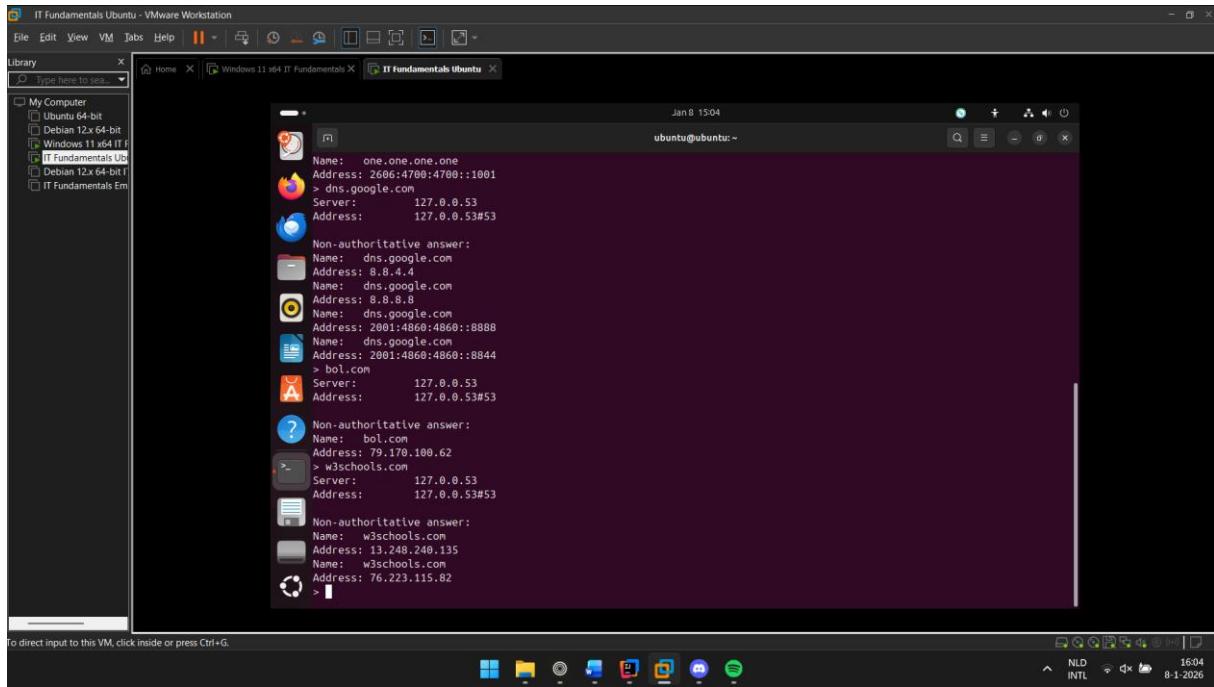
WindowsVM IP: 192.168.139.133



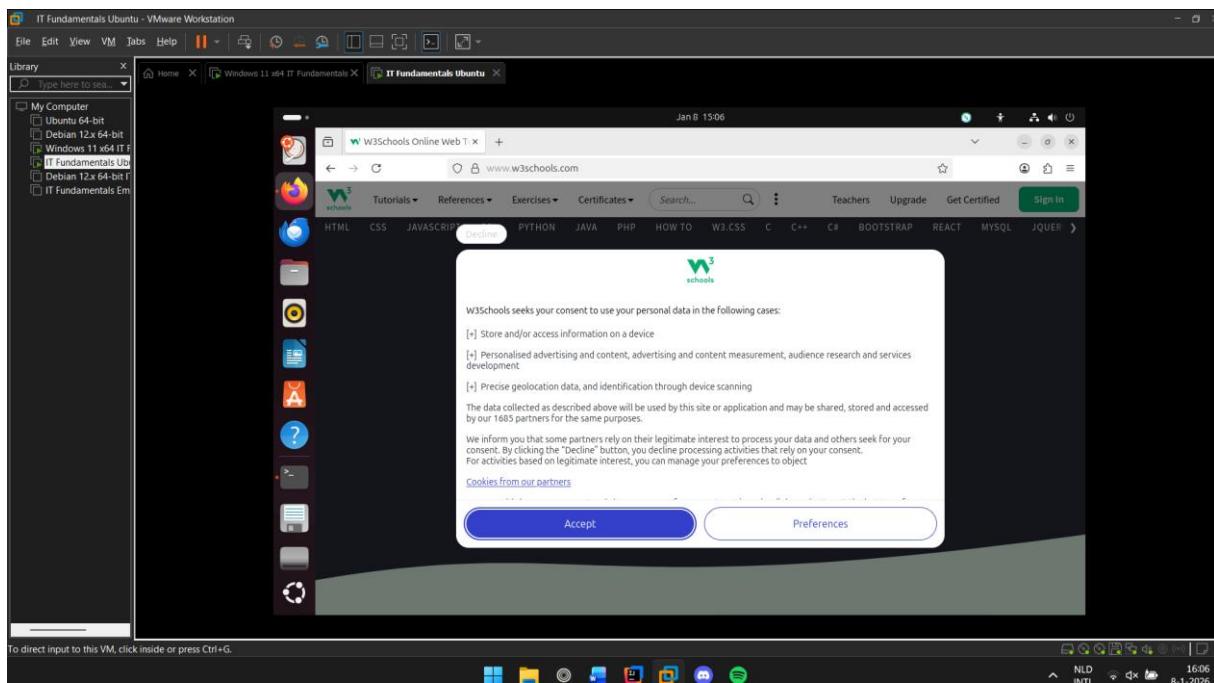
Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:





Screenshot website visit via IP address:



It redirected me to the site with DNS, so I cannot see the IP in the searchbar

Assignment 6.3: subnetting

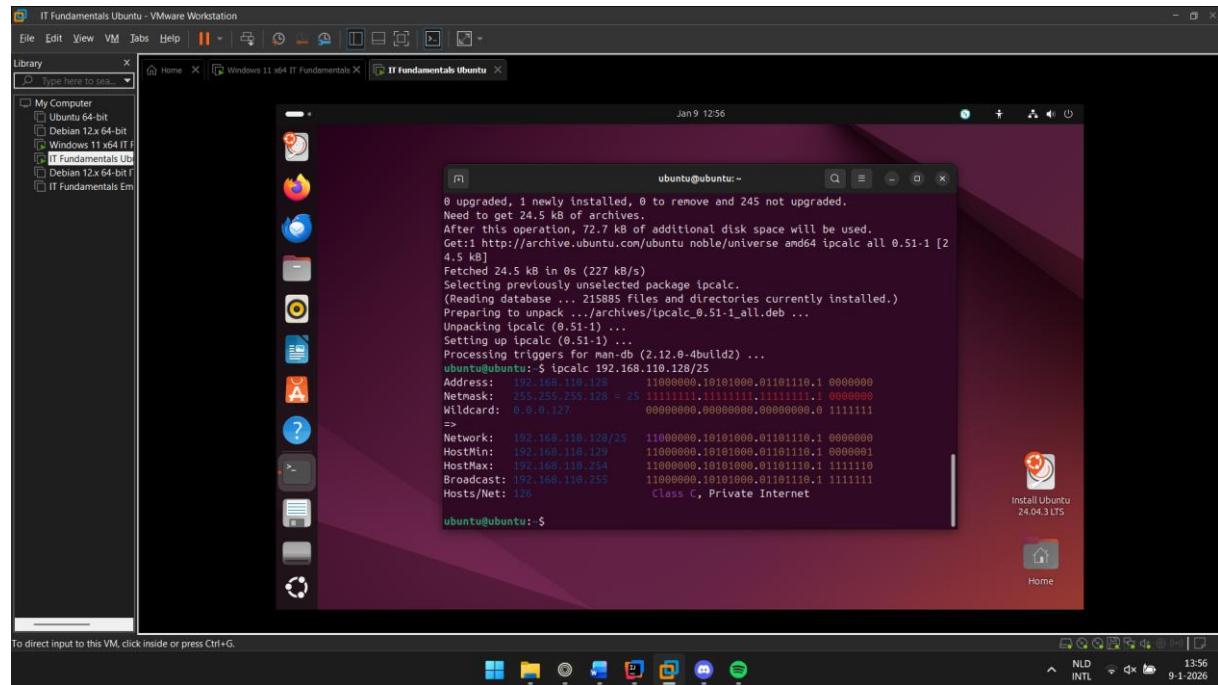
How many IP addresses are in this network configuration 192.168.110.128/25?

The number after the ip address itself '/25' stands for the amount of bits of the 32 that get used for the network, which leaves 7 bits remaining for hosts. A bit has 2 value options. So 2 to the power of 7 would give you 128 ip addresses.

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

192.168.110.129 to 192.168.110.254 is the usable IP range, because the first '128' is reserved for the network. And the last '.255' is reserved for broadcasting

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

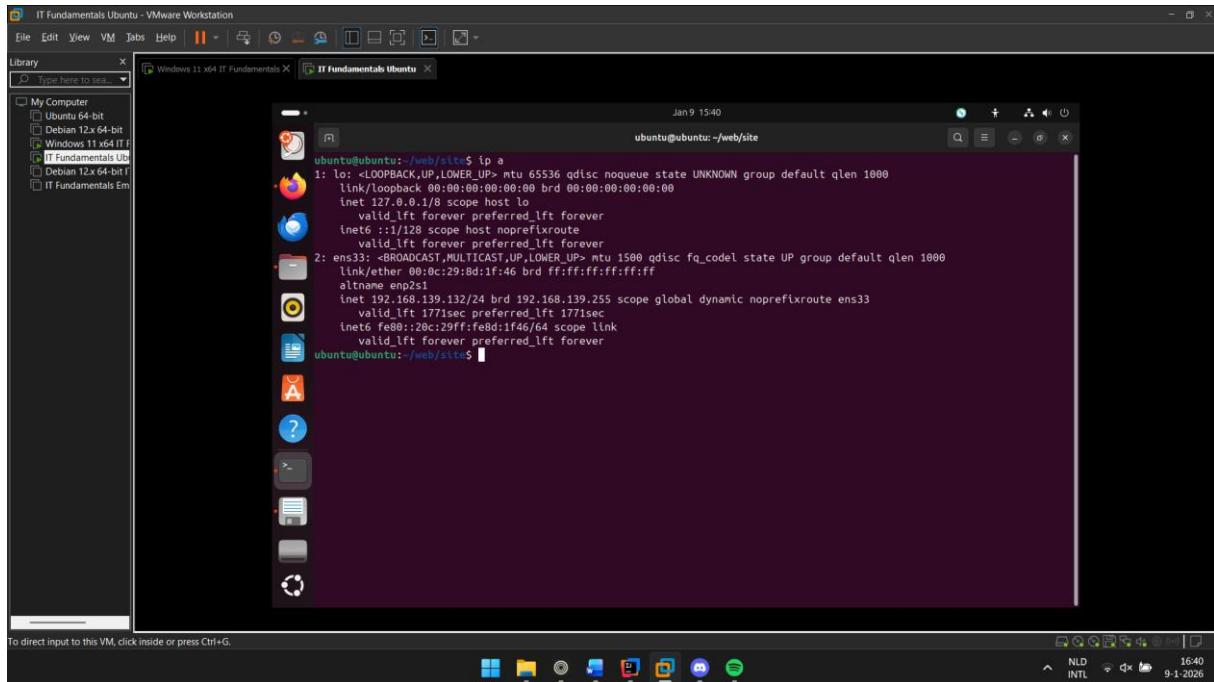


Explain the above calculation in your own words.

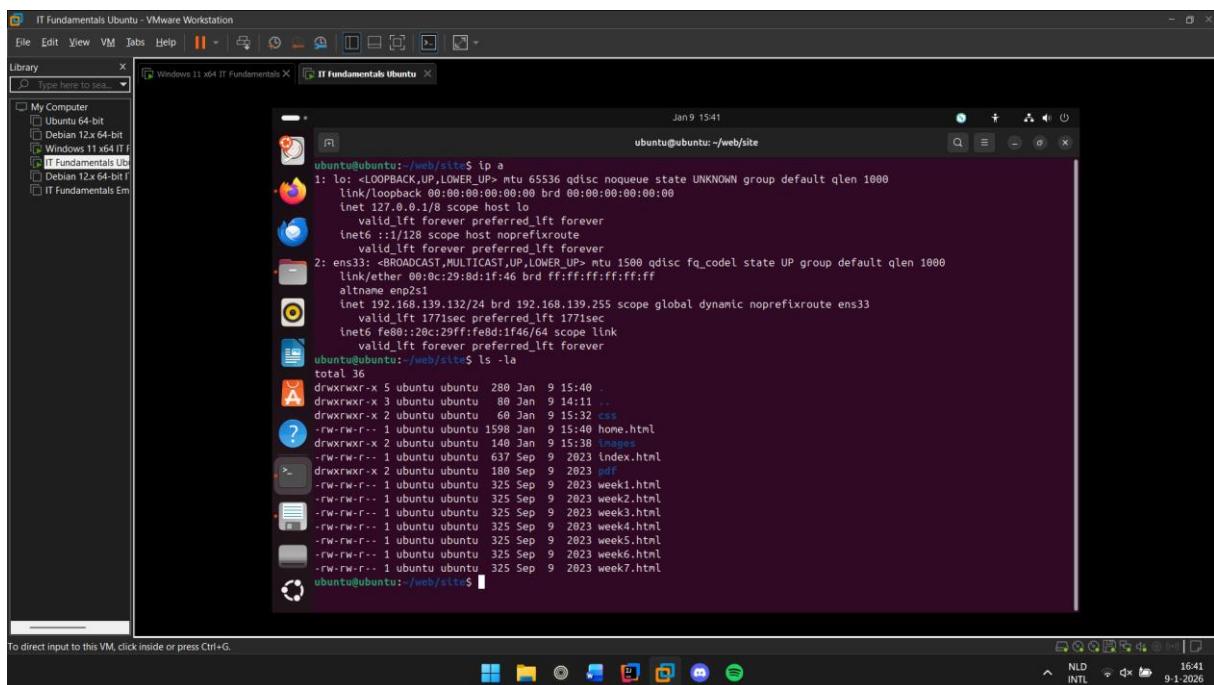
As I said earlier, the /25 means there are 25 bits, of the 32 available bits, used for the network. This leaves us with 7 bits. Each bit allows for two values, zero or one. So we can take 2 to the power of 7 to calculate the possible ip address hosts. This gives us 128 available hosts, however the first is used for the network and the last one is used for broadcast. So that leaves us with 126 available hosts.

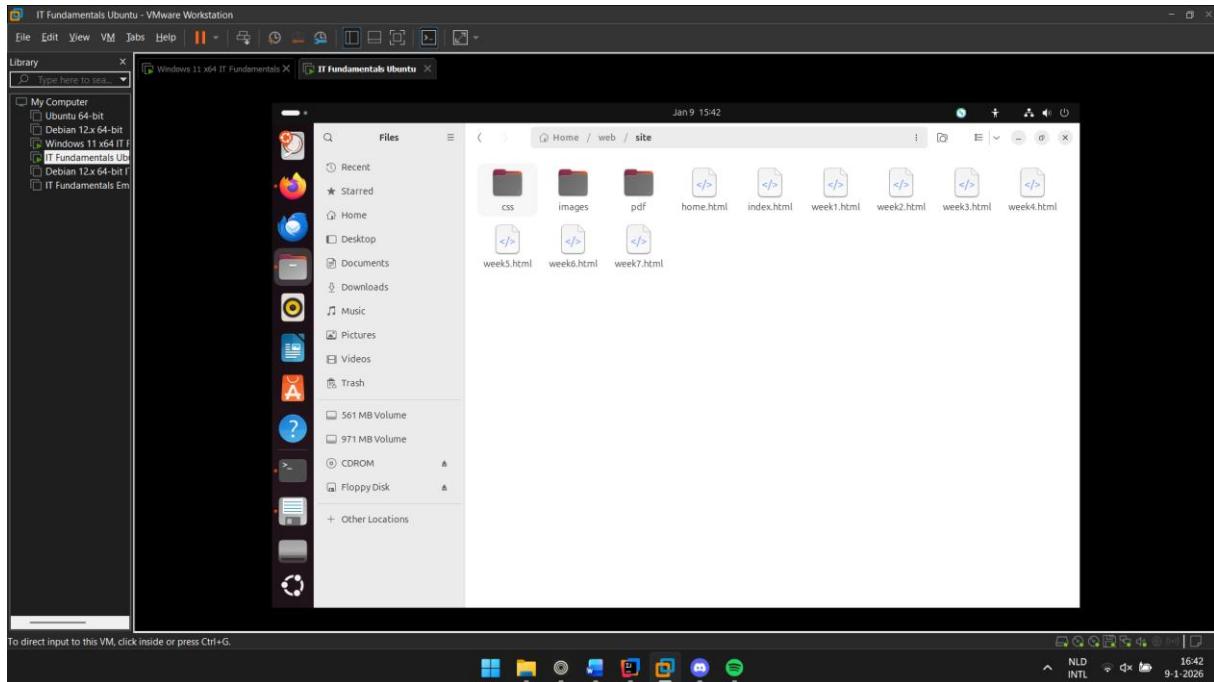
Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

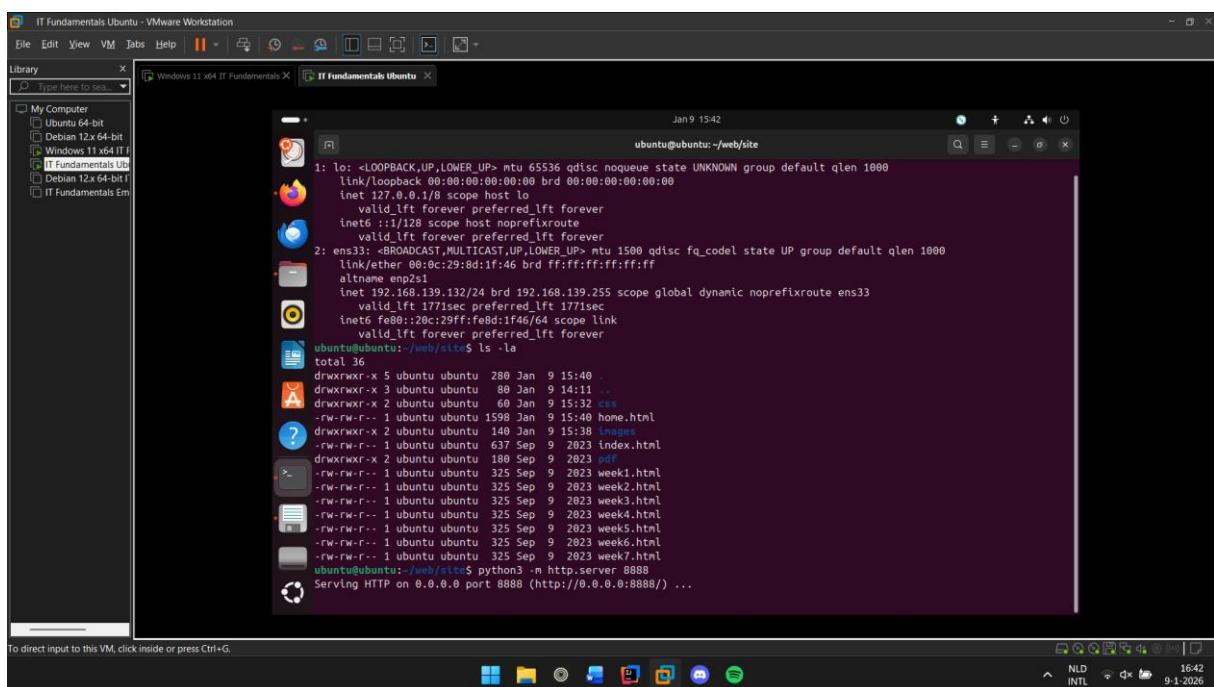


Screenshot of Site directory contents:

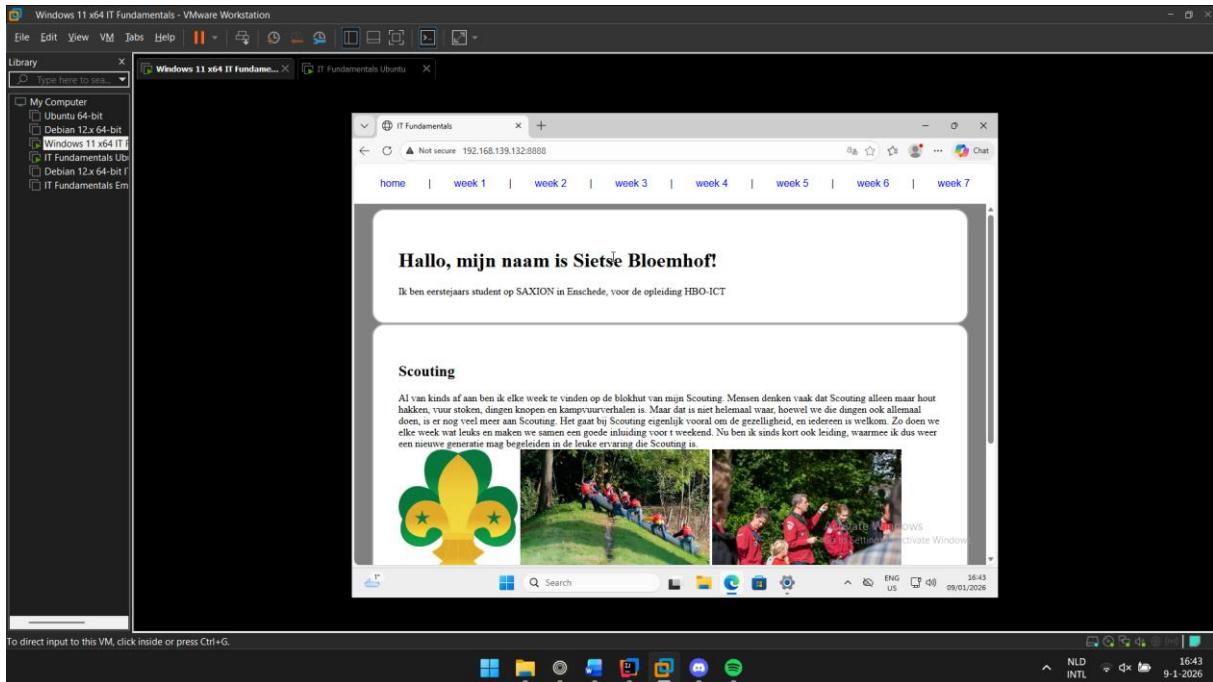




Screenshot python3 webserver command:



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 (2^5).

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 java.util.Scanner;
```

```
public class Main {
```

```
    private static Scanner s = new Scanner(System.in);
```

```

public static void main(String[] args) {

    while(true) {
        printMenu();
        switch(s.nextInt()) {
            case 1:
                printIfNumIsOdd();
                break;
            case 2:
                printIfNumIsPowerOfTwo();
                break;
            case 3:
                printTwosComplementOfNum();
                break;
            case 4:
                calculateNetworkSegment();
                break;
            case 5:
                return;
            default:
                System.out.println("Sorry, that is not a valid option");
        }
    }
}

public static void printMenu() {
    System.out.println("1. Is number odd?");
    System.out.println("2. Is number a power of 2?");
    System.out.println("3. Two's complement of number?");
    System.out.println("4. Calculate network segment?");
    System.out.println("5. Exit");
    System.out.print("Please choose one of the above options: ");
}

public static void printIfNumIsOdd() {
    System.out.print("Please enter a number: ");
    int number = s.nextInt();

    if((number & 0b0001) == 1) System.out.println("number is odd");
    else System.out.println("number is not odd");
}

public static void printIfNumIsPowerOfTwo() {
    System.out.print("Please enter a number: ");
    int number = s.nextInt();
    if(number > 0 && (number & (number-1)) == 0) System.out.println("Number is to the power of
2");
    else System.out.println("Number is not to the power of two");
}

```

```

}

public static void printTwosComplementOfNum () {
    System.out.print("Please enter a number: ");
    int number = s.nextInt();
    System.out.println(~number + 1);
}

public static void calculateNetworkSegment() {
    System.out.print("Please enter an IP address: ");
    String ipAddr = s.next();
    System.out.print("Please enter an IP subnet: ");
    String subnet = s.next();

    String[] ipParts = ipAddr.split("\\.");
    String[] subnetParts = subnet.split("\\.");

    StringBuilder networkAddr = new StringBuilder();
    StringBuilder broadcastAddr = new StringBuilder();

    for (int i = 0; i < 4; i++) {
        int ipPart = Integer.parseInt(ipParts[i]);
        int subnetPart = Integer.parseInt(subnetParts[i]);
        int networkPart = ipPart & subnetPart;
        int broadcastPart = networkPart | (~subnetPart & 255);

        networkAddr.append(networkPart);
        broadcastAddr.append(broadcastPart);
        if (i < 3) {
            networkAddr.append(".");
            broadcastAddr.append(".");
        }
    }

    System.out.println("Network address: " + networkAddr);
    System.out.println("Range: " + networkAddr + " - " + broadcastAddr);
}
}

```

```
> /EXTERNAL LIBRARIES
Run Main ×
↶ ↷ 📸 ↵ ⌂ : 
↑ C:\Users\bloem\.jdks\ms-21.0.8\bin\java.exe "-"
↓ 1. Is number odd?
≡ 2. Is number a power of 2?
≡ 3. Two's complement of number?
≡ 4. Calculate network segment?
⎙ 5. Exit
trash Please choose one of the above options: 4
Please enter an IP address: 192.168.1.100
Please enter an IP subnet: 255.255.255.224
Network address: 192.168.1.96
Range: 192.168.1.96 - 192.168.1.127
```

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