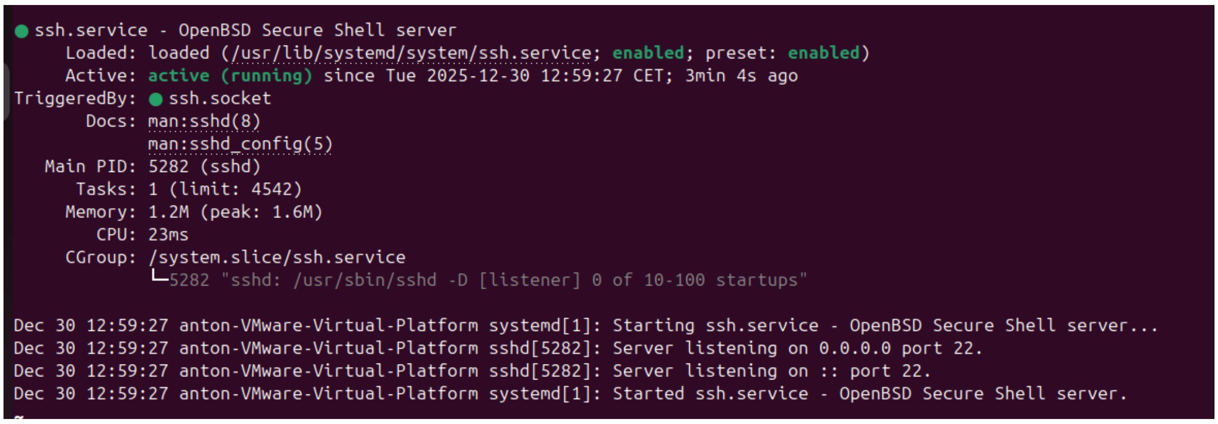
Week 6 – Networking

Student number: 581124

Assignment 6.1: Working from home

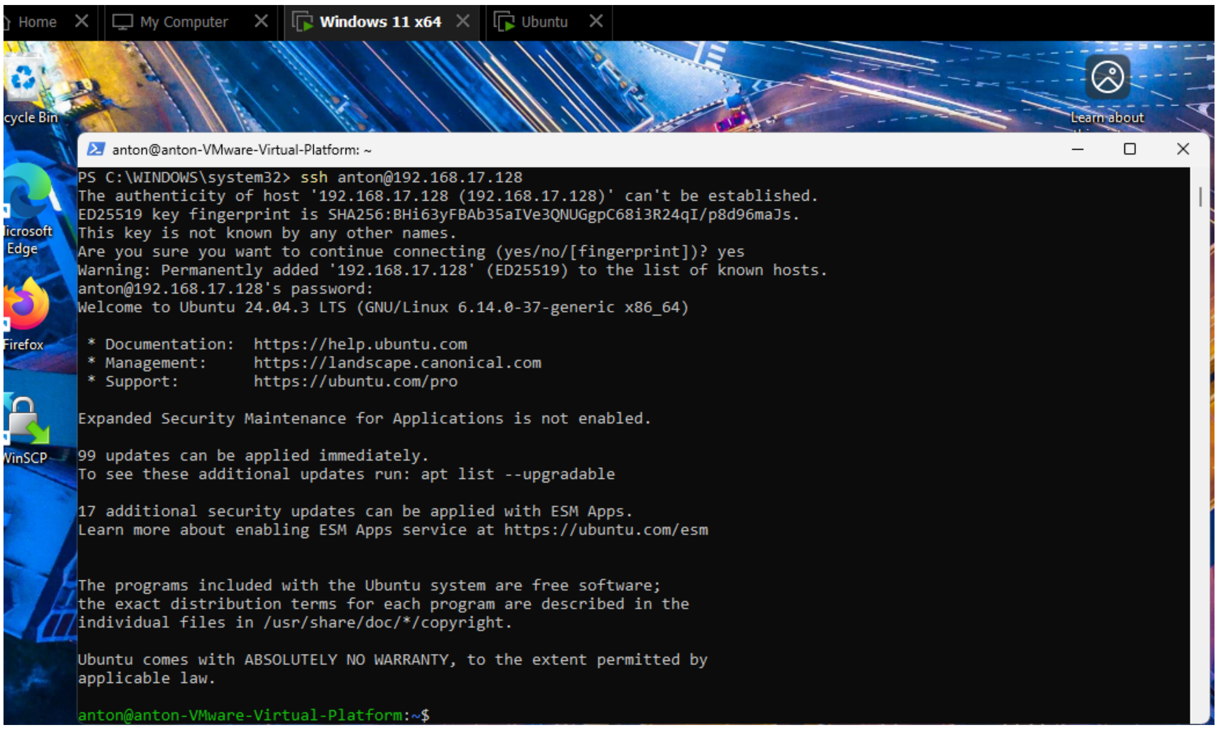
Screenshot installation openssh-server:



```
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Tue 2025-12-30 12:59:27 CET; 3min 4s ago
TriggeredBy: ● ssh.socket
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 5282 (sshd)
      Tasks: 1 (limit: 4542)
     Memory: 1.2M (peak: 1.6M)
        CPU: 23ms
     CGroup: /system.slice/ssh.service
             └─5282 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Dec 30 12:59:27 anton-VMware-Virtual-Platform systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Dec 30 12:59:27 anton-VMware-Virtual-Platform sshd[5282]: Server listening on 0.0.0.0 port 22.
Dec 30 12:59:27 anton-VMware-Virtual-Platform sshd[5282]: Server listening on :: port 22.
Dec 30 12:59:27 anton-VMware-Virtual-Platform systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
```

It shows that ssh server is installed and working properly(so openssh is installed)

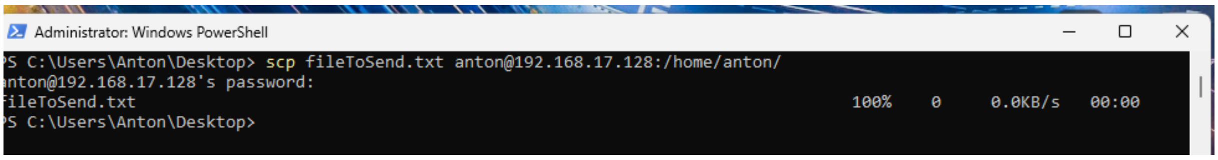
Screenshot successful SSH command execution:



```
PS C:\WINDOWS\system32> ssh anton@192.168.17.128
The authenticity of host '192.168.17.128 (192.168.17.128)' can't be established.
ED25519 key fingerprint is SHA256:BHi63yFBAb35aIVe3QNUGgpC68i3R24qI/p8d96maJs.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.17.128' (ED25519) to the list of known hosts.
anton@192.168.17.128's password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-37-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

99 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

17 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

anton@anton-VMware-Virtual-Platform:~$
```

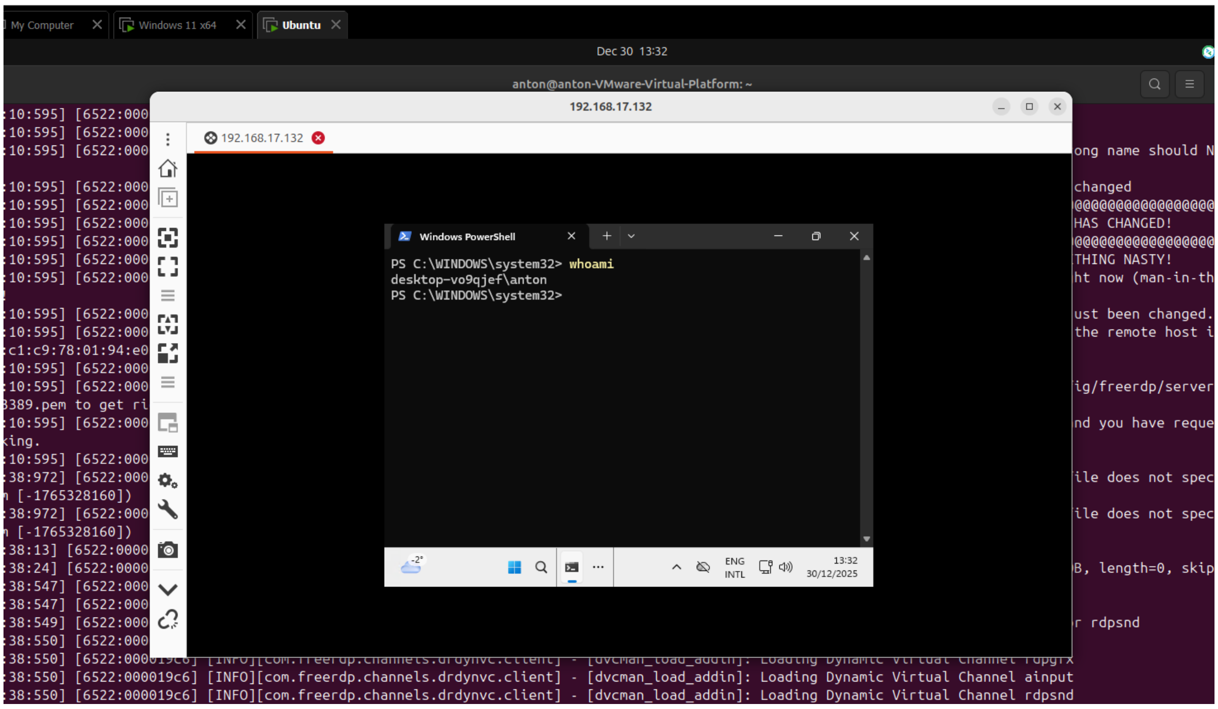
Successfully logged in Ubuntu via command prompt via the ssh.

Screenshot successful execution SCP command:



```
PS C:\Users\Anton\Desktop> scp fileToSend.txt anton@192.168.17.128:/home/anton/
anton@192.168.17.128's password:
fileToSend.txt                                                    100%    0     0.0KB/s   00:00
PS C:\Users\Anton\Desktop>
```

Screenshot remmina:

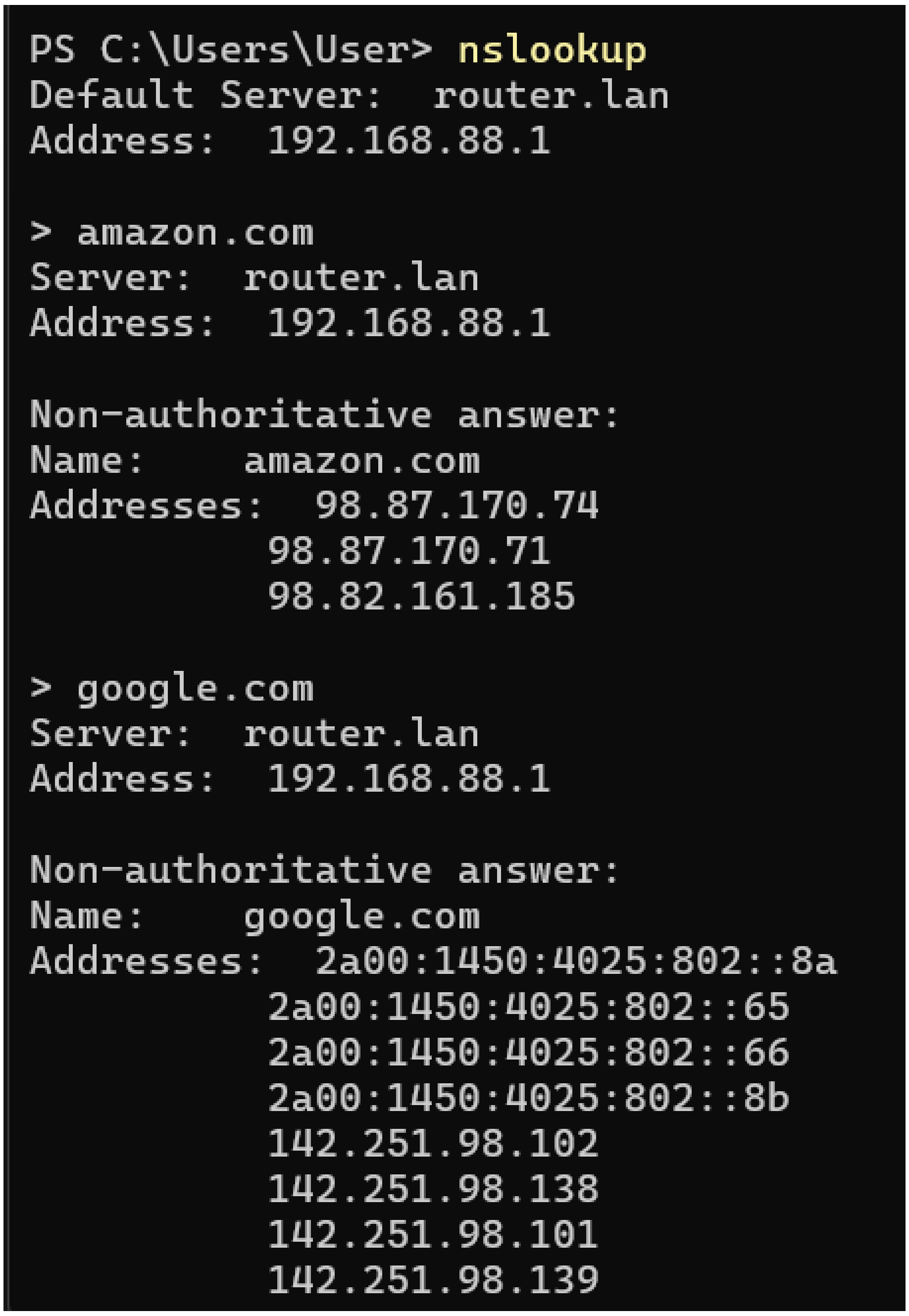


```
PS C:\WINDOWS\system32> whoami
desktop-vo9qjef\anton
PS C:\WINDOWS\system32>
```

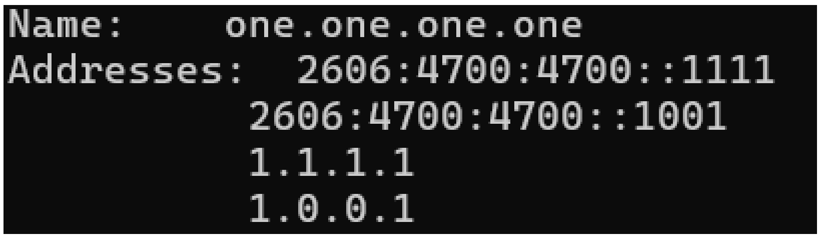
```
10:595] [6522:000
10:595] [6522:000
10:595] [6522:000                                                     ong name should N

10:595] [6522:000                                                     changed
10:595] [6522:000                                                     @@@@@@@@@@@@@@@@@
10:595] [6522:000                                                     HAS CHANGED!
10:595] [6522:000                                                     @@@@@@@@@@@@@@@@@
10:595] [6522:000                                                     THING NASTY!
10:595] [6522:000                                                     ht now (man-in-th

10:595] [6522:000                                                     ust been changed.
10:595] [6522:000                                                     the remote host i
c1:c9:78:01:94:e0
10:595] [6522:000
10:595] [6522:000                                                     ig/freerdp/server
3389.pem to get ri
10:595] [6522:000                                                     nd you have reque
king.
10:595] [6522:000
38:972] [6522:000                                                     ile does not spec
n [-1765328160])
38:972] [6522:000                                                     ile does not spec
n [-1765328160])
38:13] [6522:0000
38:24] [6522:0000                                                     B, length=0, skip
38:547] [6522:000
38:547] [6522:000
38:549] [6522:000                                                     r rdpsnd
38:550] [6522:000
38:550] [6522:000019c6] [INFO][com.freerdp.channels.drdynvc.client] - [dvcman_load_addin]: Loading Dynamic Virtual Channel rdpgfx
38:550] [6522:000019c6] [INFO][com.freerdp.channels.drdynvc.client] - [dvcman_load_addin]: Loading Dynamic Virtual Channel ainput
38:550] [6522:000019c6] [INFO][com.freerdp.channels.drdynvc.client] - [dvcman_load_addin]: Loading Dynamic Virtual Channel rdpsnd
```

Assignment 6.2: IP addresses websites

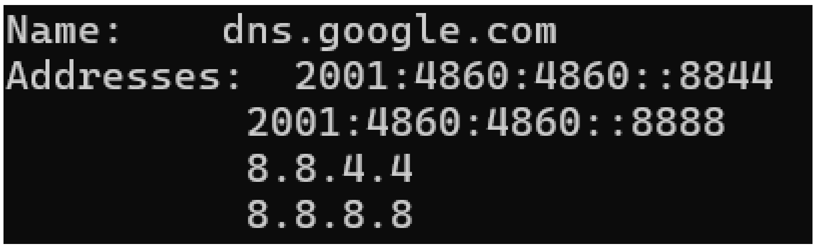
Relevant screenshots nslookup command:



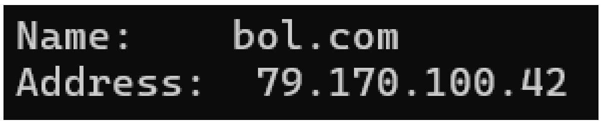
```
PS C:\Users\User> nslookup
Default Server:  router.lan
Address:  192.168.88.1

> amazon.com
Server:  router.lan
Address:  192.168.88.1

Non-authoritative answer:
Name:    amazon.com
Addresses:  98.87.170.74
          98.87.170.71
          98.82.161.185

> google.com
Server:  router.lan
Address:  192.168.88.1

Non-authoritative answer:
Name:    google.com
Addresses:  2a00:1450:4025:802::8a
          2a00:1450:4025:802::65
          2a00:1450:4025:802::66
          2a00:1450:4025:802::8b
          142.251.98.102
          142.251.98.138
          142.251.98.101
          142.251.98.139
```



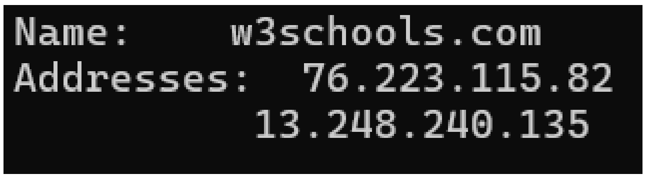
```
Name:    one.one.one.one
Addresses:  2606:4700:4700::1111
          2606:4700:4700::1001
          1.1.1.1
          1.0.0.1
```



```
Name:    dns.google.com
Addresses:  2001:4860:4860::8844
          2001:4860:4860::8888
          8.8.4.4
          8.8.8.8
```

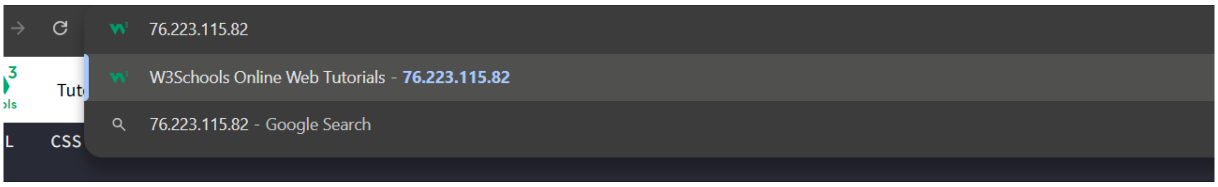


```
Name:    bol.com
Address:  79.170.100.42
```



```
Name:    w3schools.com
Addresses:  76.223.115.82
          13.248.240.135
```

Screenshot website visit via IP address:



76.223.115.82

W3Schools Online Web Tutorials - 76.223.115.82

76.223.115.82 - Google Search

Assignment 6.3: subnetting

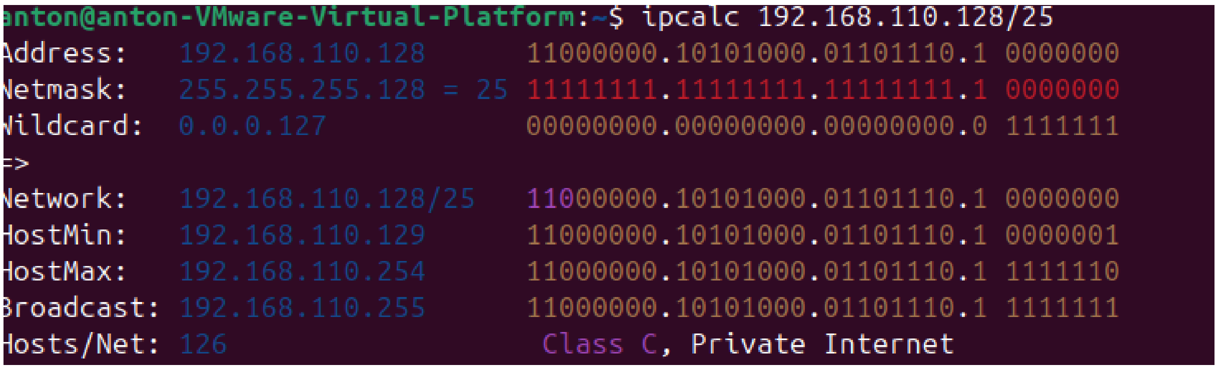
How many IP addresses are in this network configuration 192.168.110.128/25?

128

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

192.168.110.129 – 192.168.110.254

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



```
anton@anton-VMware-Virtual-Platform:~$ ipcalc 192.168.110.128/25
Address:   192.168.110.128      11000000.10101000.01101110.1 0000000
Netmask:   255.255.255.128 = 25 11111111.11111111.11111111.1 0000000
Wildcard:  0.0.0.127            00000000.00000000.00000000.0 1111111
=>
Network:   192.168.110.128/25   11000000.10101000.01101110.1 0000000
HostMin:   192.168.110.129      11000000.10101000.01101110.1 0000001
HostMax:   192.168.110.254      11000000.10101000.01101110.1 1111110
Broadcast: 192.168.110.255      11000000.10101000.01101110.1 1111111
Hosts/Net: 126                   Class C, Private Internet
```

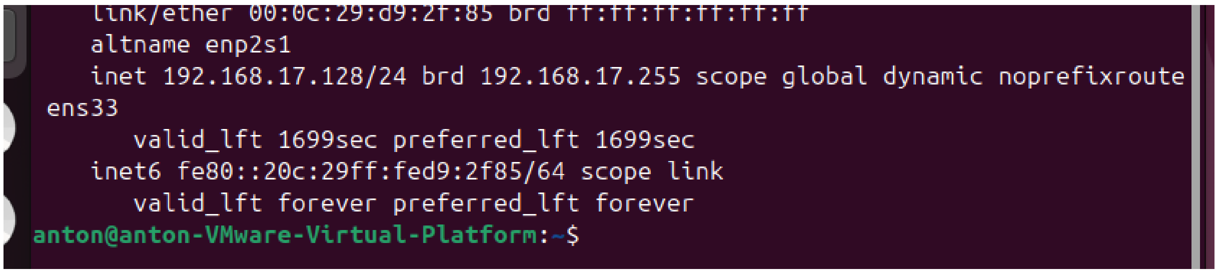
Explain the above calculation in your own words.

To calculate the number of IP addresses, I raised 2 to the power of (32-25), which results into 128 addresses.

To determine the usable IP range, I converted the given IP address and Network Mask into binary and used bitwise operator AND between the two of them. It resulted in the network address (the first IP address in the network). Also the last address can't be used because it's broadcast one, so the usable range is between the network address and broadcast one (129-254).

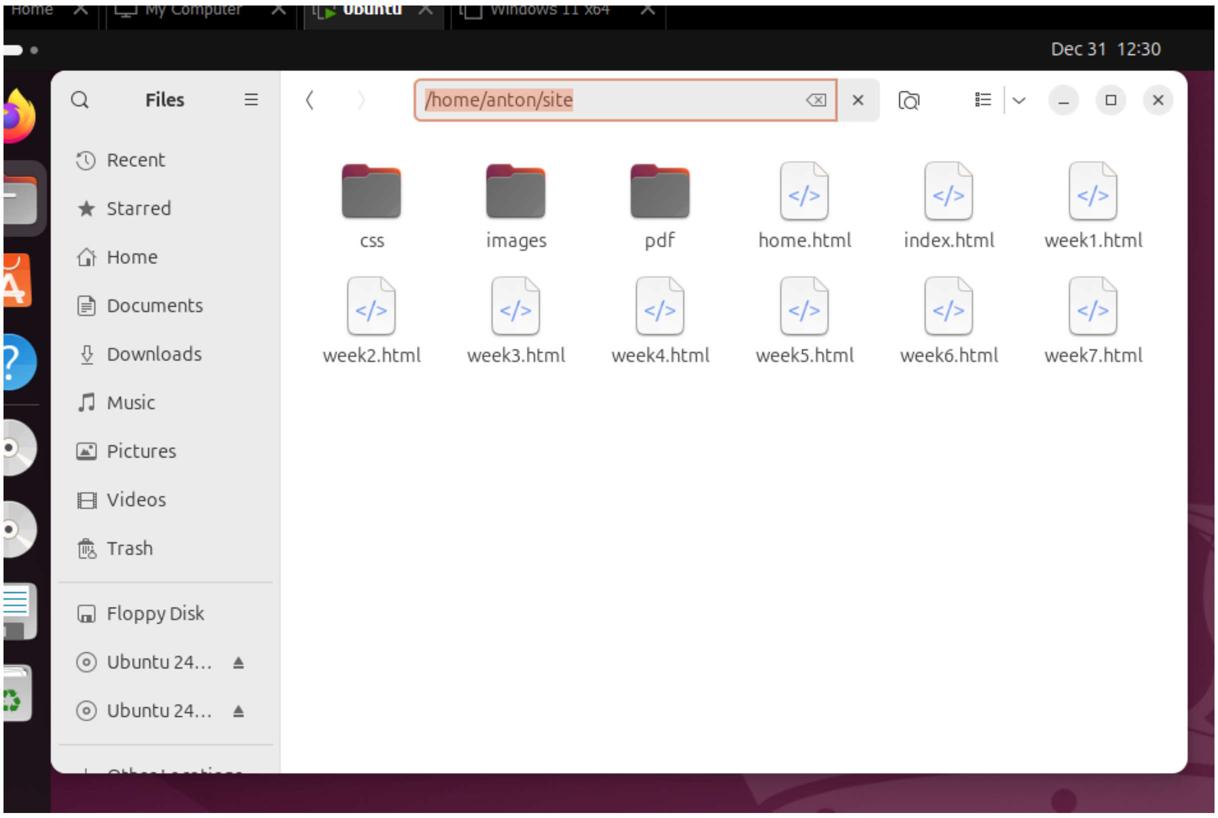
Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

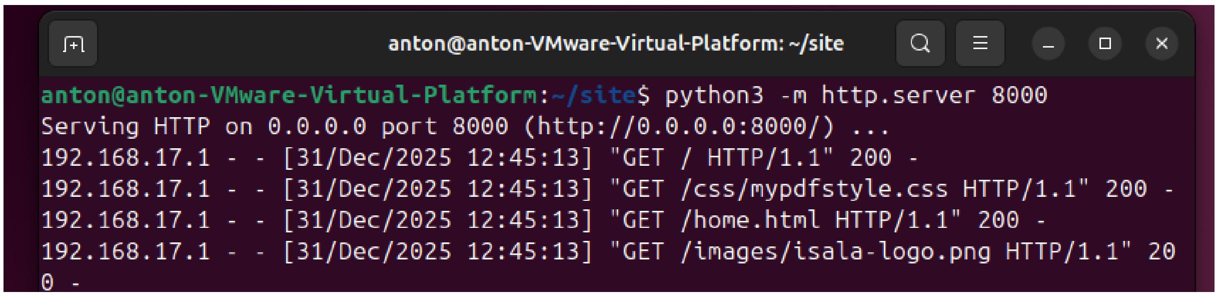


```
    link/ether 00:0c:29:d9:2f:85 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.17.128/24 brd 192.168.17.255 scope global dynamic noprefixroute
 ens33
       valid_lft 1699sec preferred_lft 1699sec
    inet6 fe80::20c:29ff:fed9:2f85/64 scope link
       valid_lft forever preferred_lft forever
anton@anton-VMware-Virtual-Platform:~$
```

Screenshot of Site directory contents:

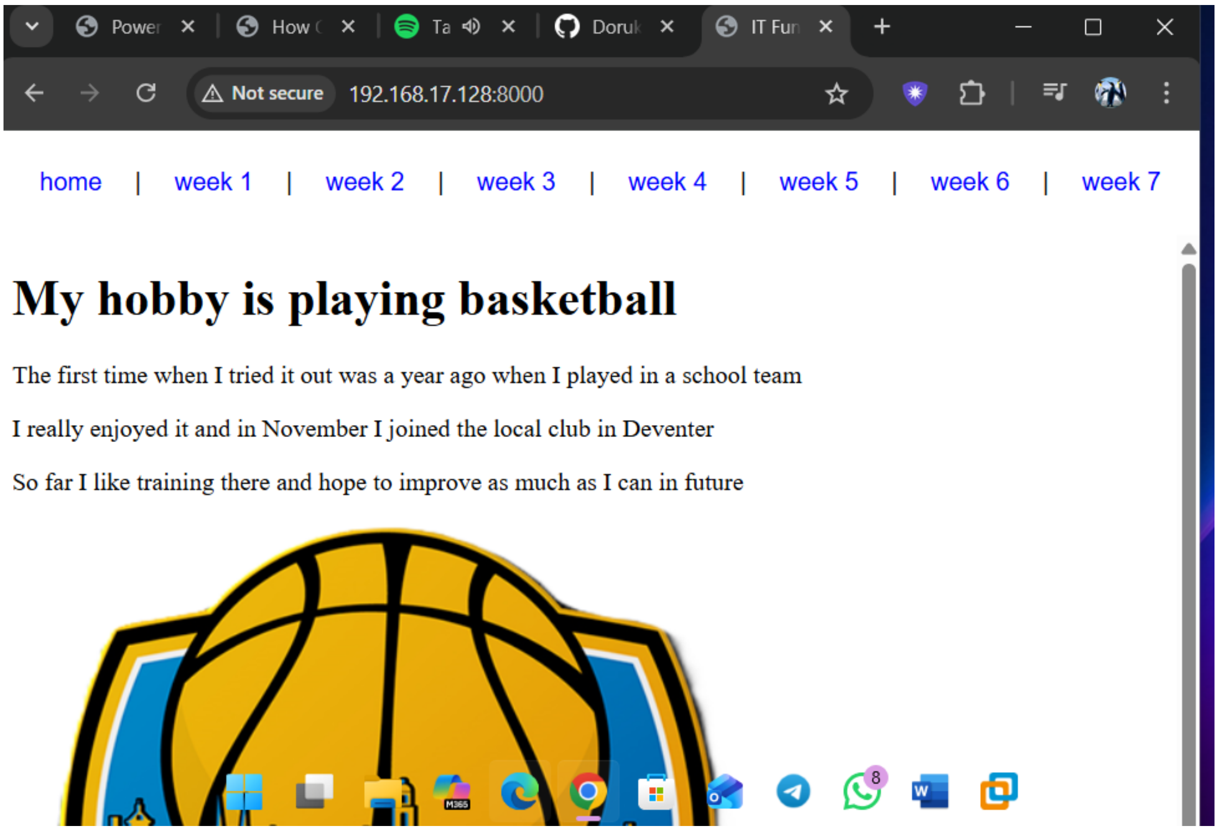


Screenshot python3 webserver command:



```
anton@anton-VMware-Virtual-Platform:~/site$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.17.1 - - [31/Dec/2025 12:45:13] "GET / HTTP/1.1" 200 -
192.168.17.1 - - [31/Dec/2025 12:45:13] "GET /css/mypdfstyle.css HTTP/1.1" 200 -
192.168.17.1 - - [31/Dec/2025 12:45:13] "GET /home.html HTTP/1.1" 200 -
192.168.17.1 - - [31/Dec/2025 12:45:13] "GET /images/isala-logo.png HTTP/1.1" 20
0 -
```

Screenshot web browser visits your site


Not secure
192.168.17.128:8000
home | week 1 | week 2 | week 3 | week 4 | week 5 | week 6 | week 7

My hobby is playing basketball

The first time when I tried it out was a year ago when I played in a school team

I really enjoyed it and in November I joined the local club in Deventer

So far I like training there and hope to improve as much as I can in future

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;

import java.util.ArrayList;

public class Application implements Runnable {

    public static void main(String[] args) {SaxionApp.start(new Application(), 800, 800);}

    public void run() {
        SaxionApp.print("IP address: ");
        String ip = SaxionApp.readString();

        SaxionApp.print("Subnet Mask: ");
        String subnet = SaxionApp.readString();

        // Parse IP address and subnet mask into integer blocks
        ArrayList<Integer> parsedIP = parseIPv4(ip);
        ArrayList<Integer> parsedSubnet = parseIPv4(subnet);

        // Calculate network and broadcast addresses
        ArrayList<Integer> networkAddress = getNetworkAddress(parsedIP, parsedSubnet);
        ArrayList<Integer> broadcastAddress = getBroadcastAddress(networkAddress, parsedSubnet);

        SaxionApp.printLine("The network address is " + displayIP(networkAddress) + ".");
        SaxionApp.printLine("Each segment has " + getIPAmount(parsedSubnet) + " IP addresses.");
        SaxionApp.printLine("The range of this network segment is from " + displayIP(networkAddress) +
" to " + displayIP(broadcastAddress));}


    //Parses an IPv4 address into four integer blocks
    //Returns null if the format or range is invalid

    public ArrayList<Integer> parseIPv4(String ip) {
        String[] splittedIP = ip.split("\\.");

        if (splittedIP.length != 4) {
            SaxionApp.printLine("Error in parsing");
            return null;
        }

        ArrayList<Integer> blocks = new ArrayList<>();

        // Convert each block to integer and validate range (0–255)
        for (int i = 0; i < splittedIP.length; i++) {
            Integer number = Integer.valueOf(splittedIP[i]);
            if (number <= 255 && number >= 0) {
                blocks.add(number);
            } else {
                SaxionApp.printLine("Invalid IP range");
                return null;
            }
        }

        return blocks;
    }

     //Calculates the network address by applying a bitwise AND & between the IP address and the
subnet mask

    public ArrayList<Integer> getNetworkAddress(ArrayList<Integer> ip, ArrayList<Integer> subnet) {
        ArrayList<Integer> networkAddress = new ArrayList<>();

        if (ip.size() != subnet.size()) {
            SaxionApp.printLine("Error with ip or subnet occurred");
            return null;
        }

        for (int i = 0; i < ip.size(); i++) {
            networkAddress.add(ip.get(i) & subnet.get(i));
        }

        return networkAddress;
    }


    public ArrayList<Integer> getBroadcastAddress(ArrayList<Integer> networkAddress,
ArrayList<Integer> subnet) {
        ArrayList<Integer> broadcastAddress = new ArrayList<>(4);

        for (int i = 0; i < 4; i++) {
            int net = networkAddress.get(i);
            int mask = subnet.get(i);

            int hostMask = 255 - mask;

            // Broadcast address is network OR host mask
            int broadcast = net | hostMask;

            broadcastAddress.add(broadcast);
        }

        return broadcastAddress;
    }

     //Calculates the total number of IP addresses in the subnet based on the number of host bits
    public int getIPAmount(ArrayList<Integer> subnet) {
        int amountOfOnesBits = 0;

        for (int block : subnet) {
            amountOfOnesBits += Integer.bitCount(block);
        }

        int hostBits = 32 - amountOfOnesBits;
        return 1 << hostBits;
    }


     //Converts an IP address from a list of integers into a readable format
    public StringBuilder displayIP(ArrayList<Integer> ip) {
        StringBuilder IP = new StringBuilder();

        for (int i = 0; i < ip.size(); i++) {
            if (i != ip.size() - 1) {
                IP.append(ip.get(i)).append(".");
            } else {
                IP.append(ip.get(i));
            }
        }

        return IP;
    }
}
```