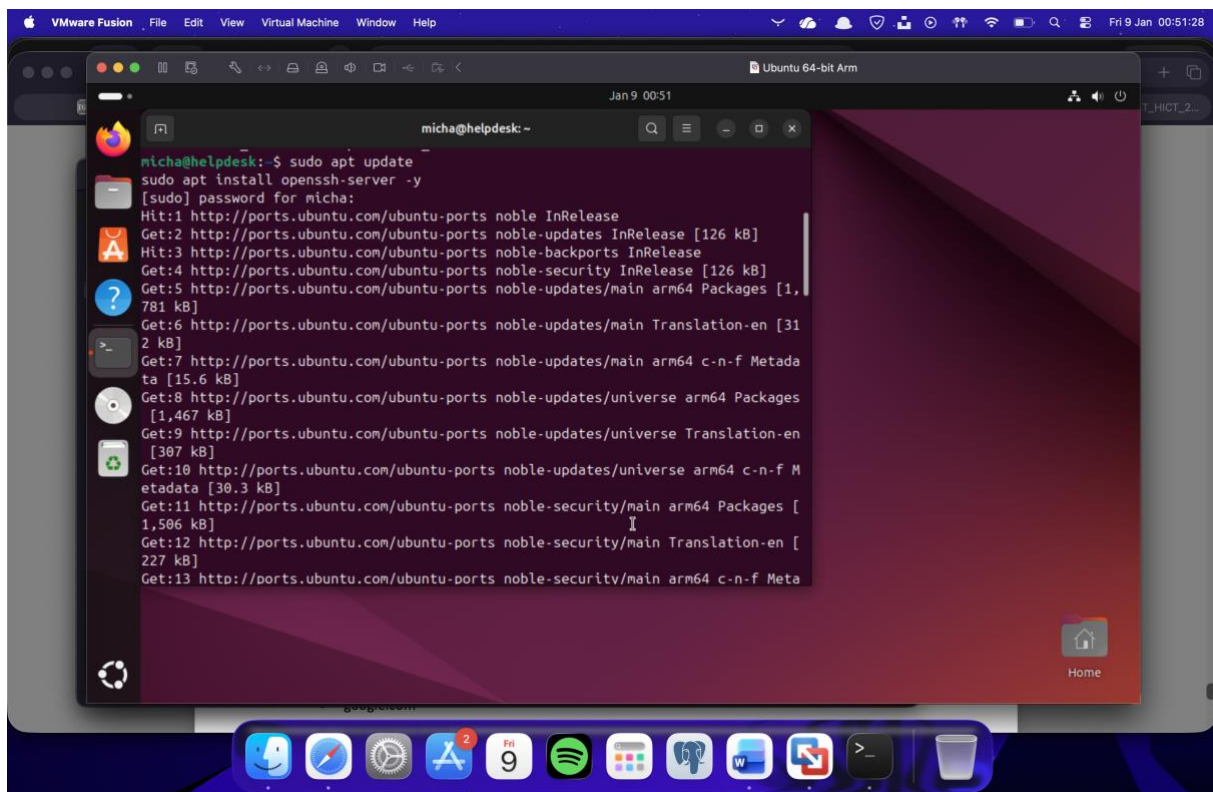


# Template Week 6 – Networking

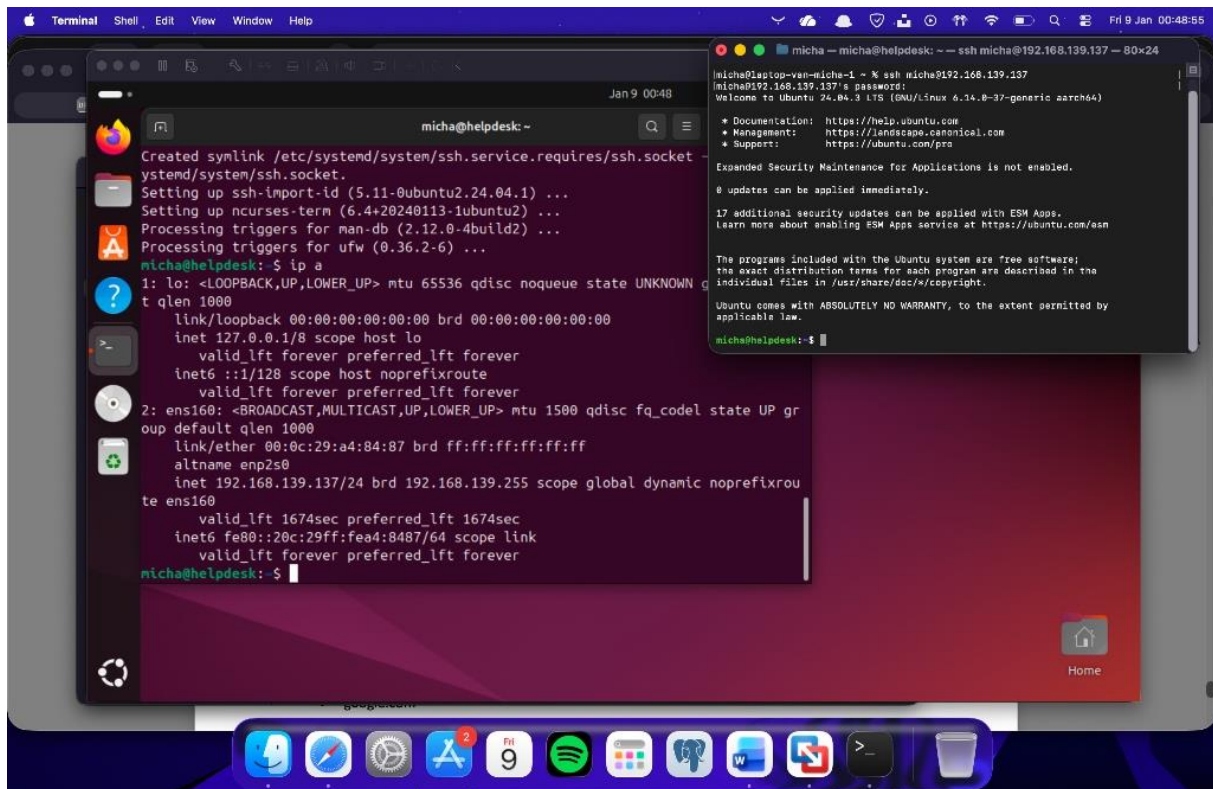
Student number: 587910

## Assignment 6.1: Working from home

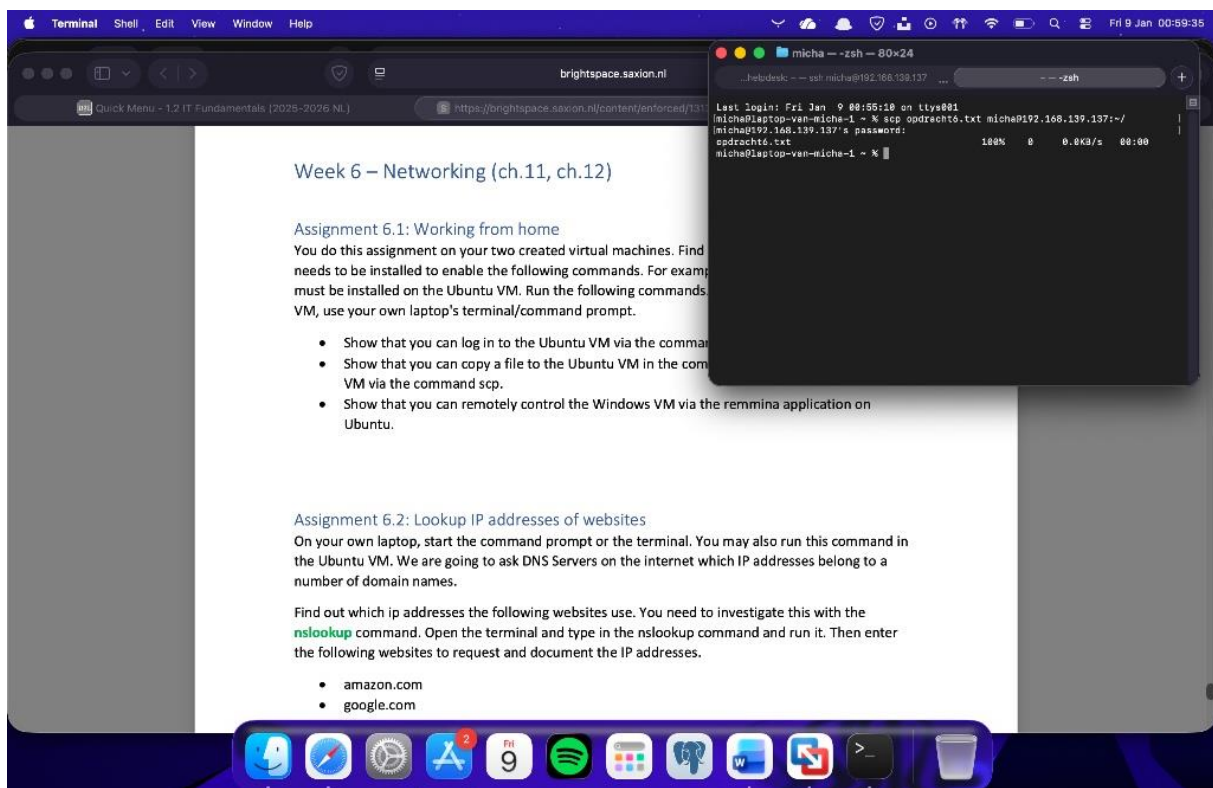
Screenshot installation openssh-server:

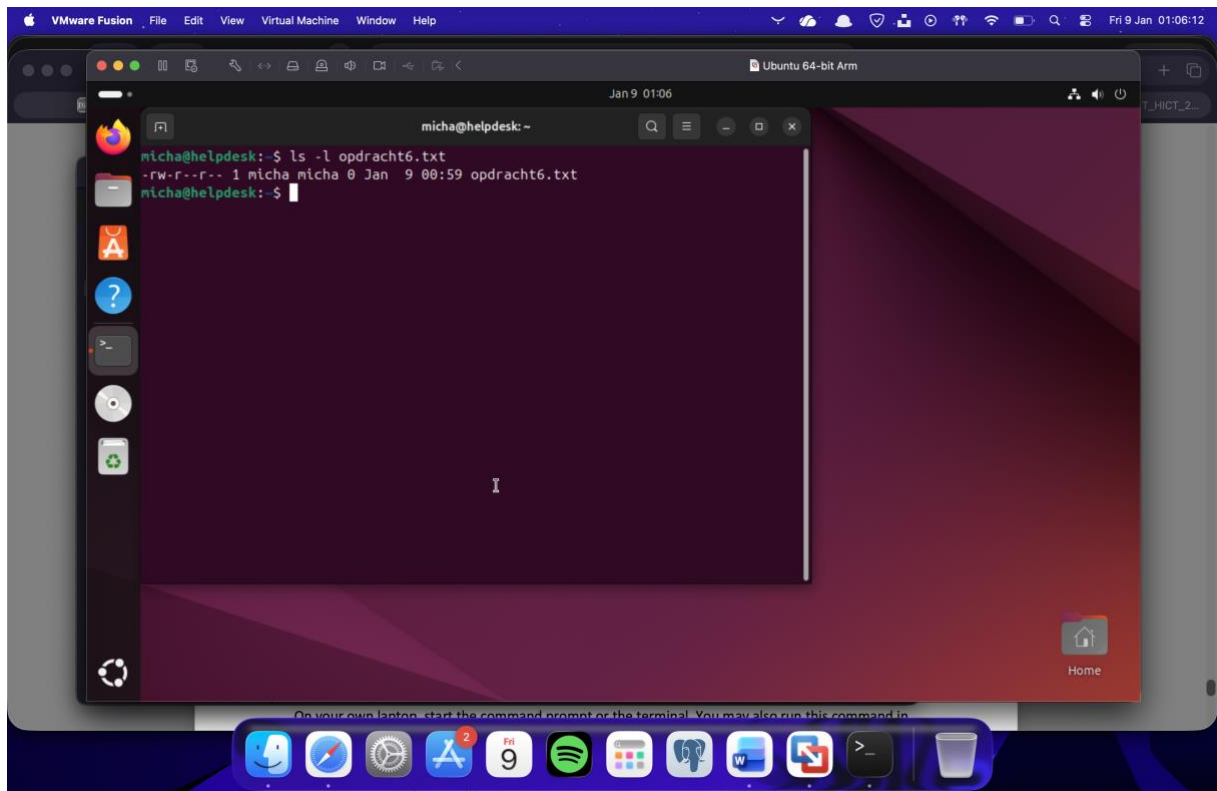


Screenshot successful SSH command execution:

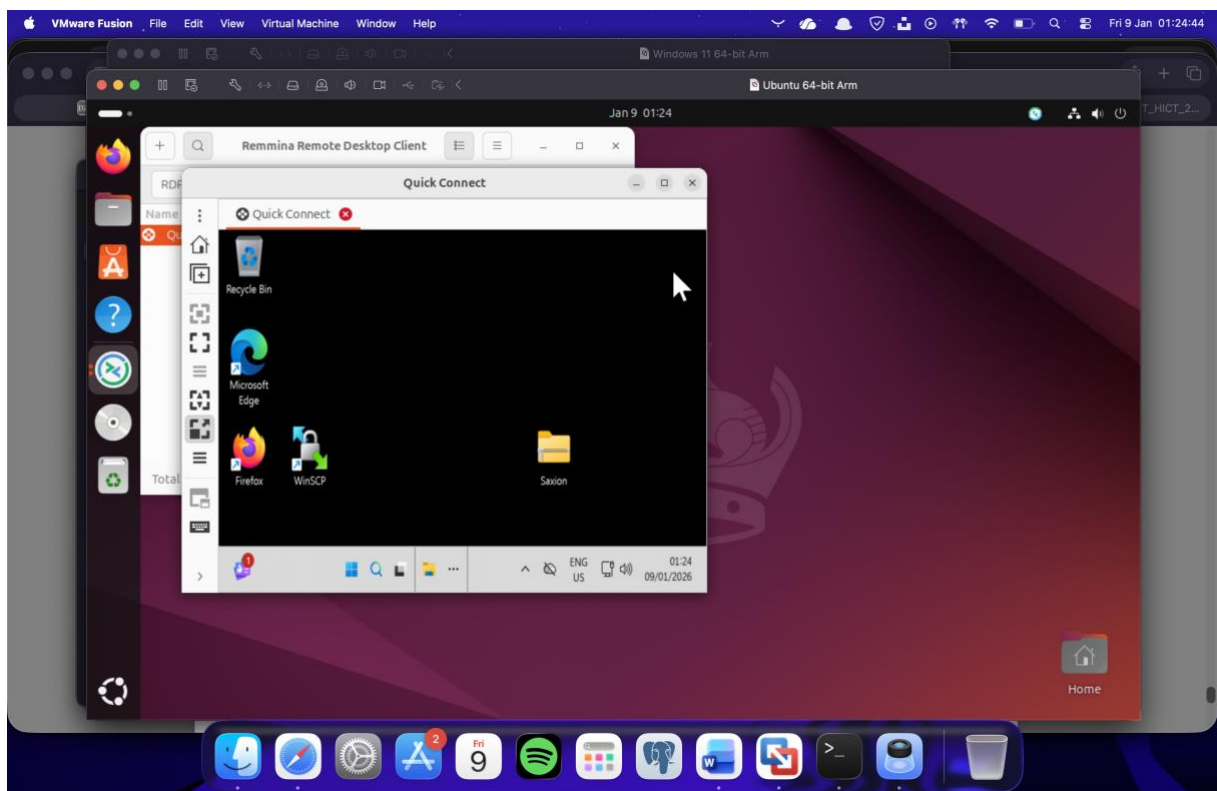


Screenshot successful execution SCP command:



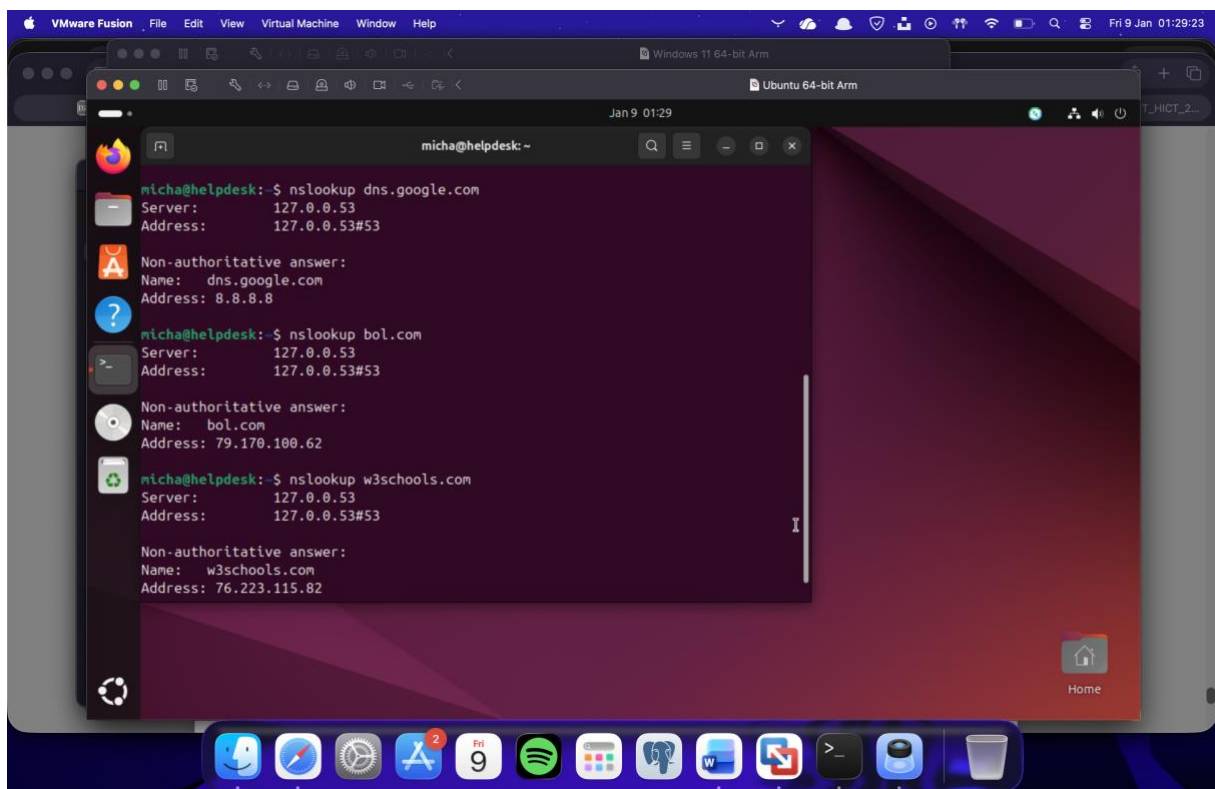
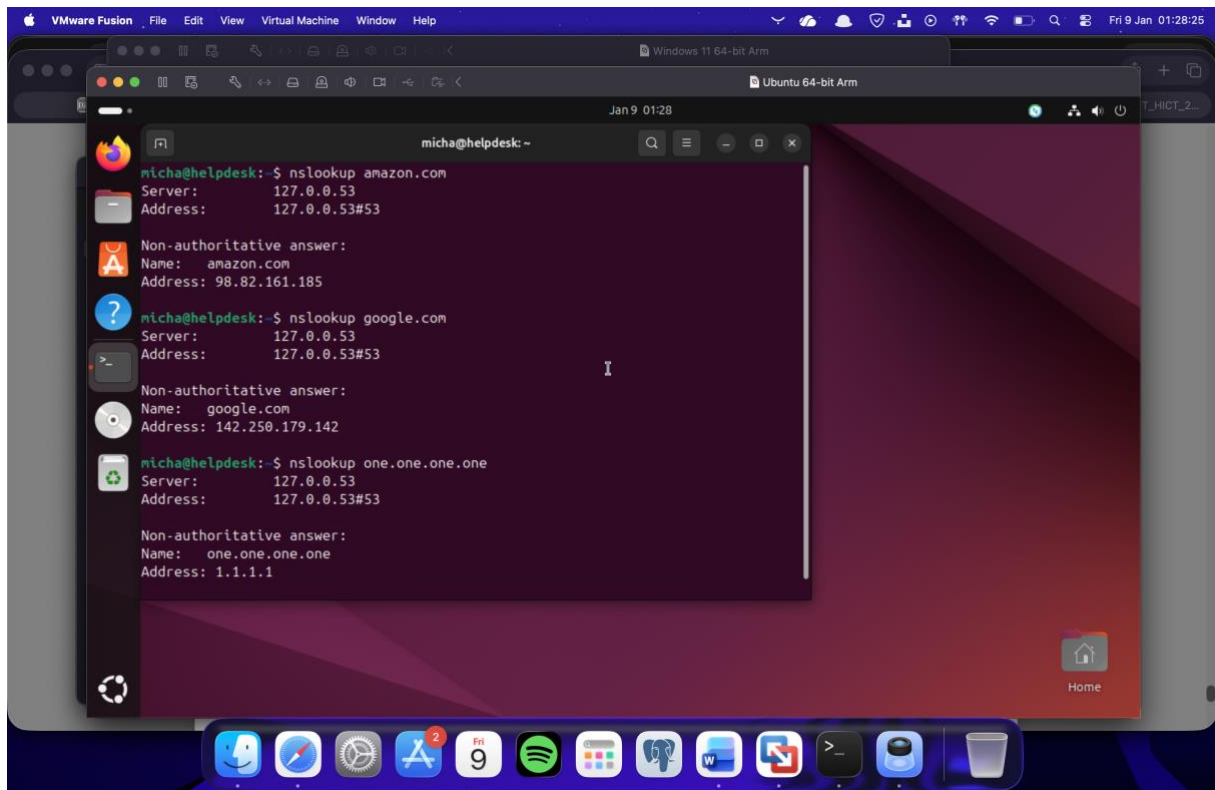


Screenshot remmina:

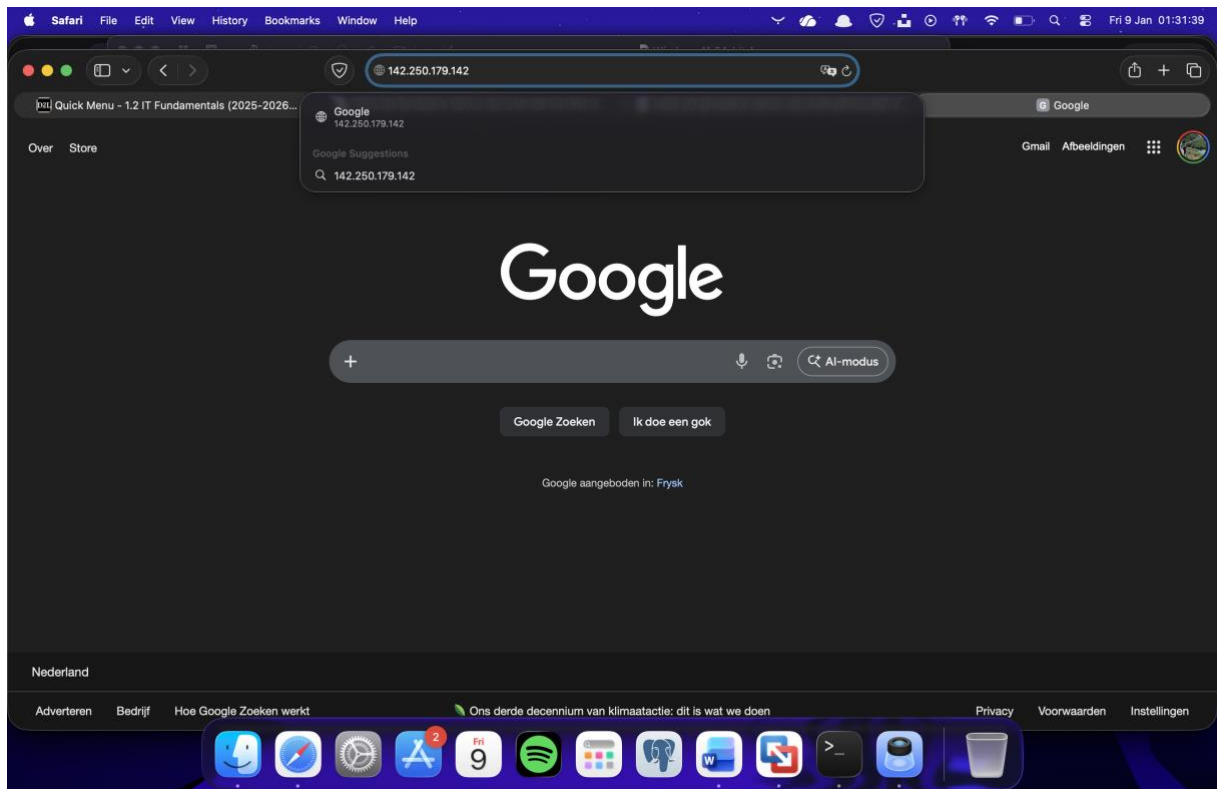


## Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:



Screenshot website visit via IP address:



### Assignment 6.3: subnetting

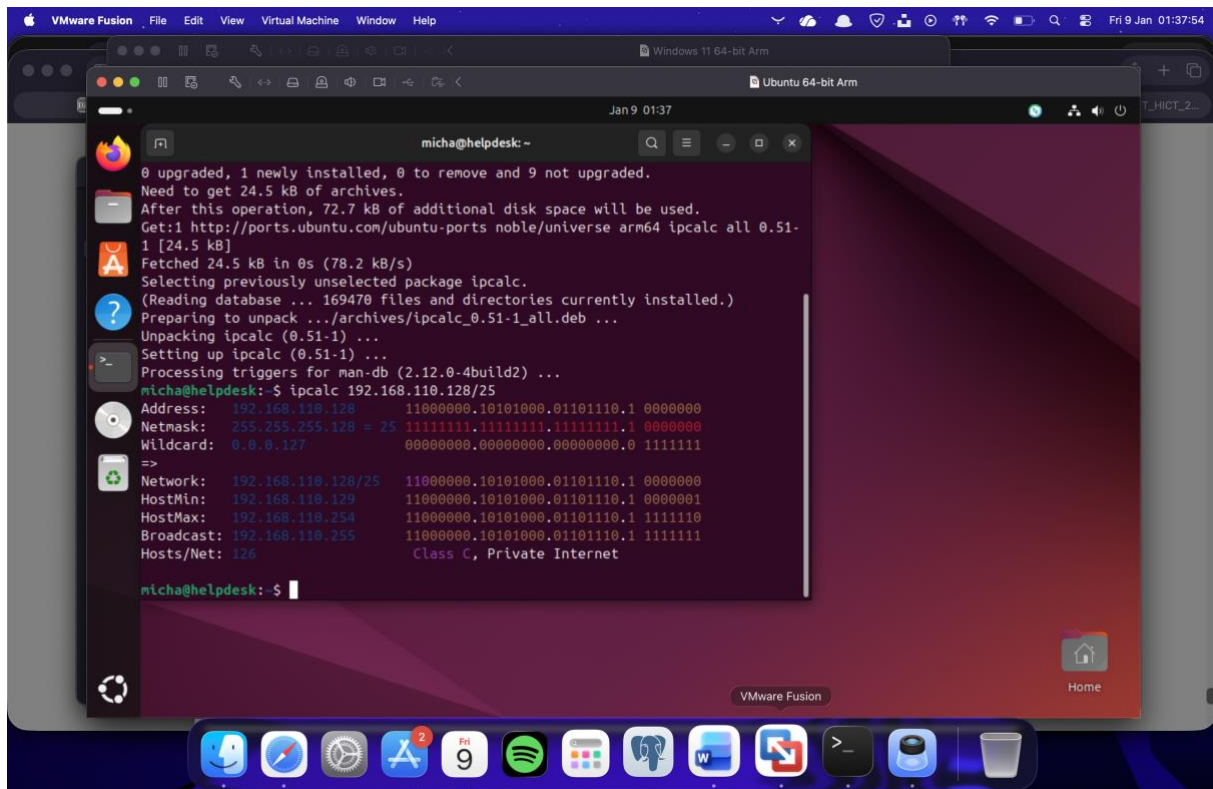
How many IP addresses are in this network configuration 192.168.110.128/25?

Een volledig IPv4-adres heeft 32 bits. Er zijn nog  $32 - 25 = 7$  bits over dus dat is  $2$  tot de macht  $7 = 128$  ip adressen.

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

De bruikbare range is van 129 t/m 254, want 128 is altijd het netwerk id en 255 is de broadcast

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

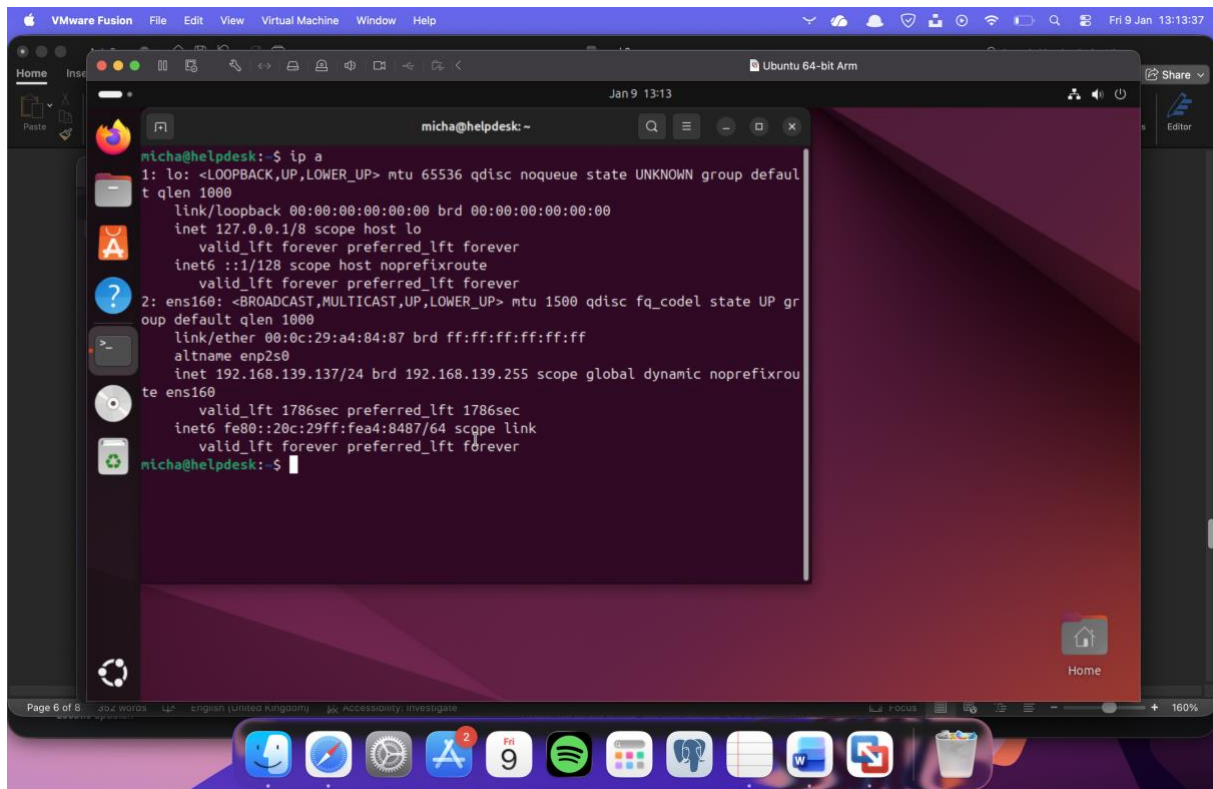


Explain the above calculation in your own words.

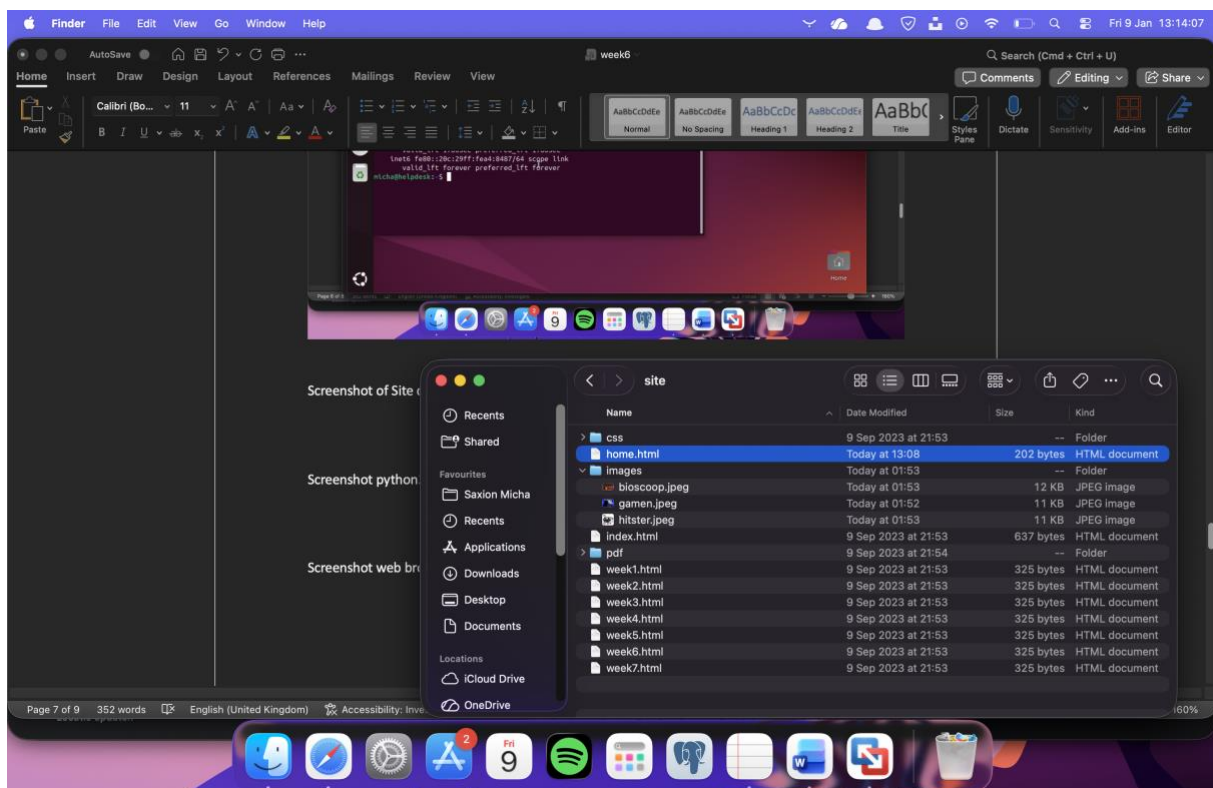
Een /25 netwerkmasker betekent dat de eerste 25 bits van het IP-adres gereserveerd zijn voor het netwerkgedeelte. Hierdoor blijven er 7 bits over voor de hosts. Met 7 bits kun je 128 unieke combinaties maken. Omdat het eerste adres het netwerk zelf is en het laatste adres het broadcast-adres is, blijven er exact 126 adressen over voor apparaten zoals computers.

#### Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

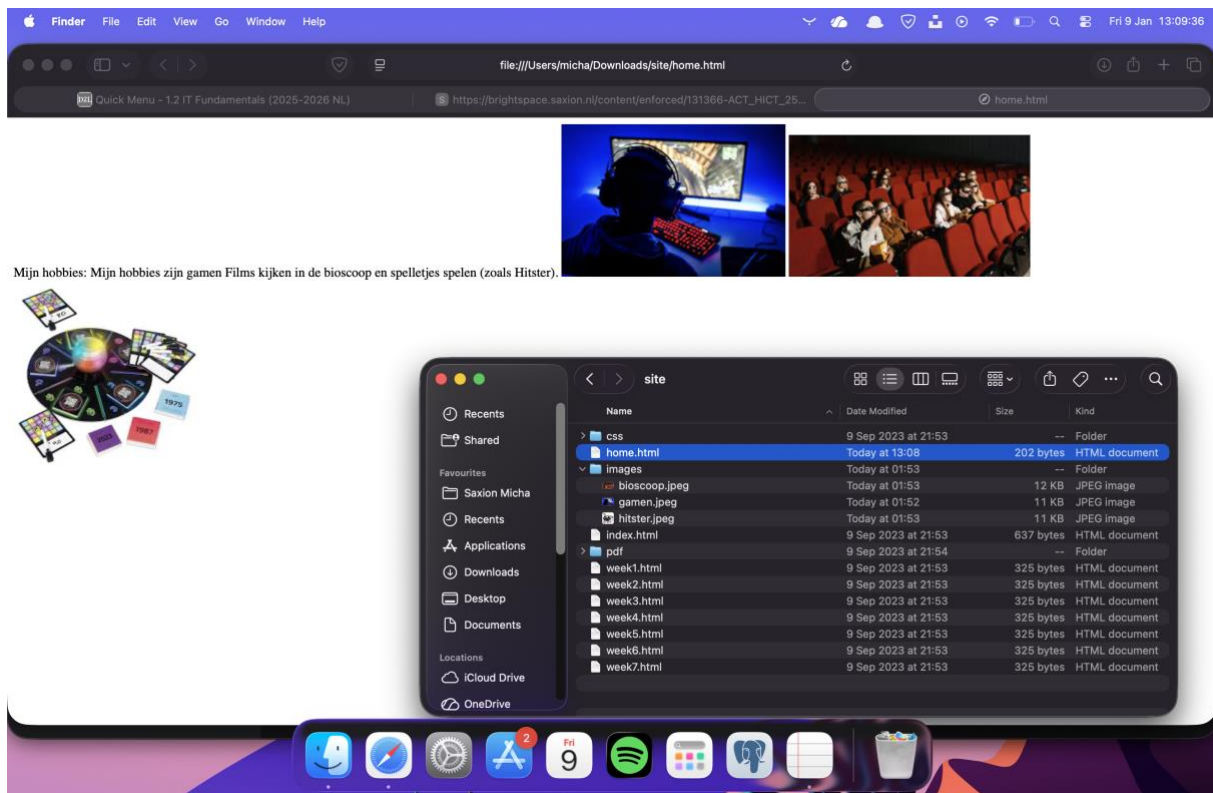


Screenshot of Site directory contents:



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 nl.saxion.app.SaxionApp;

public class Main implements Runnable {

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

    public void run() {
        SaxionApp.printLine("1. Is number odd?");
        SaxionApp.printLine("2. Is number a power of 2?");
        SaxionApp.printLine("3. Two's complement of number?");
        SaxionApp.printLine("4. Network segment");
        SaxionApp.print("\nSelect an option: ");

        int choice = SaxionApp.readInt();

        if (choice == 1) {
            SaxionApp.print("Enter a number: ");
            int num = SaxionApp.readInt();
            if ((num & 1) != 0) {
                SaxionApp.printLine(num + " is ODD.");
            } else {
                SaxionApp.printLine(num + " is EVEN.");
            }
        }

        } else if (choice == 2) {
            SaxionApp.print("Enter a number: ");
            int num = SaxionApp.readInt();
```

```

    if (num > 0 && (num & (num - 1)) == 0) {
        SaxionApp.println(num + " is a power of 2!");
    } else {
        SaxionApp.println(num + " is NOT a power of 2.");
    }

} else if (choice == 3) {
    SaxionApp.print("Enter a number: ");
    int num = SaxionApp.readInt();
    int result = ~num + 1;
    SaxionApp.println("Complement: " + result);

} else if (choice == 4) {
    SaxionApp.print("Enter IP: ");
    String ip = SaxionApp.readString();
    SaxionApp.print("Enter Subnet: ");
    String mask = SaxionApp.readString();

    String[] ipBits = ip.split("\\.");
    String[] maskBits = mask.split("\\.");

    String result = "";

    for (int i = 0; i < 4; i++) {
        int part1 = Integer.parseInt(ipBits[i]);
        int part2 = Integer.parseInt(maskBits[i]);

        int calculation = part1 & part2;

        result = result + calculation;

        if (i < 3) {

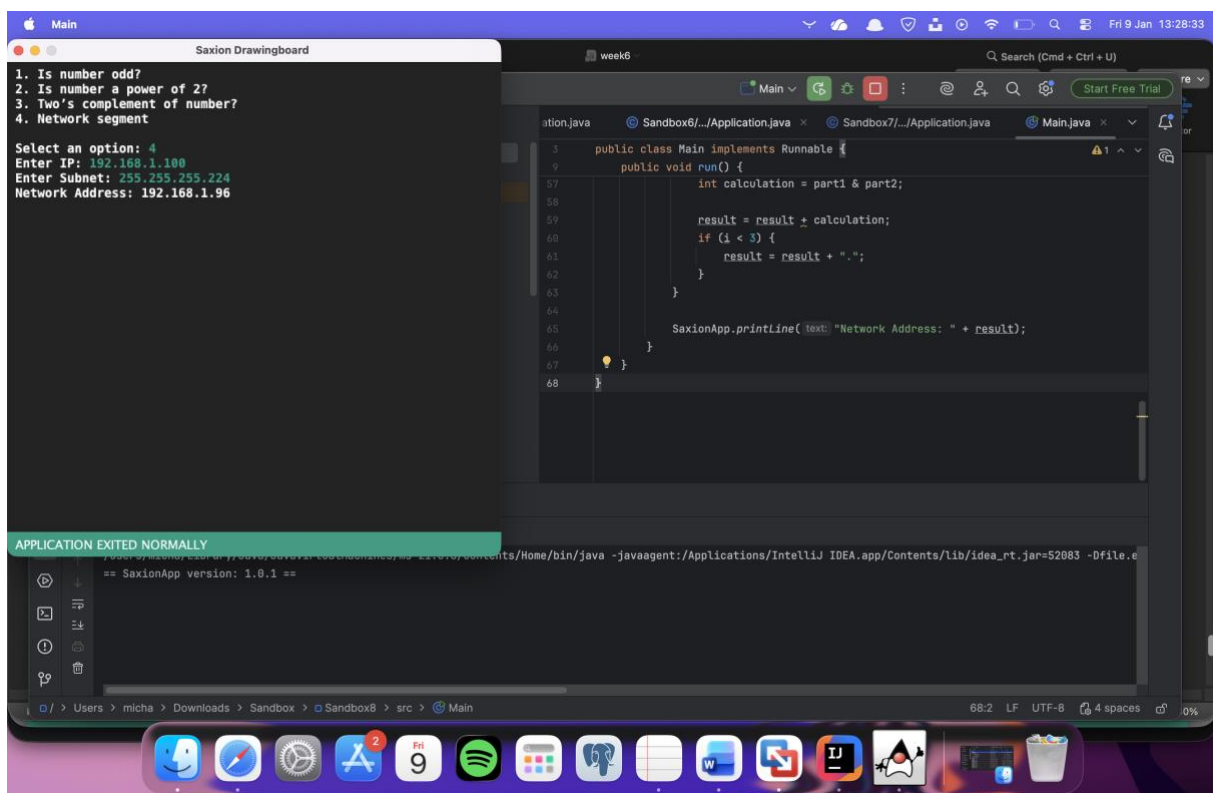
```

```

        result = result + ".";
    }
}

SaxionApp.println("Network Address: " + result);
}
}
}
}
}

```



Ready? Save this file and export it as a pdf file with the name: [week6.pdf](#)