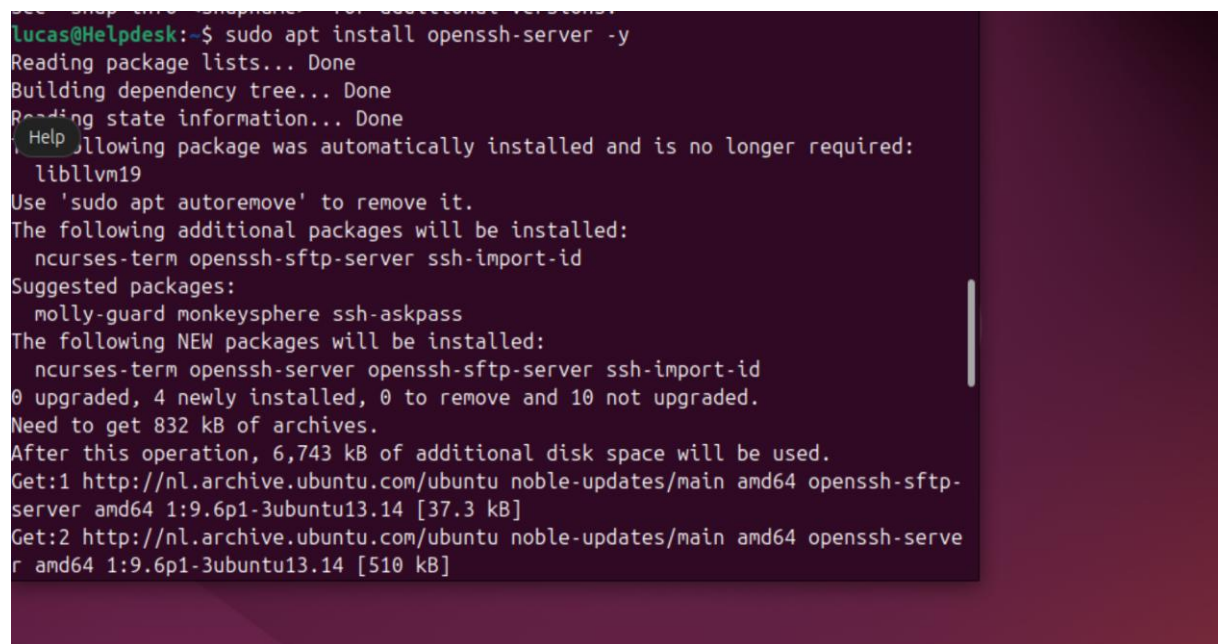


Template Week 6 – Networking

Student number:

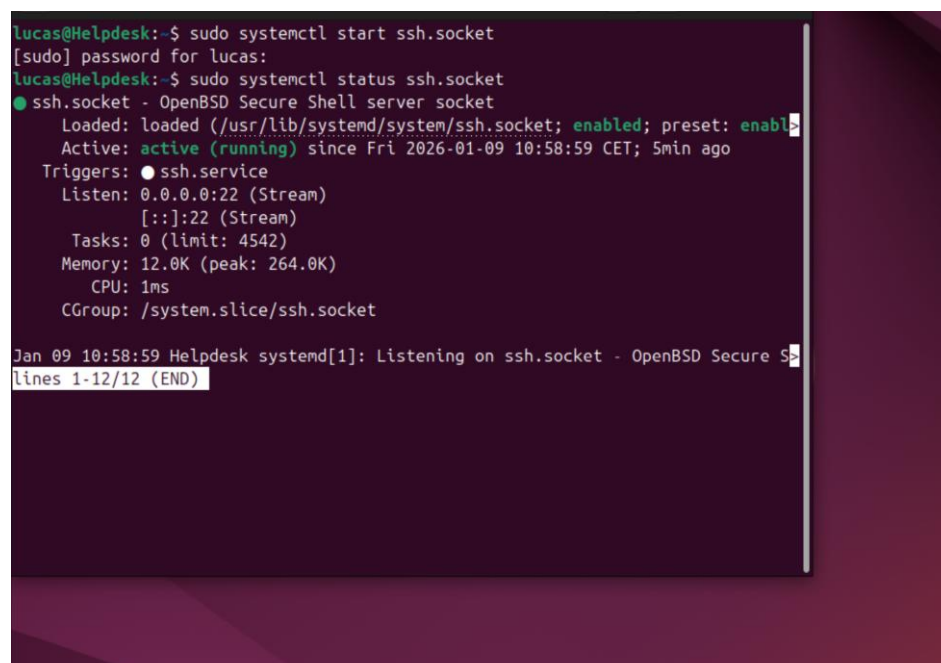
Assignment 6.1: Working from home

Screenshot installation openssh-server:

A terminal window showing the installation of openssh-server. The user runs 'sudo apt install openssh-server -y'. The output shows the package lists, dependency tree, and state information. It indicates that libllvm19 was automatically installed and is no longer required. It also lists additional packages to be installed: ncurses-term, openssh-sftp-server, and ssh-import-id. Suggested packages include molly-guard, monkeysphere, and ssh-askpass. The terminal shows the disk space requirements and the download progress for the packages. The user is prompted for a password, which is masked with asterisks.

```
Lucas@Helpdesk:~$ sudo apt install openssh-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere ssh-askpass
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 upgraded, 4 newly installed, 0 to remove and 10 not upgraded.
Need to get 832 kB of archives.
After this operation, 6,743 kB of additional disk space will be used.
Get:1 http://nl.archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-sftp-
server amd64 1:9.6p1-3ubuntu13.14 [37.3 kB]
Get:2 http://nl.archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-serve
r amd64 1:9.6p1-3ubuntu13.14 [510 kB]
```

Screenshot successful SSH command execution:

A terminal window showing the successful execution of SSH commands. The user runs 'sudo systemctl start ssh.socket' and 'sudo systemctl status ssh.socket'. The output shows that the service is active and running. The terminal also displays the system logs for the SSH service, indicating that it is listening on the specified port.

```
Lucas@Helpdesk:~$ sudo systemctl start ssh.socket
[sudo] password for Lucas:
Lucas@Helpdesk:~$ sudo systemctl status ssh.socket
● ssh.socket - OpenBSD Secure Shell server socket
   Loaded: loaded (/usr/lib/systemd/system/ssh.socket; enabled; preset: enabl>
   Active: active (running) since Fri 2026-01-09 10:58:59 CET; 5min ago
   Triggers: ● ssh.service
   Listen: 0.0.0.0:22 (Stream)
           [::]:22 (Stream)
   Tasks: 0 (limit: 4542)
   Memory: 12.0K (peak: 264.0K)
   CPU: 1ms
   CGroup: /system.slice/ssh.socket

Jan 09 10:58:59 Helpdesk systemd[1]: Listening on ssh.socket - OpenBSD Secure S
lines 1-12/12 (END)
```

```

ED25519 key fingerprint is SHA256:Y/vgBh0mrhFjJDyMqs2tRs+YrQCC5C/Yuun/Xp44xXE.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yeas
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.139.132' (ED25519) to the list of known hosts.
lucas@192.168.139.132'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.

1 update can be applied immediately.
1 of these updates is a standard security update.
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.

lucas@Helpdesk:~$ |

```

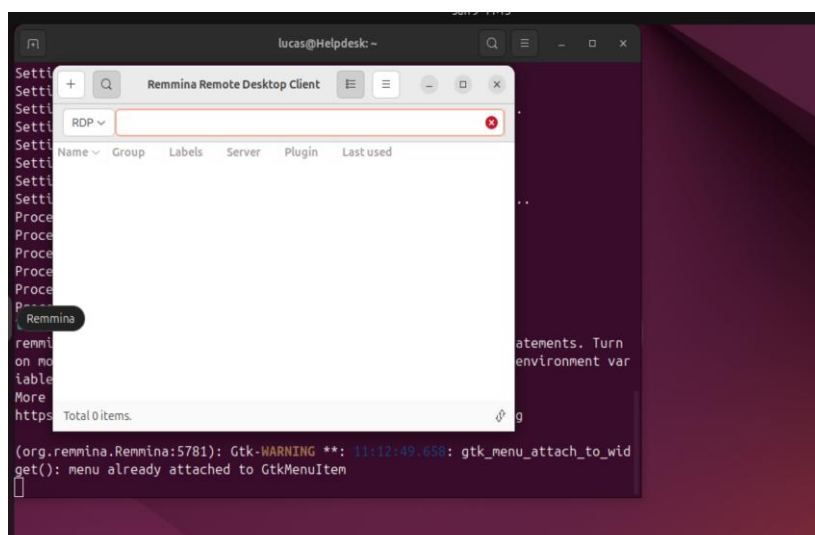
Screenshot successful execution SCP command:

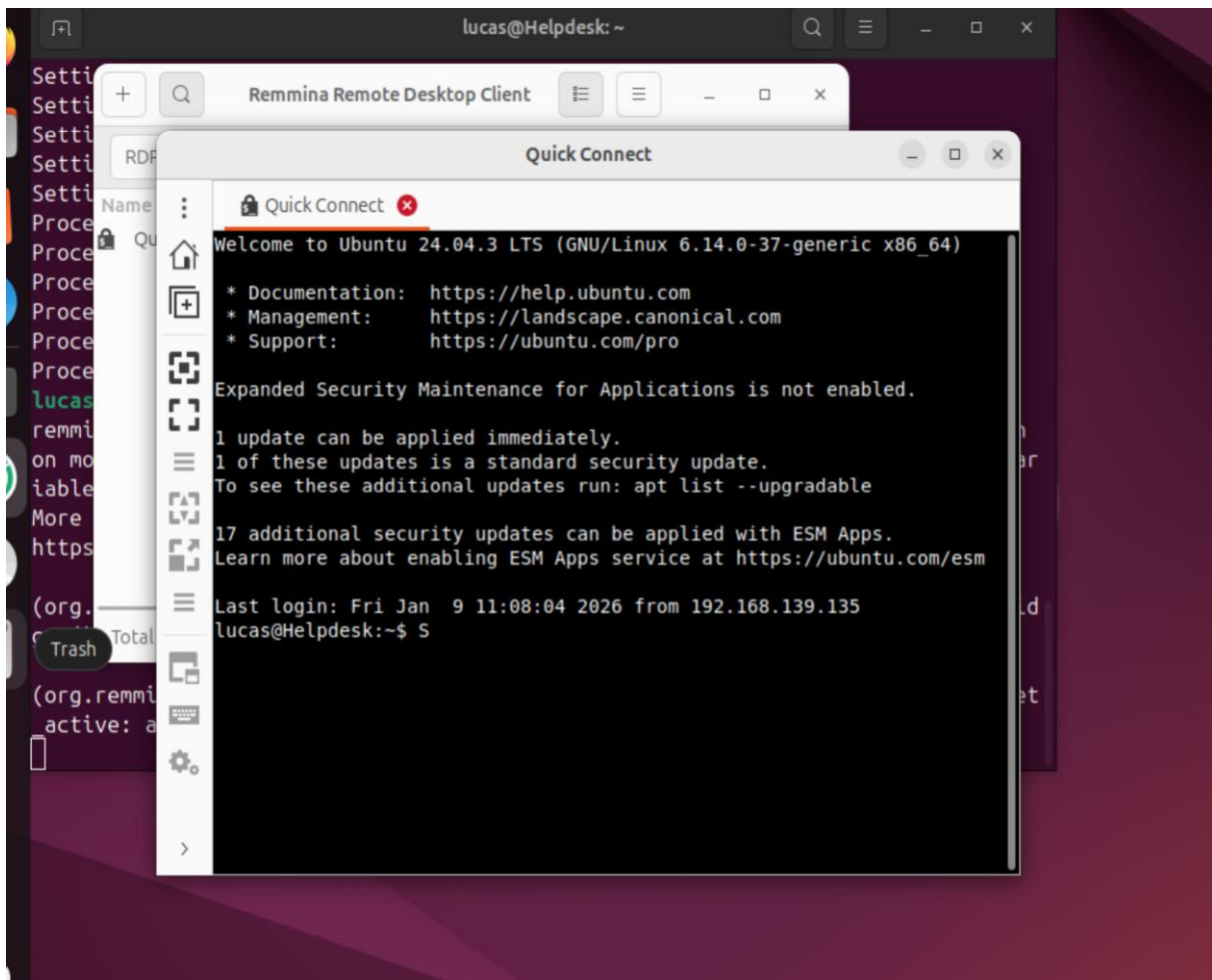
```

PS C:\Users\Lucas> scp C:\Users\Lucas\Desktop\test.txt lucas@192.168.139.132:/home/Lucas/
lucas@192.168.139.132's password:
test.txt                                     100%   4    1.0KB/s   00:00
PS C:\Users\Lucas>

```

Screenshot remmina:





Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:

```

lucas@Helpdesk:~$ nslookup www.youtube.com
Server:      127.0.0.53
Address:     127.0.0.53#53

```

```

lucas@Helpdesk:~$ nslookup www.microsoft.com
Server:      127.0.0.53
Address:     127.0.0.53#53

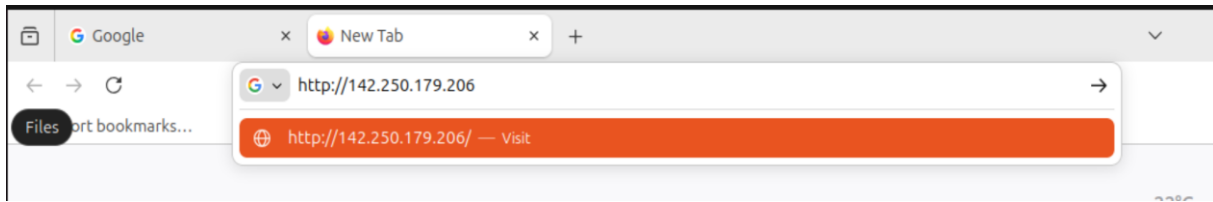
```

```

lucas@Helpdesk:~$ nslookup www.saxion.nl
nslookup www.microsoft.com
nslookup www.youtube.comServer:      127.0.0.53
Address:     127.0.0.53#53

```

Screenshot website visit via IP address:



Assignment 6.3: subnetting

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

Het subnet gebruikt een prefix.

Een IPv4-adres heeft 32 bits. Bij worden 25 bits gebruikt voor het netwerkdeel en blijven er 7 bits over voor hosts.

Het aantal mogelijke IP-adressen in dit subnet is daarom:

IP-adressen in totaal.

Dus: er zitten 128 IP-adressen in dit netwerk.

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

In elk subnet zijn:

- het eerste adres het netwerkadres (niet uitdeelbaar)
- het laatste adres het broadcastadres (niet uitdeelbaar)

Voor geldt:

- Netwerkadres: 192.168.110.128
- Broadcastadres: 192.168.110.255

De bruikbare hostadressen zitten daartussen:

- Eerste bruikbare IP: 192.168.110.129
- Laatste bruikbare IP: 192.168.110.254

Dus de bruikbare IP-range voor computers is:

192.168.110.129 – 192.168.110.254

In totaal zijn dit 126 bruikbare IP-adressen.

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

```
Lucas@Helpdesk:~$ 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

Lucas@Helpdesk:~$
```

Dit bevestigt dat:

- er 126 bruikbare hosts zijn
- de usable range loopt van 192.168.110.129 t/m 192.168.110.254

Explain the above calculation in your own words.

Bij subnetting wordt een IP-adres in twee delen gesplitst: een netwerkdeel en een hostdeel. De geeft aan hoeveel bits voor het netwerk worden gebruikt. Omdat er 32 bits in totaal zijn en 25 voor het netwerk, blijven er 7 bits over voor hosts. Met 7 bits kun je combinaties maken, dus 128 IP-adressen in het subnet.

Van deze 128 adressen zijn er 2 speciaal:

- het eerste adres is het netwerkadres (om het subnet zelf aan te duiden)
- het laatste adres is het broadcastadres (om een bericht naar alle hosts in het subnet te sturen)

Daarom blijven er $128 - 2 = 126$ adressen over die je echt aan computers kunt uitdelen. Die lopen van 192.168.110.129 tot en met 192.168.110.254. Met een tool zoals kun je dit automatisch laten uitrekenen en controleren.

Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

```
Lucas@Helpdesk:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default
    link/ether 00:0c:29:cd:ac:13 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.139.132/24 brd 192.168.139.255 scope global dynamic noprefixroute
        valid_lft 1032sec preferred_lft 1032sec
    inet6 fe80::20c:29ff:fedc:ac13/64 scope link
        valid_lft forever preferred_lft forever
```

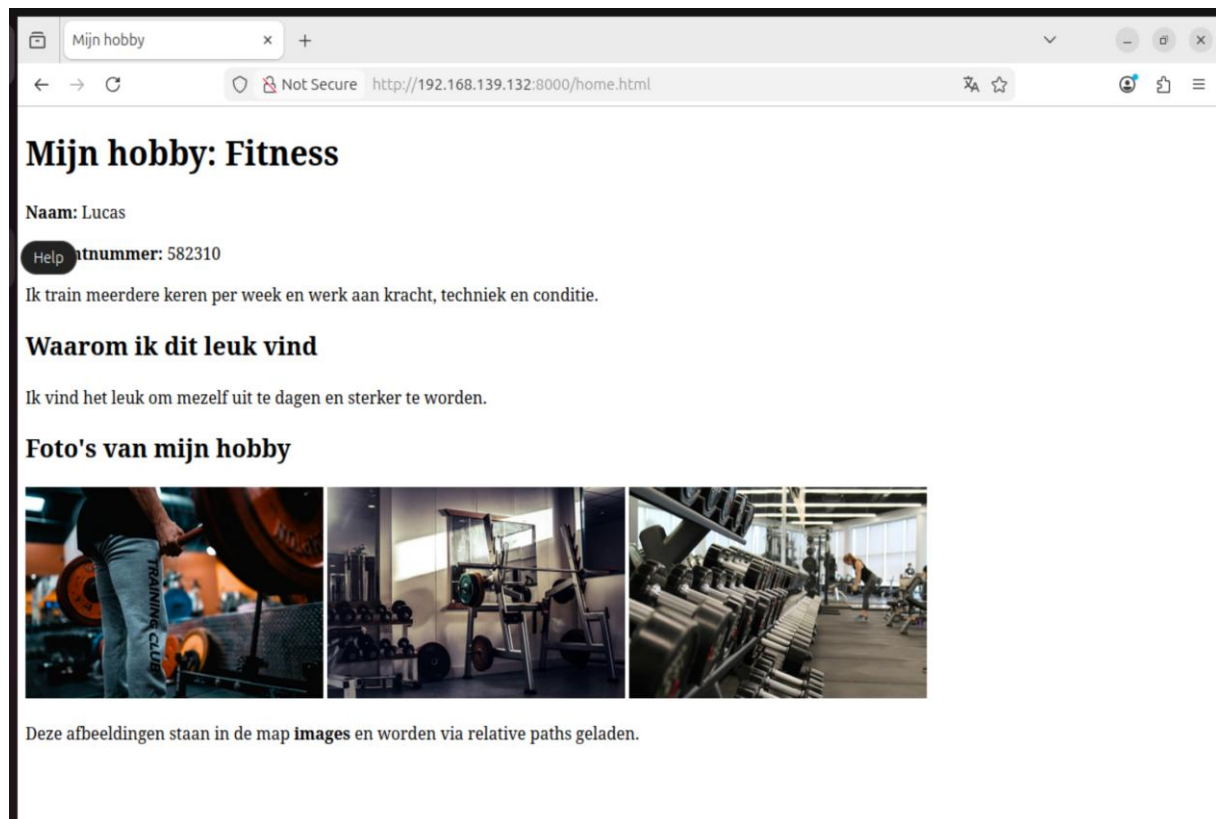

Screenshot of Site directory contents:

```
lucas@Helpdesk:~/site$ ls -l
total 48
drwxrwxr-x 2 lucas lucas 4096 Sep  9 2023 css
-rw-rw-r-- 1 lucas lucas  619 Jan  9 11:51 home.html
drwxrwxr-x 2 lucas lucas 4096 Jan  9 11:55 images
-rw-rw-r-- 1 lucas lucas  637 Sep  9 2023 index.html
drwxrwxr-x 2 lucas lucas 4096 Sep  9 2023 pdf
-rw-rw-r-- 1 lucas lucas  325 Sep  9 2023 week1.html
-rw-rw-r-- 1 lucas lucas  325 Sep  9 2023 week2.html
-rw-rw-r-- 1 lucas lucas  325 Sep  9 2023 week3.html
-rw-rw-r-- 1 lucas lucas  325 Sep  9 2023 week4.html
-rw-rw-r-- 1 lucas lucas  325 Sep  9 2023 week5.html
-rw-rw-r-- 1 lucas lucas  325 Sep  9 2023 week6.html
-rw-rw-r-- 1 lucas lucas  325 Sep  9 2023 week7.html
```

Screenshot python3 webserver command:

```
lucas@Helpdesk:~/site$ cd ~/site
lucas@Helpdesk:~/site$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

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 {
```

```
    public static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in);
```

```
        System.out.print("Enter subnet: ");
```

```
        String subnet = scanner.nextLine();
```

```
        System.out.print("Enter amount: ");
```

```
        int amount = scanner.nextInt();
```

```
        int[] subnetInt = parseAddress(subnet);
```

```
        int[] subnetMask = parseAddress(subnet.split("/")[1]);
```

```
        int[] newSubnet = new int[4];
```

```
        int start = 0;
```

```
        for (int i = 0; i < amount; i++) {
```

```
            newSubnet[3] = i;
```

```
            System.out.printf("Subnet #%d: %s\n", i + 1, formatAddress(newSubnet));
```

```

    }

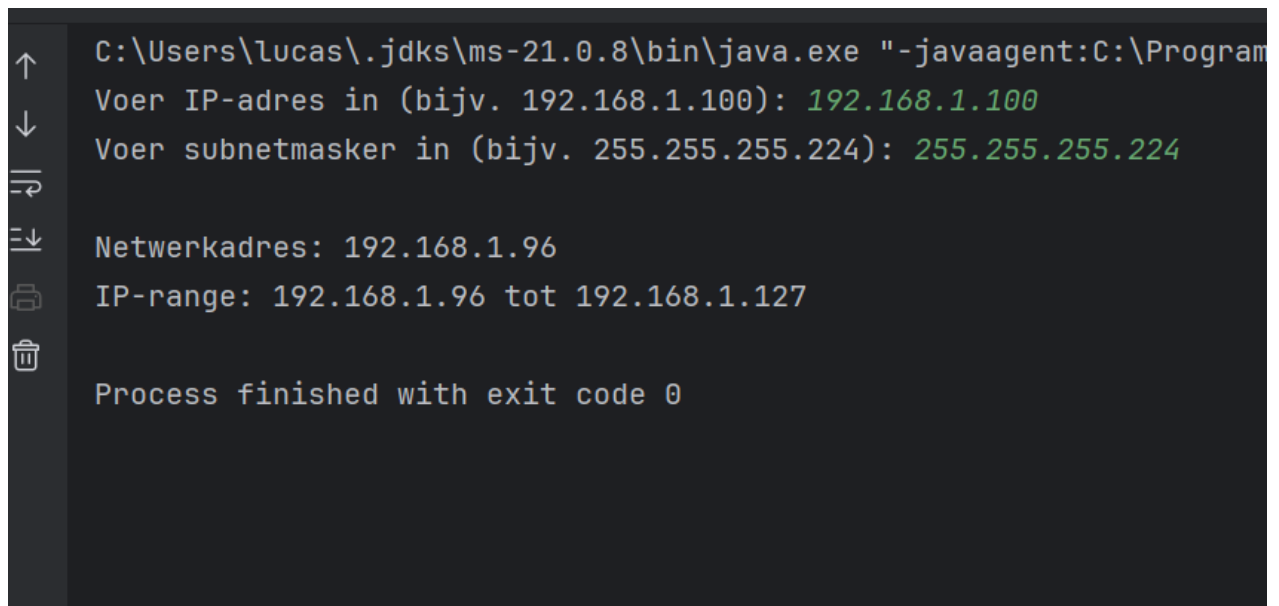
    int hosts = 256 - subnetInt[3];
    int end = start + hosts - 1;
    start = subnetInt[3];

    System.out.printf("Range: %s - %s\n", subnetInt[0] + "." + subnetInt[1] + "." + subnetInt[2] + "." +
start,
        subnetInt[0] + "." + subnetInt[1] + "." + subnetInt[2] + "." + end);
    }

    public static int[] parseAddress(String address) {
        String[] parts = address.split("\\.");
        int[] result = new int[4];
        for (int i = 0; i < 4; i++) {
            result[i] = Integer.parseInt(parts[i]);
        }
        return result;
    }

    public static String formatAddress(int[] address) {
        return address[0] + "." + address[1] + "." + address[2] + "." + address[3];
    }
}

```

```
C:\Users\lucas\.jdk\ms-21.0.8\bin\java.exe "-javaagent:C:\Program...
Voer IP-adres in (bijv. 192.168.1.100): 192.168.1.100
Voer subnetmasker in (bijv. 255.255.255.224): 255.255.255.224

Netwerkadres: 192.168.1.96
IP-range: 192.168.1.96 tot 192.168.1.127

Process finished with exit code 0
```

Voor Assignment 6.5 heb ik mijn Java-applicatie uitgebreid met een functie om netwerksegmenten te berekenen. De gebruiker voert een IP-adres en subnetmasker in. De applicatie zet deze om naar integers en berekent het netwerkadres met de bitwise AND-operator.

Voorbeeld:

- IP: 192.168.1.100
- Subnet: 255.255.255.224 (/27)
- Netwerkadres: 192.168.1.96
- IP-range: 192.168.1.96 tot 192.168.1.127

De berekening gebeurt per octet met:

De applicatie toont het netwerkadres en de volledige IP-range van het subnet.

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