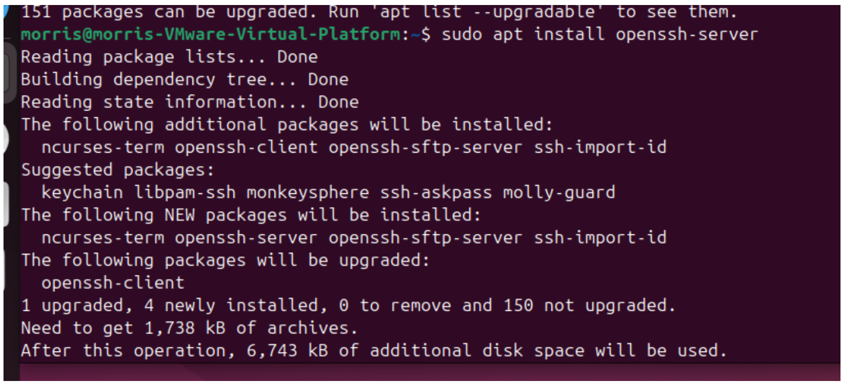
Template Week 6 – Networking

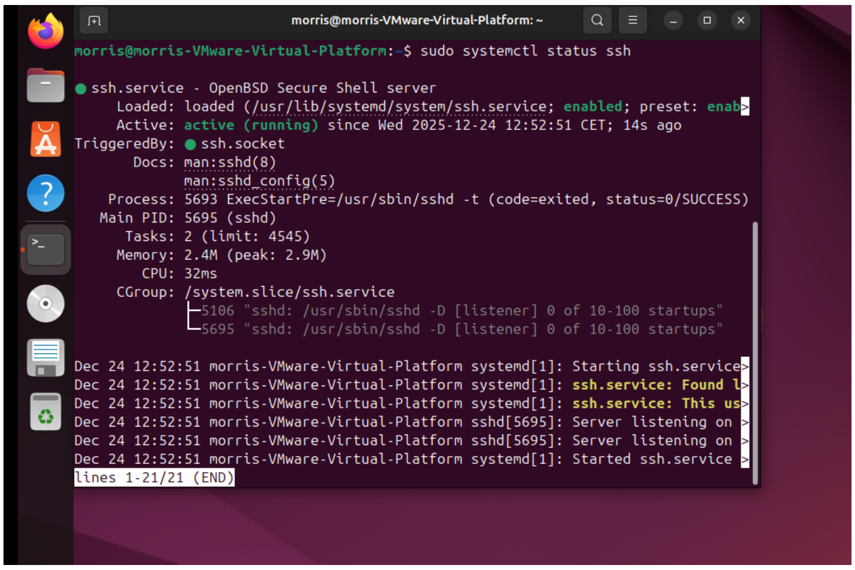
Student number: 589948

Assignment 6.1: Working from home

Screenshot installation openssh-server:

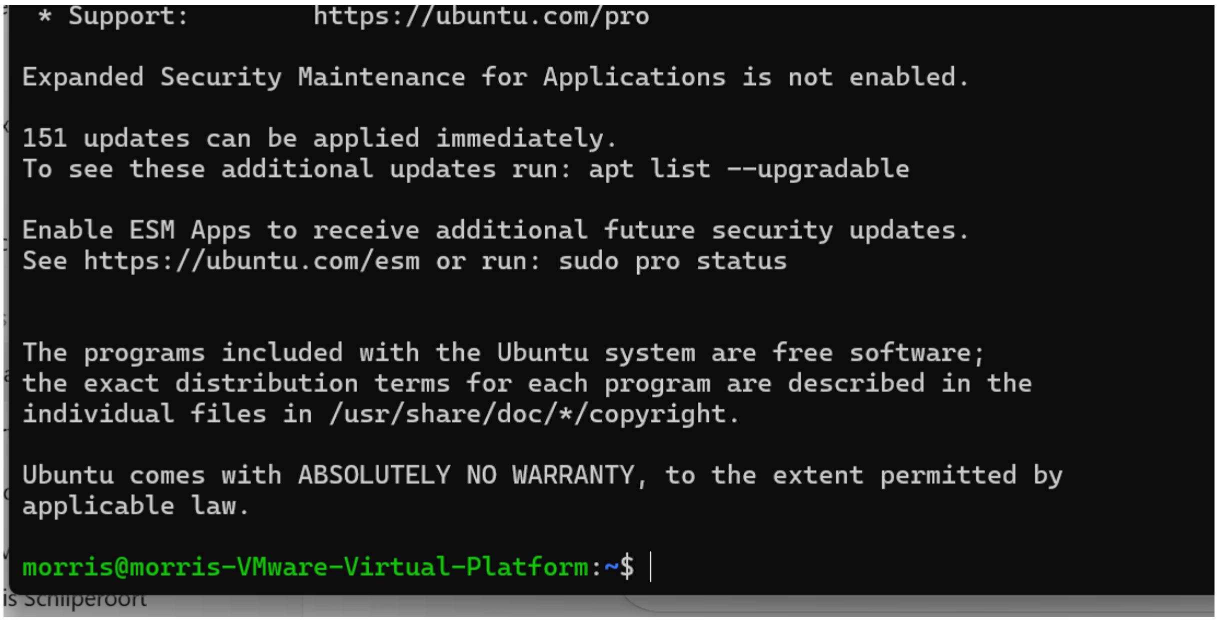


```
151 packages can be upgraded. Run 'apt list --upgradable' to see them.
morris@morris-VMware-Virtual-Platform:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ncurses-term 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 150 not upgraded.
Need to get 1,738 kB of archives.
After this operation, 6,743 kB of additional disk space will be used.
```



```
morris@morris-VMware-Virtual-Platform:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enab>
     Active: active (running) since Wed 2025-12-24 12:52:51 CET; 14s ago
TriggeredBy: ● ssh.socket
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 5693 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 5695 (sshd)
      Tasks: 2 (limit: 4545)
     Memory: 2.4M (peak: 2.9M)
        CPU: 32ms
     CGroup: /system.slice/ssh.service
             ├─5106 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
             └─5695 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Dec 24 12:52:51 morris-VMware-Virtual-Platform systemd[1]: Starting ssh.service>
Dec 24 12:52:51 morris-VMware-Virtual-Platform systemd[1]: ssh.service: Found l>
Dec 24 12:52:51 morris-VMware-Virtual-Platform systemd[1]: ssh.service: This us>
Dec 24 12:52:51 morris-VMware-Virtual-Platform sshd[5695]: Server listening on >
Dec 24 12:52:51 morris-VMware-Virtual-Platform sshd[5695]: Server listening on >
Dec 24 12:52:51 morris-VMware-Virtual-Platform systemd[1]: Started ssh.service >
lines 1-21/21 (END)
```

Screenshot successful SSH command execution:



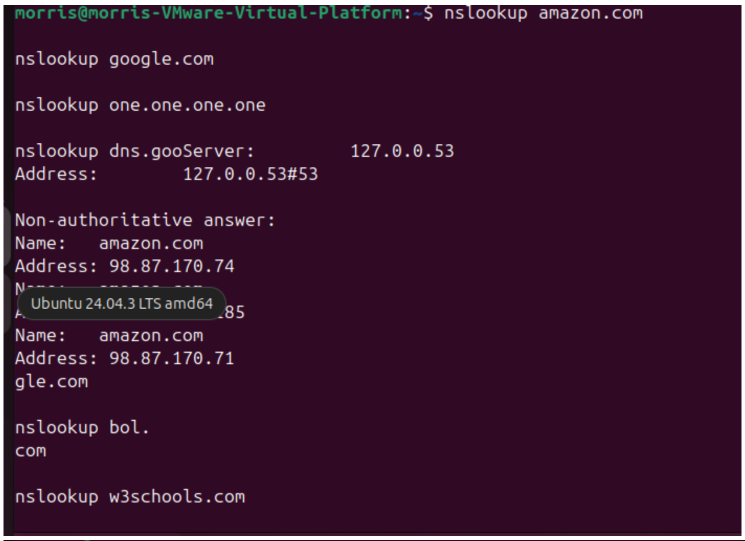
```
 * Support:        https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

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

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


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.

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

Screenshot successful execution SCP command:

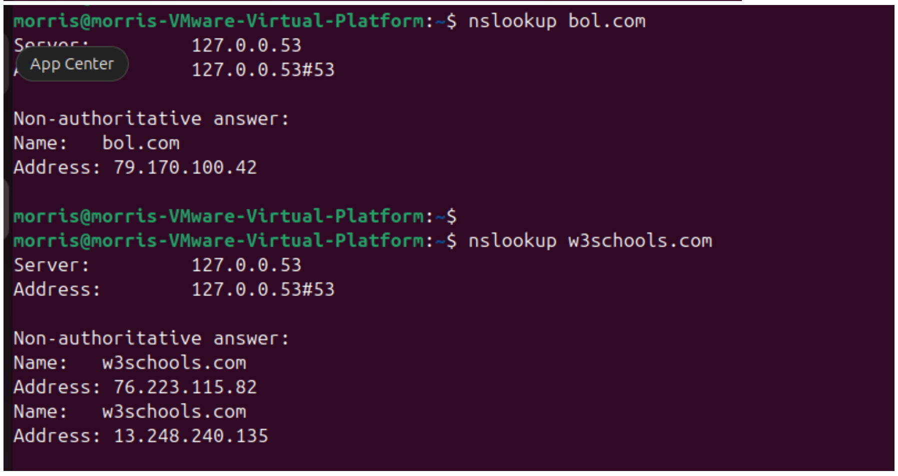
Screenshot remmina:

Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:

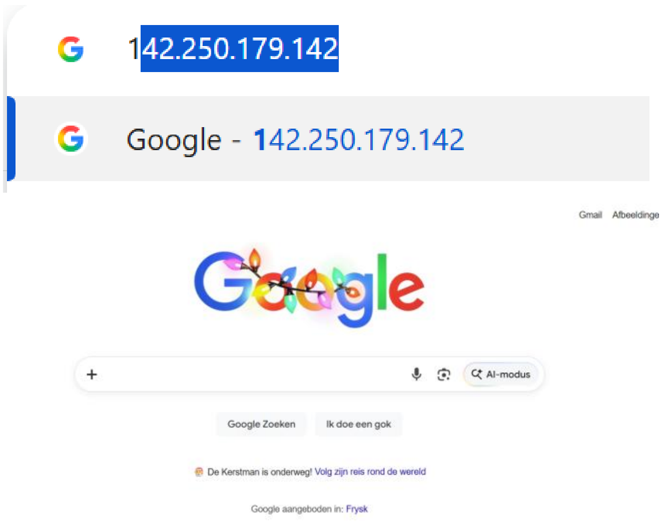


```
morris@morris-VMware-Virtual-Platform:~$ nslookup amazon.com

nslookup google.com

nslookup one.one.one.one

nslookup dns.gooServer:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   amazon.com
Address: 98.87.170.74
N
A                                   85
Name:   amazon.com
Address: 98.87.170.71
gle.com

nslookup bol.
com

nslookup w3schools.com
```



```
morris@morris-VMware-Virtual-Platform:~$ nslookup bol.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   bol.com
Address: 79.170.100.42

morris@morris-VMware-Virtual-Platform:~$
morris@morris-VMware-Virtual-Platform:~$ nslookup w3schools.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   w3schools.com
Address: 76.223.115.82
Name:   w3schools.com
Address: 13.248.240.135
```

Screenshot website visit via IP address:



Assignment 6.3: subnetting

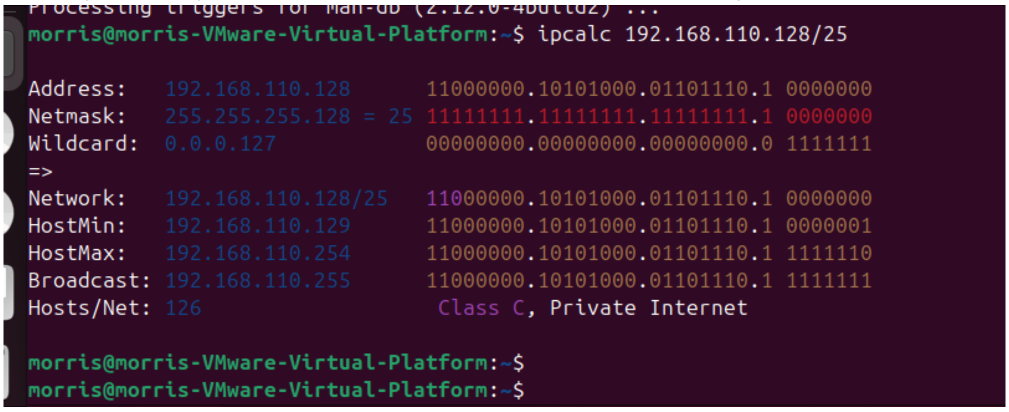
How many IP addresses are in this network configuration 192.168.110.128/25?

- /25 = 128 addressen

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

- 126 bruikbare ip’s.

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

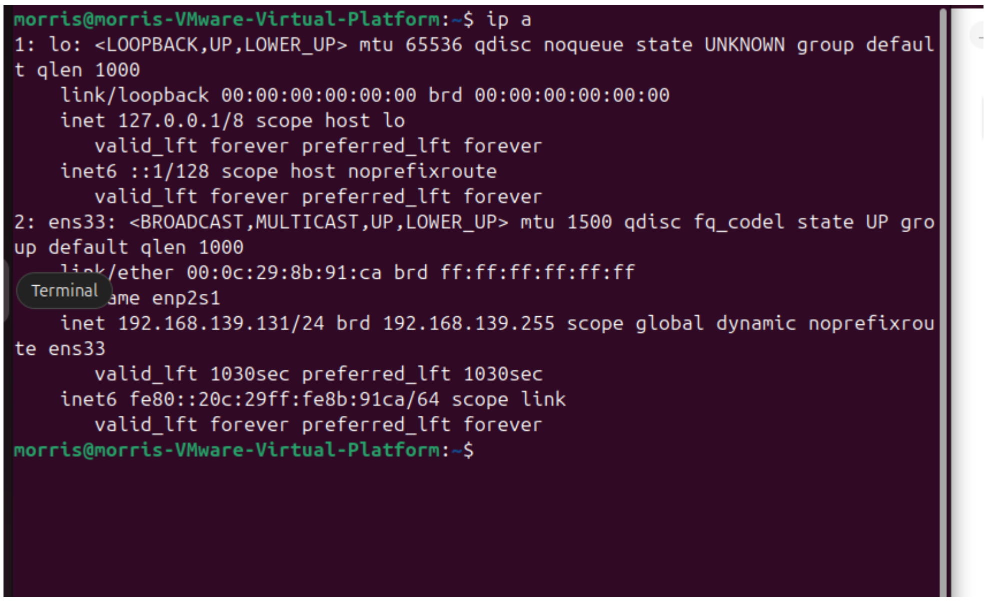


```
Processing triggers for man-db (2.12.0-4build2) ...
morris@morris-VMware-Virtual-Platform:~$ ipcalc 192.168.110.128/25

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

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

Explain the above calculation in your own words:
- Een /25 betekent dat de eerste 25 bits van het IP-adres het netwerkdeel zijn en de laatste 7 bits voor hosts zijn. Daardoor krijg je 2^(32−25)=2^7=128 adressen in dit subnet. Het netwerkadres is 192.168.110.128 en het broadcastadres is 192.168.110.255. De eerste bruikbare host is 192.168.110.129 en de laatste bruikbare host is 192.168.110.254. In totaal zijn er 126 bruikbare host-adressen omdat netwerk- en broadcastadres niet gebruikt kunnen worden

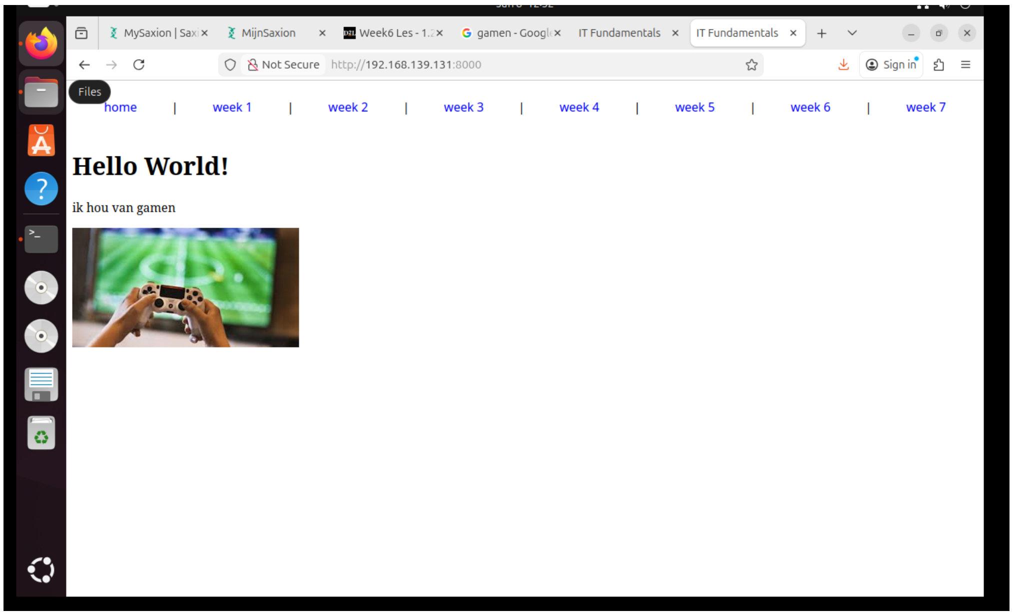
Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:



```
morris@morris-VMware-Virtual-Platform:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    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 gro
up default qlen 1000
    link/ether 00:0c:29:8b:91:ca brd ff:ff:ff:ff:ff:ff
        ame enp2s1
    inet 192.168.139.131/24 brd 192.168.139.255 scope global dynamic noprefixrou
te ens33
       valid_lft 1030sec preferred_lft 1030sec
    inet6 fe80::20c:29ff:fe8b:91ca/64 scope link
       valid_lft forever preferred_lft forever
morris@morris-VMware-Virtual-Platform:~$
```

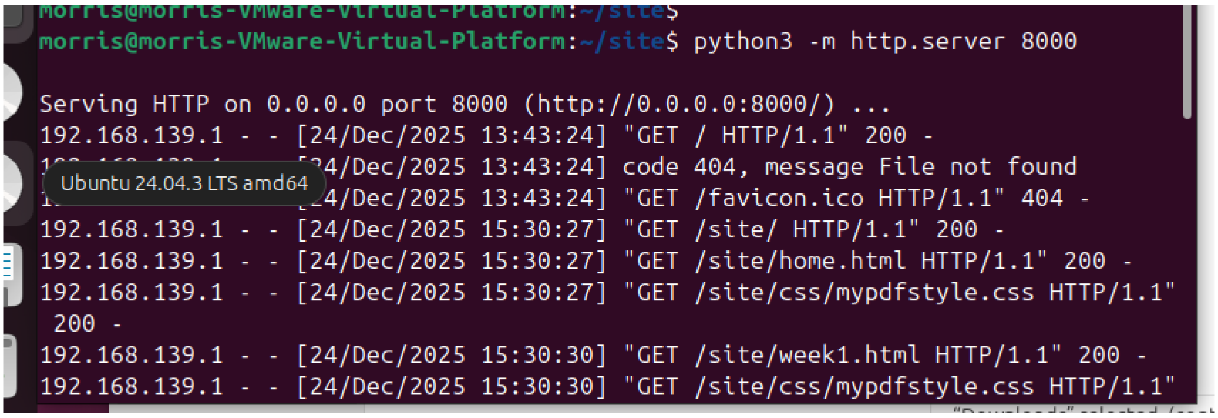
Screenshot of Site directory contents:



Hello World!

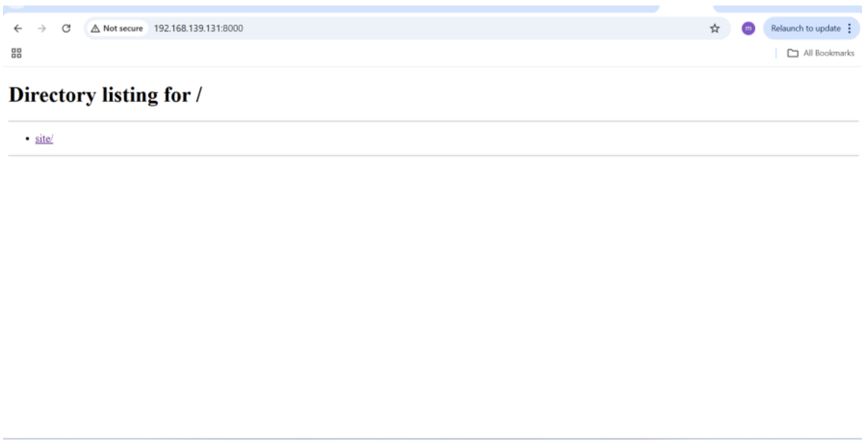
ik hou van gamen

Screenshot python3 webserver command:



```
morris@morris-VMware-Virtual-Platform:~/site$
morris@morris-VMware-Virtual-Platform:~/site$ python3 -m http.server 8000

Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.139.1 - - [24/Dec/2025 13:43:24] "GET / HTTP/1.1" 200 -
192.168.139.1 - - [24/Dec/2025 13:43:24] code 404, message File not found
1[illegible]4/Dec/2025 13:43:24] "GET /favicon.ico HTTP/1.1" 404 -
192.168.139.1 - - [24/Dec/2025 15:30:27] "GET /site/ HTTP/1.1" 200 -
192.168.139.1 - - [24/Dec/2025 15:30:27] "GET /site/home.html HTTP/1.1" 200 -
192.168.139.1 - - [24/Dec/2025 15:30:27] "GET /site/css/mypdfstyle.css HTTP/1.1"
 200 -
192.168.139.1 - - [24/Dec/2025 15:30:30] "GET /site/week1.html HTTP/1.1" 200 -
192.168.139.1 - - [24/Dec/2025 15:30:30] "GET /site/css/mypdfstyle.css HTTP/1.1"
```

Screenshot web browser visits your site



Directory listing for /

- 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 java.util.Scanner;


public class NetworkSegmentCalculator {


  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);


    System.out.print("Enter IP address (e.g. 192.168.1.100): ");
    String ipStr = sc.nextLine().trim();


    System.out.print("Enter subnet mask (e.g. 255.255.255.224) OR prefix (e.g. /27): ");
    String maskOrPrefix = sc.nextLine().trim();


    int ip = parseIPv4ToInt(ipStr);


    int mask;
    int prefixLen;
    if (maskOrPrefix.startsWith("/")) {
      prefixLen = Integer.parseInt(maskOrPrefix.substring(1));
      mask = prefixToMaskInt(prefixLen);
    } else {
      mask = parseIPv4ToInt(maskOrPrefix);
      prefixLen = maskToPrefix(mask);
    }


    int network = ip & mask;
    int broadcast = network | (~mask);
```

```
        long ipU = toUnsignedLong(ip);
        long maskU = toUnsignedLong(mask);
        long netU = toUnsignedLong(network);
        long bcU = toUnsignedLong(broadcast);

        System.out.println("\n=== Calculate the network segment ===");
        System.out.println("IP Address:   " + ipStr + "  (" + prefixLen + ")");
        System.out.println("Subnet Mask:  " + intToIPv4(mask) + "  (/" + prefixLen + ")");
        System.out.println();

        System.out.println("IP Address (bin):   " + toBinaryDotted(ip));
        System.out.println("Subnet Mask (bin):  " + toBinaryDotted(mask));
        System.out.println("-------------------------------------------");
        System.out.println("Network Addr (bin): " + toBinaryDotted(network));
        System.out.println("Network Addr:       " + intToIPv4(network));
        System.out.println();

        int blockSize = 1 << (32 - prefixLen); // e.g. /27 => 32
        System.out.println("For a /" + prefixLen + " subnet, each segment has " + blockSize + " IP
addresses (2^" + (32 - prefixLen) + ").");

        System.out.println("Range (segment):    " + intToIPv4(network) + "  to  " + intToIPv4(broadcast));

        if (prefixLen <= 30) {
            int firstHost = network + 1;
            int lastHost = broadcast - 1;
            System.out.println("Usable hosts:       " + intToIPv4(firstHost) + "  to  " + intToIPv4(lastHost));
            System.out.println("Usable host count:   " + (blockSize - 2));
        } else {
            // /31 or /32 special cases
            System.out.println("Usable hosts:        (special case /31 or /32)");
        }

        sc.close();
    }

    private static int parseIPv4ToInt(String ip) {
        String[] parts = ip.split("\\.");
        if (parts.length != 4) throw new IllegalArgumentException("Invalid IPv4: " + ip);

        int result = 0;
        for (String p : parts) {
            int oct = Integer.parseInt(p);
            if (oct < 0 || oct > 255) throw new IllegalArgumentException("Invalid octet: " + p);
            result = (result << 8) | oct;
        }
        return result;
    }

    private static String intToIPv4(int value) {
        return ((value >>> 24) & 0xFF) + "." +
               ((value >>> 16) & 0xFF) + "." +
               ((value >>> 8) & 0xFF) + "." +
               (value & 0xFF);
    }

    private static String toBinaryDotted(int value) {
        return to8BitBinary((value >>> 24) & 0xFF) + "." +
               to8BitBinary((value >>> 16) & 0xFF) + "." +
               to8BitBinary((value >>> 8) & 0xFF) + "." +
               to8BitBinary(value & 0xFF);
    }

    private static String to8BitBinary(int octet) {
        String s = Integer.toBinaryString(octet & 0xFF);
        return "0".repeat(8 - s.length()) + s;
    }

    private static int prefixToMaskInt(int prefixLen) {
        if (prefixLen < 0 || prefixLen > 32) throw new IllegalArgumentException("Invalid prefix: " +
prefixLen);
        if (prefixLen == 0) return 0;
        return (int)(0xFFFFFFFFL << (32 - prefixLen));
    }

    private static int maskToPrefix(int mask) {
        int count = 0;
        for (int i = 31; i >= 0; i--) {
            if (((mask >>> i) & 1) == 1) count++;
            else break;
        }
        return count;
    }

    private static long toUnsignedLong(int x) {
        return x & 0xFFFFFFFFL;
    }
}
```

Result:

```
Enter IP address (e.g. 192.168.1.100): 192.168.1.100
Enter subnet mask (e.g. 255.255.255.224) OR prefix (e.g. /27): /27

=== Calculate the network segment ===
IP Address:   192.168.1.100  (27)
Subnet Mask:  255.255.255.224  (/27)

IP Address (bin):   11000000.10101000.00000001.01100100
Subnet Mask (bin):  11111111.11111111.11111111.11100000
-------------------------------------------
Network Addr (bin): 11000000.10101000.00000001.01100000
Network Addr:      192.168.1.96

For a /27 subnet, each segment has 32 IP addresses (2^5).
Range (segment):    192.168.1.96  to  192.168.1.127
Usable hosts:       192.168.1.97  to  192.168.1.126
Usable host count:   30

=== Code Execution Successful ===
```

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