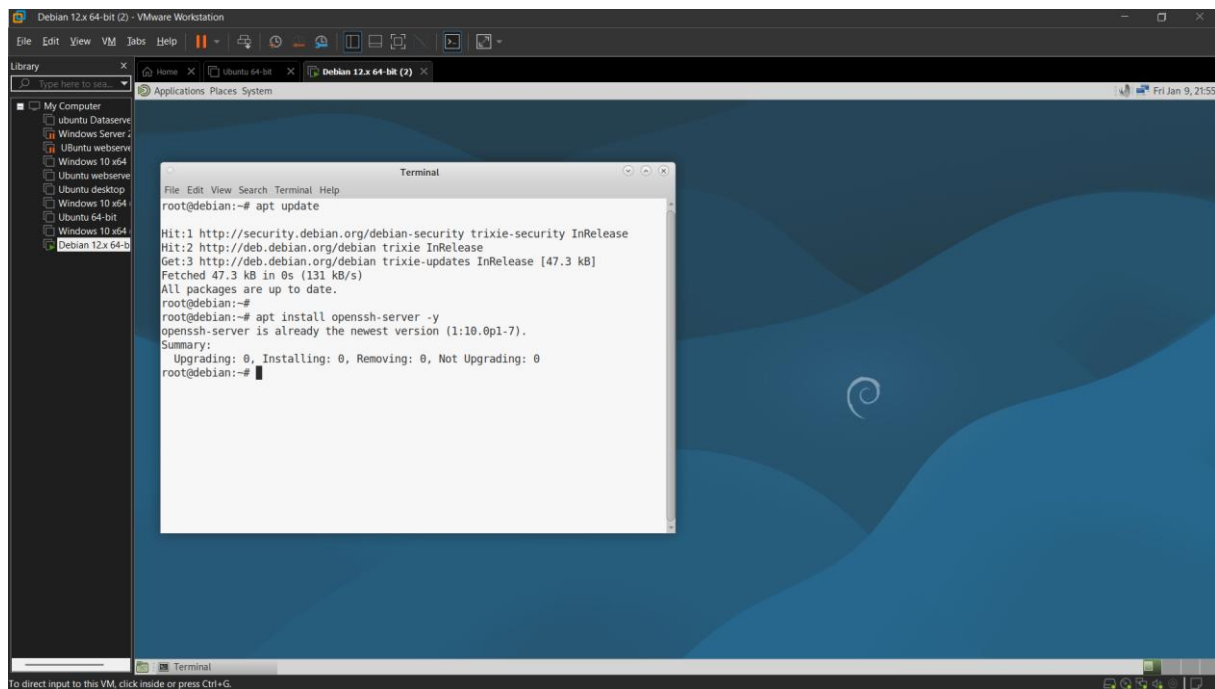


Template Week 6 – Networking

Student number: 528668

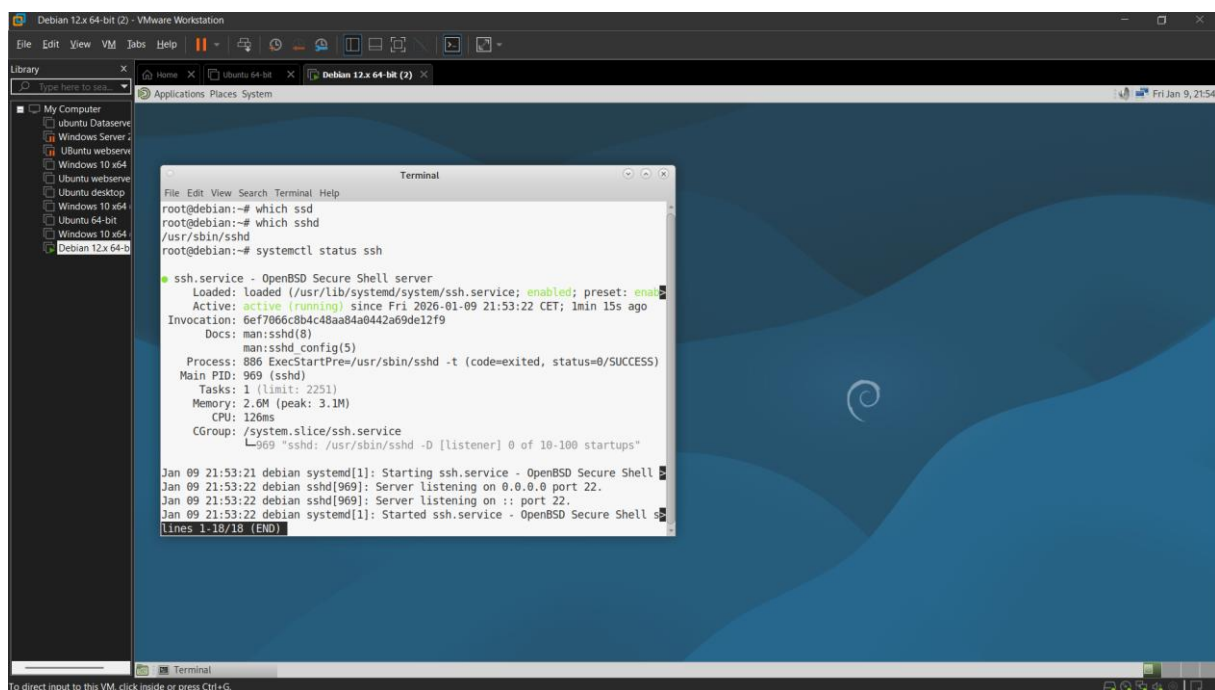
Assignment 6.1: Working from home

Screenshot installation openssh-server:



```
root@debian:~# apt update
Hit:1 http://security.debian.org/debian-security trixie-security InRelease
Hit:2 http://deb.debian.org/debian trixie InRelease
Get:3 http://deb.debian.org/debian trixie-updates InRelease [47.3 kB]
Fetched 47.3 kB in 0s (131 kB/s)
All packages are up to date.
root@debian:~# apt install openssh-server -y
openssh-server is already the newest version (1:10.0p1-7).
Summary:
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0
root@debian:~#
```

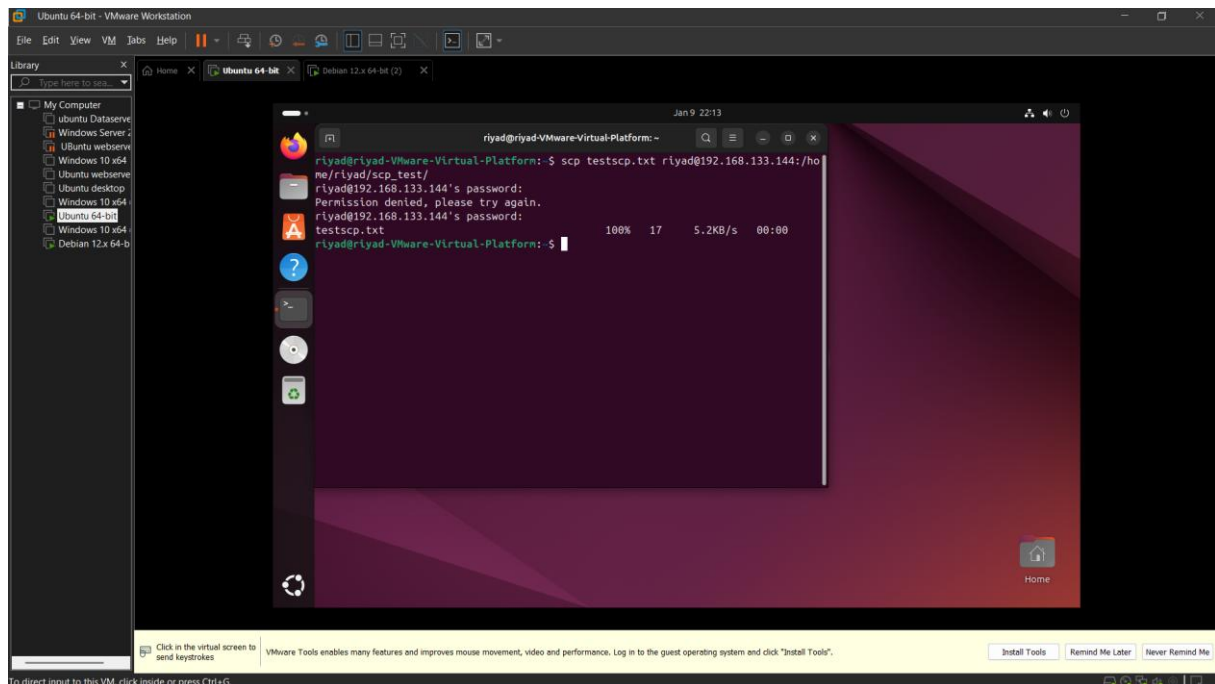
Screenshot successful SSH command execution:



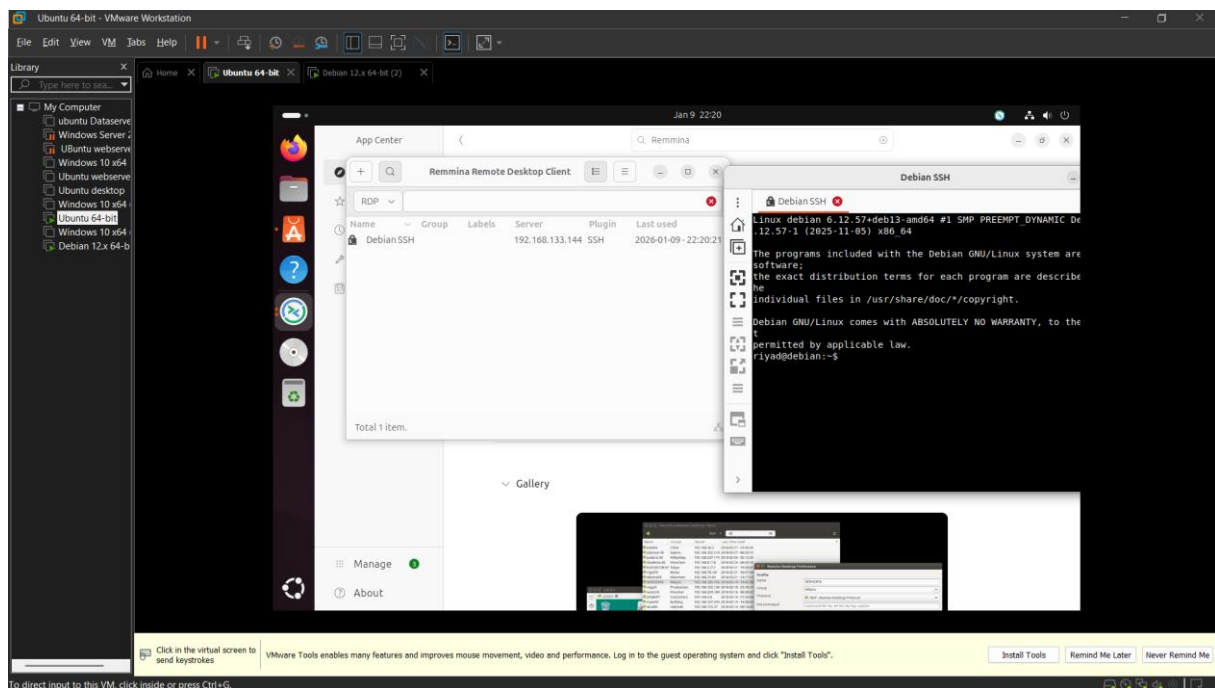
```
root@debian:~# which sshd
/usr/sbin/sshd
root@debian:~# systemctl status sshd
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enab
   Active: active (running) since Fri 2026-01-09 21:53:22 CET; 1min 15s ago
   Invocation: 6ef7066c8b4c48aa84a0442a69de12f9
   Docs: man:sshd(8)
        man:sshd_config(5)
   Process: 886 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 969 (sshd)
   Tasks: 1 (limit: 2251)
   Memory: 2.6M (peak: 3.1M)
   CPU: 126ms
   CGroup: /system.slice/ssh.service
           └─969 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Jan 09 21:53:21 debian systemd[1]: Starting ssh.service - OpenBSD Secure Shell
Jan 09 21:53:22 debian sshd[969]: Server listening on 0.0.0.0 port 22.
Jan 09 21:53:22 debian sshd[969]: Server listening on :: port 22.
Jan 09 21:53:22 debian systemd[1]: Started ssh.service - OpenBSD Secure Shell s
[Lines 1-18/18 (END)]
```

Screenshot successful execution SCP command:

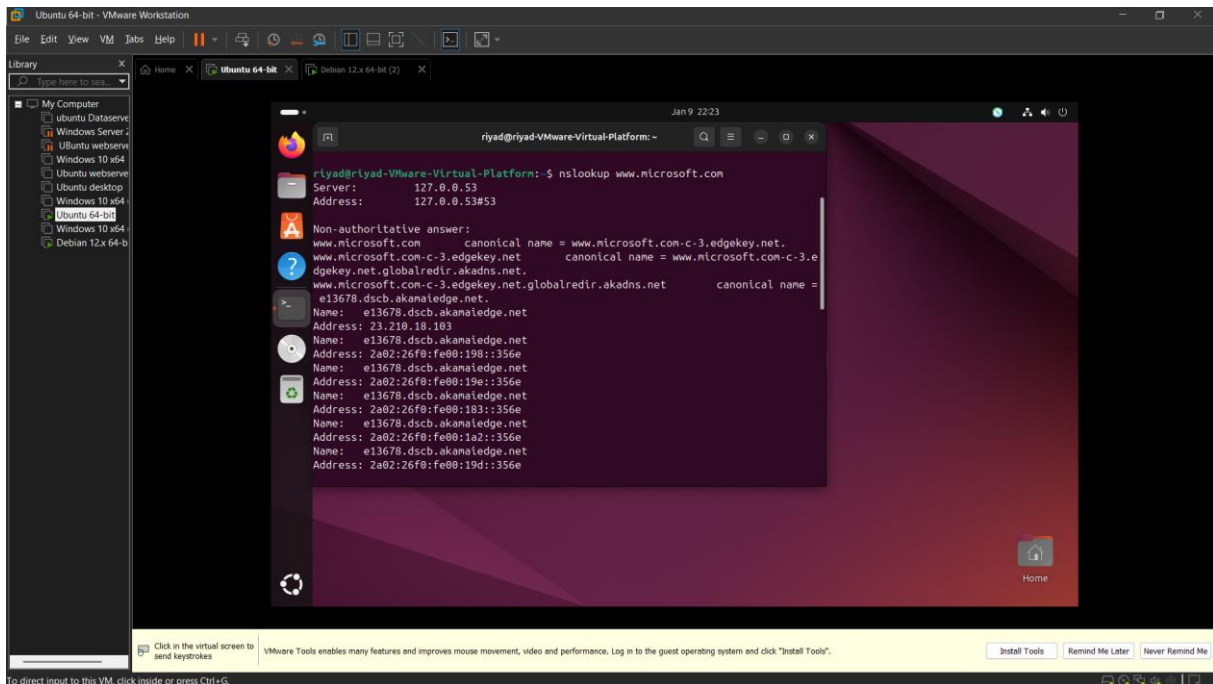


Screenshot remmina:

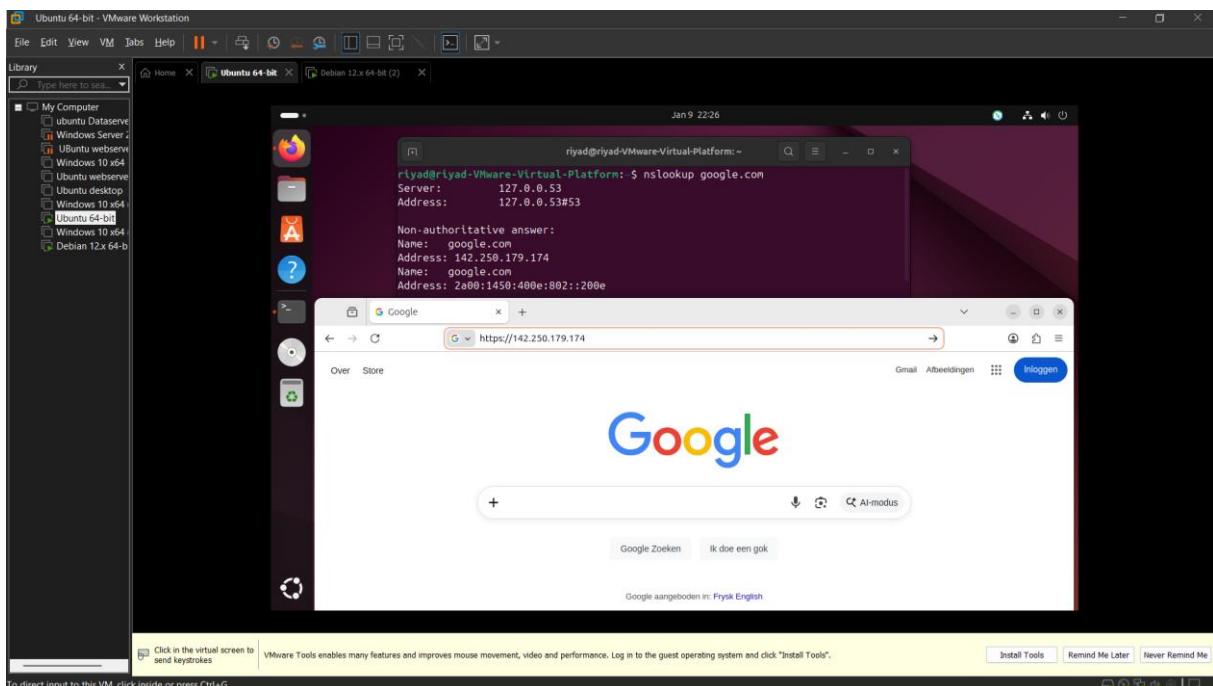


Assignment 6.2: IP addresses websites

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 /25 betekent: $32(\text{totaal aantal bits}) - 25 = 7$ host-bits.

Aantal IP-adressen = $2^7 = 128$

Dus 128 IP adressen

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

Bij een subnet zijn altijd 2 adressen “gereserveerd”:

Network address (eerste adres)

Broadcast address (laatste adres)

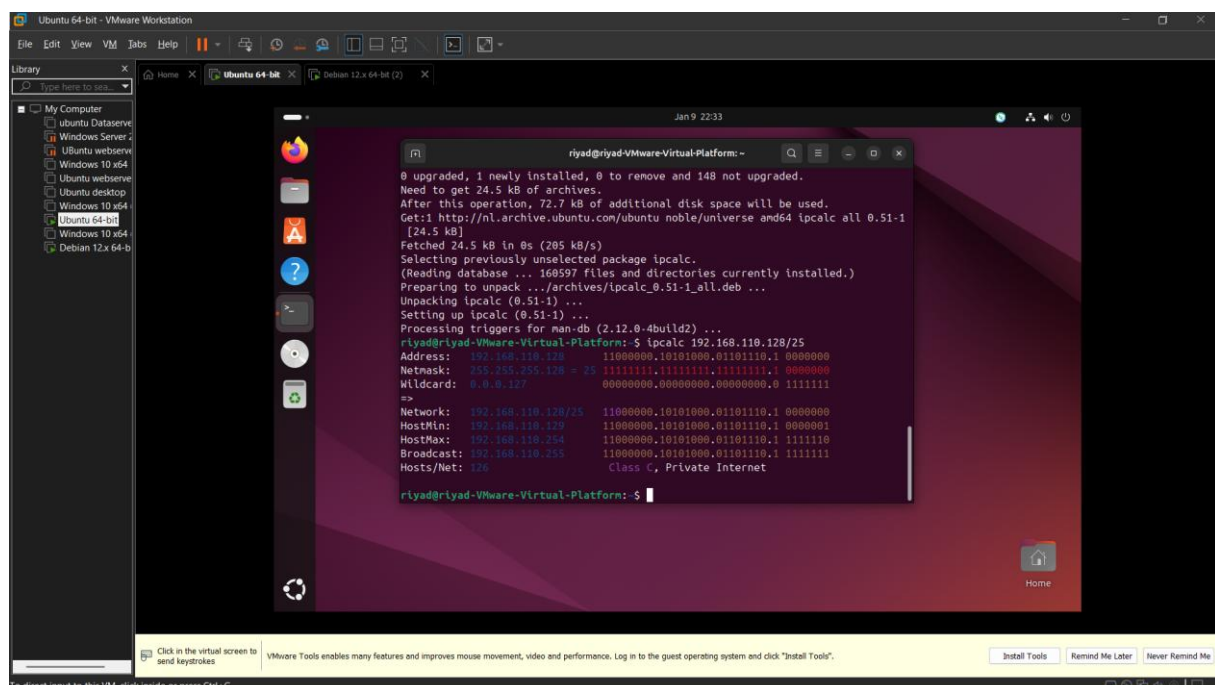
Voor 192.168.110.128/25:

Network = 192.168.110.128

Broadcast = 192.168.110.255

Dus usable hosts is $255 - 128 = 126$

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



```
riyad@riyad-Virtual-Platform: ~  
0 upgraded, 1 newly installed, 0 to remove and 148 not upgraded.  
Need to get 24.5 kB of archives:  
After this operation, 72.7 kB of additional disk space will be used.  
Get:1 http://nl.archive.ubuntu.com/ubuntu noble/universe amd64 ipcalc 0.51-1  
[24.5 kB]  
Fetched 24.5 kB in 0s (205 kB/s)  
Selecting previously unselected package ipcalc.  
(Reading database ... 160597 files and directories currently installed.)  
Preparing to unpack .../archives/ipcalc_0.51-1_all.deb ...  
Unpacking ipcalc (0.51-1) ...  
Setting up ipcalc (0.51-1) ...  
Processing triggers for man-db (2.12.0-4build2) ...  
riyad@riyad-Virtual-Platform:~$ ipcalc 192.168.110.128/25  
Address: 192.168.110.128 11000000.10101000.01101110.1 00000000  
Netmask: 255.255.255.128 = 25 11111111.11111111.11111111.1 00000000  
Wildcard: 0.0.0.127 00000000.00000000.00000000.0 11111111  
=>  
Network: 192.168.110.128/25 11000000.10101000.01101110.1 00000000  
HostMin: 192.168.110.129 11000000.10101000.01101110.1 00000001  
HostMax: 192.168.110.254 11000000.10101000.01101110.1 11111110  
Broadcast: 192.168.110.255 11000000.10101000.01101110.1 11111111  
Hosts/Net: 126 Class C, Private Internet  
riyad@riyad-Virtual-Platform:~$
```

Explain the above calculation in your own words.

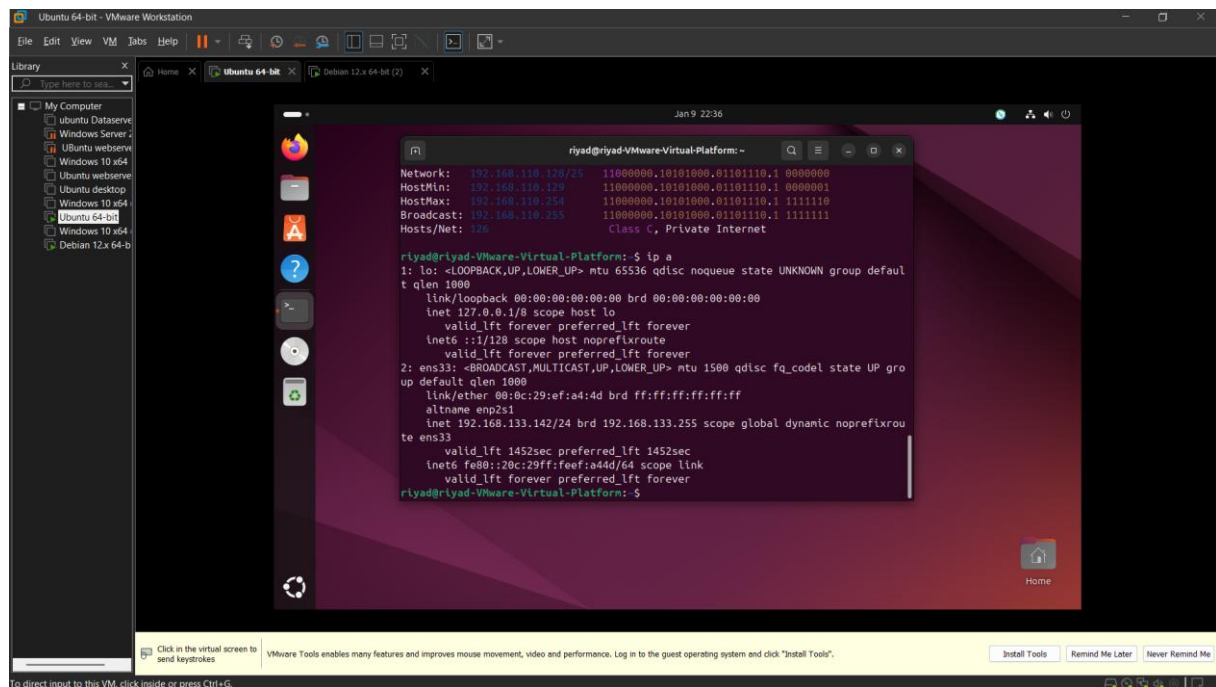
Een /25 subnet betekent dat de eerste 25 bits van het IP-adres het netwerkdeel zijn en dat er 7 bits overblijven voor hosts. Met 7 host-bits kun je $2^7 = 128$ adressen maken in dat subnet.

Het eerste adres (192.168.110.128) is het netwerkadres en wordt gebruikt om het netwerk zelf aan te duiden. Het laatste adres (192.168.110.255) is het broadcastadres, dat gebruikt wordt om naar alle apparaten in het subnet tegelijk te sturen.

Daarom kun je alleen de adressen ertussen uitdelen aan computers: 192.168.110.129 tot en met 192.168.110.254 (in totaal 126 bruikbare adressen).

Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:



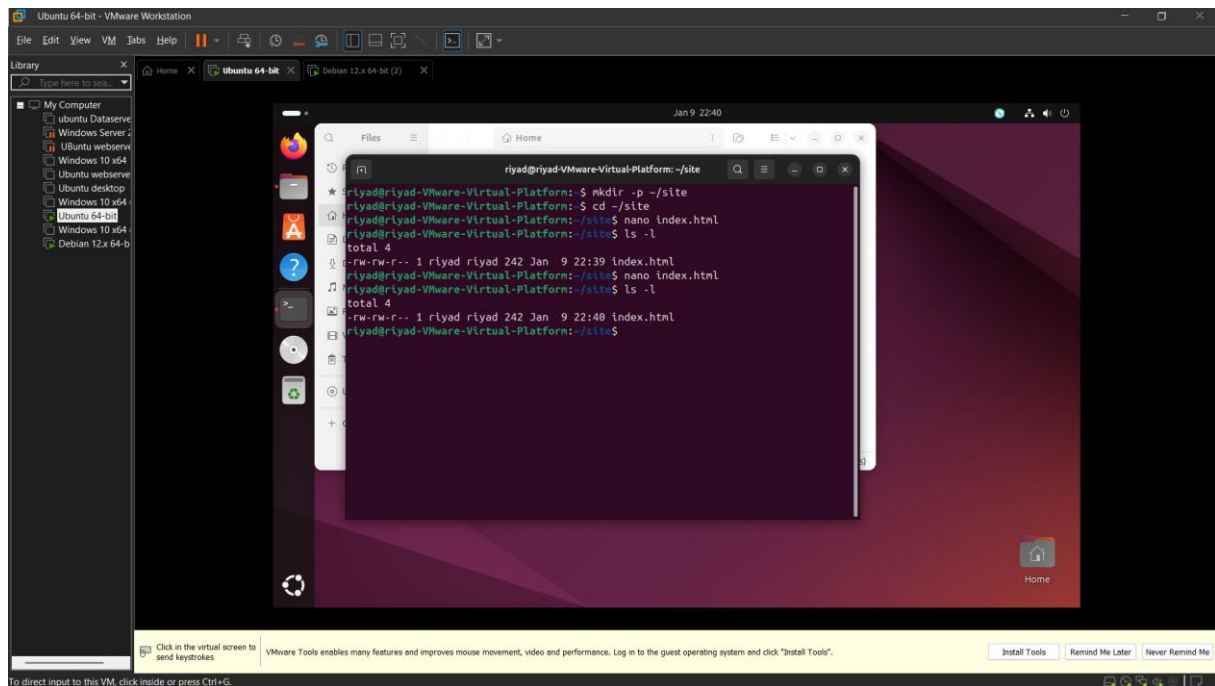
The screenshot shows a terminal window within a VMware Workstation environment. The terminal displays the output of the 'ip a' command, showing the configuration for the loopback interface 'lo' and the ethernet interface 'ens33'. The 'ens33' interface is configured with the IP address 192.168.133.142/24, which is a /24 subnet, not a /25 as mentioned in the text. The terminal output is as follows:

```
Network: 192.168.110.128/25 11000000.10101000.01101110.1 00000000
HostMin: 192.168.110.129 11000000.10101000.01101110.1 00000001
HostMax: 192.168.110.254 11000000.10101000.01101110.1 11111110
Broadcast: 192.168.110.255 11000000.10101000.01101110.1 11111111
Hosts/Net: 128 Class C, Private Internet

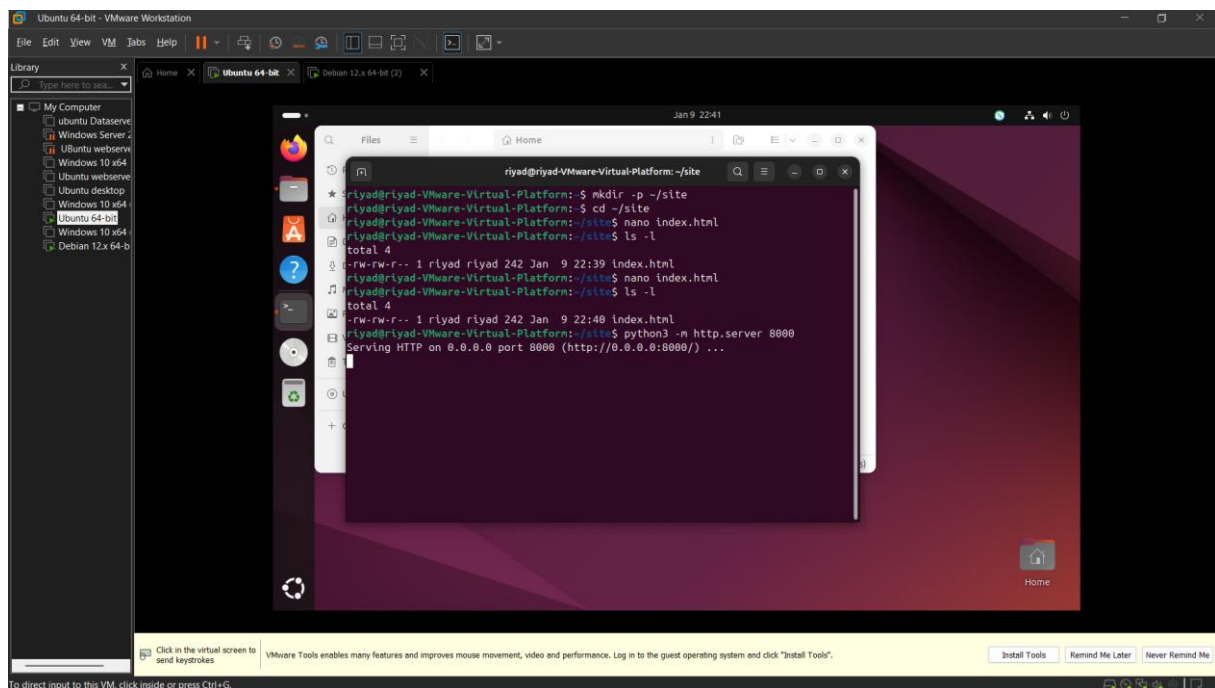
riyad@riyad-Virtual-Platform:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:ef:a4:4d brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.133.142/24 brd 192.168.133.255 scope global dynamic noprefixroute
        valid_lft 1452sec preferred_lft 1452sec
    inet6 fe80::20c:29ff:feef:a44d/64 scope link
        valid_lft forever preferred_lft forever
riyad@riyad-Virtual-Platform:~$
```

Ip address: 192.168.142/24

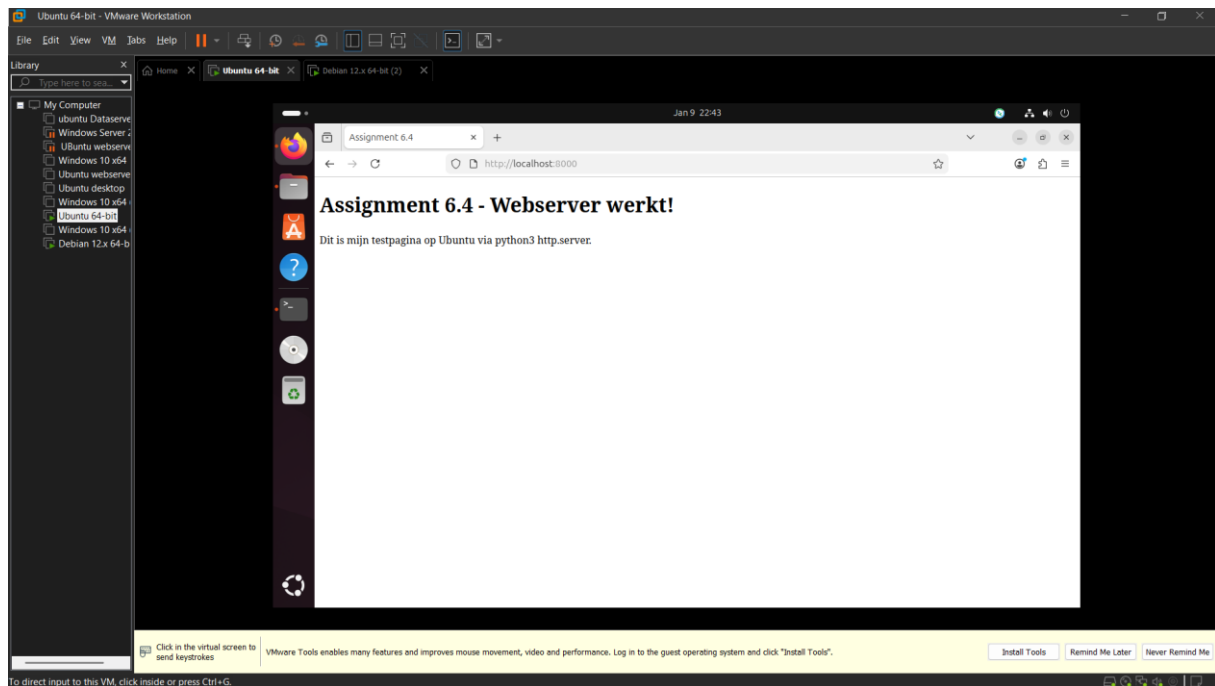
Screenshot of Site directory contents:



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

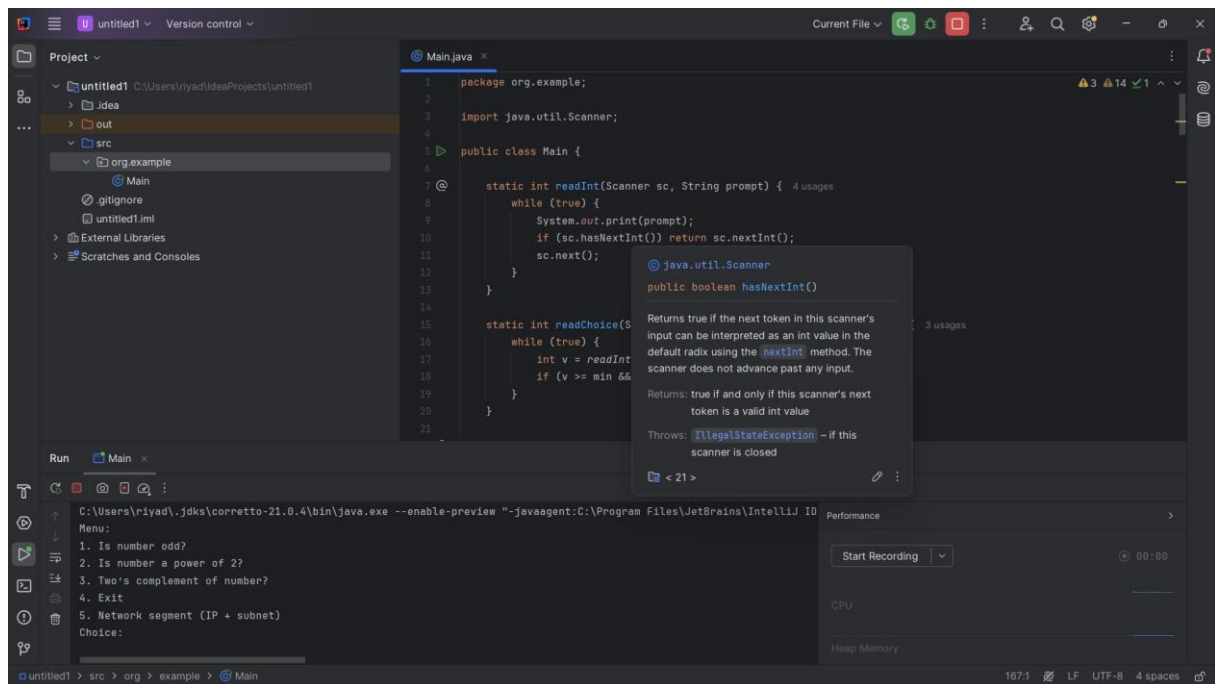
Network Addr: 11000000.10101000.00000001.01100000

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.



```
package org.example;
```

```
import java.util.Scanner;
```

```
public class Main {
```

```
    static int readInt(Scanner sc, String prompt) {
```

```
        while (true) {
```

```
            System.out.print(prompt);
```

```
            if (sc.hasNextInt()) return sc.nextInt();
```

```
            sc.next();
```

```
        }
```

```
    }
```

```
    static int readChoice(Scanner sc, String prompt, int min, int max) {
```

```
        while (true) {
```

```
            int v = readInt(sc, prompt);
```

```
            if (v >= min && v <= max) return v;
```

```
        }
```



```
}
```

```
static String readLine(Scanner sc, String prompt) {  
    System.out.print(prompt);  
    String line = sc.nextLine();  
    while (line.trim().isEmpty()) line = sc.nextLine();  
    return line.trim();  
}
```

```
static boolean isOdd(int n) {  
    return (n & 1) == 1;  
}
```

```
static boolean isPowerOfTwo(int n) {  
    return n > 0 && (n & (n - 1)) == 0;  
}
```

```
static long maskForBits(int bits) {  
    if (bits == 32) return 0xFFFFFFFFL;  
    return (1L << bits) - 1;  
}
```

```
static String toBinaryPadded(long v, int bits) {  
    String s = Long.toBinaryString(v);  
    if (s.length() > bits) s = s.substring(s.length() - bits);  
    while (s.length() < bits) s = "0" + s;  
    return s;  
}
```

```
static long twosComplement(long value, int bits) {  
    long mask = maskForBits(bits);
```

```

        return (~value + 1) & mask;
    }

```

```

static void runOdd(Scanner sc) {
    int n = readInt(sc, "Enter number: ");
    System.out.println(isOdd(n) ? "number is odd" : "number is even");
}

```

```

static void runPowerOfTwo(Scanner sc) {
    int n = readInt(sc, "Enter number: ");
    System.out.println(isPowerOfTwo(n) ? "number is a power of 2" : "number isn't a power of 2");
}

```

```

static void runTwosComplement(Scanner sc) {
    int n = readInt(sc, "Enter number: ");
    int bits = readChoice(sc, "Bit-width (8/16/32): ", 1, 32);
    while (bits != 8 && bits != 16 && bits != 32) {
        bits = readChoice(sc, "Bit-width (8/16/32): ", 1, 32);
    }
    long mask = maskForBits(bits);
    long value = ((long) n) & mask;
    long tc = twosComplement(value, bits);
    System.out.println("number: " + toBinaryPadded(value, bits));
    System.out.println("two's complement: " + toBinaryPadded(tc, bits));
}

```

```

// ===== Assignment 6.5 helpers =====

```

```

static int ipv4ToInt(String ip) {
    String[] parts = ip.split("\\.");
    if (parts.length != 4) throw new IllegalArgumentException("Invalid IP address");
}

```

```

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

```

```

static String intToIpv4(int value) {
    return ((value >>> 24) & 255) + "." +
        ((value >>> 16) & 255) + "." +
        ((value >>> 8) & 255) + "." +
        (value & 255);
}

```

```

static String toBinaryIpv4(int value) {
    String b1 = toBinaryPadded((value >>> 24) & 255, 8);
    String b2 = toBinaryPadded((value >>> 16) & 255, 8);
    String b3 = toBinaryPadded((value >>> 8) & 255, 8);
    String b4 = toBinaryPadded(value & 255, 8);
    return b1 + "." + b2 + "." + b3 + "." + b4;
}

```

```

static int prefixFromMask(int mask) {
    // Count leading 1-bits (assumes valid subnet mask)
    int prefix = 0;
    for (int i = 31; i >= 0; i--) {
        if (((mask >>> i) & 1) == 1) prefix++;
        else break;
    }
}

```

```

    return prefix;
}

static void runNetworkSegment(Scanner sc) {
    sc.nextLine(); // consume leftover newline from previous nextInt()
    String ipStr = readLine(sc, "Enter IP address (e.g. 192.168.1.100): ");
    String maskStr = readLine(sc, "Enter subnet mask (e.g. 255.255.255.224): ");

    int ip = ipv4ToInt(ipStr);
    int mask = ipv4ToInt(maskStr);

    // Bitwise AND for network address (kern van de opdracht)
    int network = ip & mask;

    int prefix = prefixFromMask(mask);
    int hostBits = 32 - prefix;
    int blockSize = 1 << hostBits;    // /27 -> 32
    int broadcast = network + blockSize - 1;

    System.out.println();
    System.out.println("Example: " + ipStr + "/" + prefix);
    System.out.println("Calculate the network segment");
    System.out.println("IP Address: " + toBinaryIpv4(ip));
    System.out.println("Subnet Mask: " + toBinaryIpv4(mask));
    System.out.println("-----");
    System.out.println("Network Addr: " + toBinaryIpv4(network));
    System.out.println();
    System.out.println("Network address (decimal): " + intToIpv4(network));
    System.out.println("Broadcast address:      " + intToIpv4(broadcast));
    System.out.println("Block size (IPs):      " + blockSize);
}

```

```

        System.out.println("Range of this segment:  " + intToIpv4(network) + " to " +
intToIpv4(broadcast));
    }

```

```

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    while (true) {
        System.out.println("Menu:");
        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. Exit");
        System.out.println("5. Network segment (IP + subnet)"); // NEW

        int choice = readChoice(sc, "Choice: ", 1, 5);
        if (choice == 4) break;
        if (choice == 1) runOdd(sc);
        if (choice == 2) runPowerOfTwo(sc);
        if (choice == 3) runTwosComplement(sc);
        if (choice == 5) runNetworkSegment(sc);

        System.out.println();
    }
}

```

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