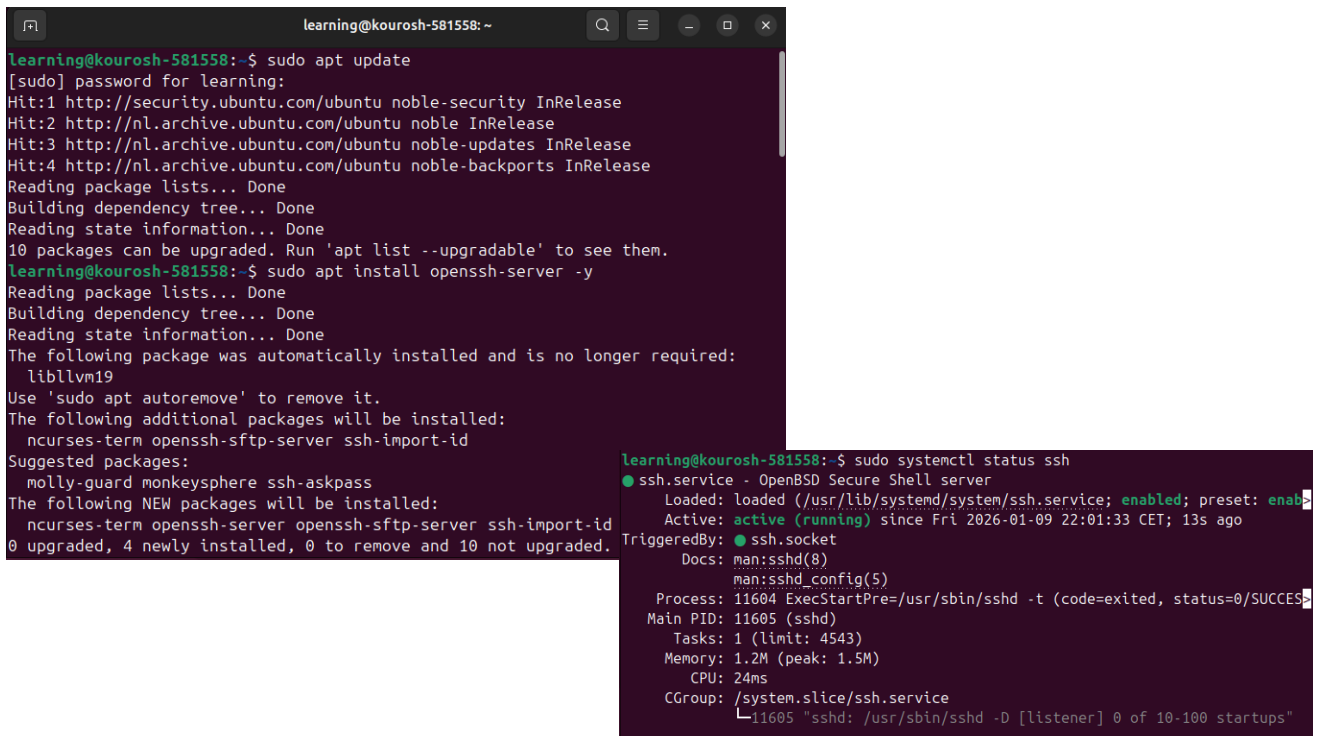


Template Week 6 – Networking

Student number: 581558

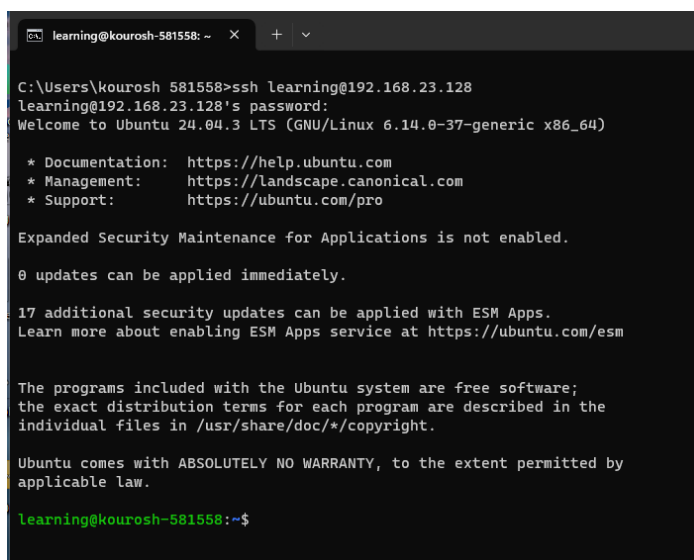
Assignment 6.1: Working from home

Screenshot installation openssh-server:



```
learning@kourosh-581558: ~  
learning@kourosh-581558:~$ sudo apt update  
[sudo] password for learning:  
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease  
Hit:2 http://nl.archive.ubuntu.com/ubuntu noble InRelease  
Hit:3 http://nl.archive.ubuntu.com/ubuntu noble-updates InRelease  
Hit:4 http://nl.archive.ubuntu.com/ubuntu noble-backports InRelease  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
10 packages can be upgraded. Run 'apt list --upgradable' to see them.  
learning@kourosh-581558:~$ 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:  
  liblvm19  
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.  
learning@kourosh-581558:~$ 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 Fri 2026-01-09 22:01:33 CET; 13s ago  
   TriggeredBy: ● ssh.socket  
     Docs: man:sshd(8)  
           man:sshd_config(5)  
   Process: 11604 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS  
   Main PID: 11605 (sshd)  
     Tasks: 1 (limit: 4543)  
    Memory: 1.2M (peak: 1.5M)  
       CPU: 24ms  
   CGroup: /system.slice/ssh.service  
           └─11605 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
```

Screenshot successful SSH command execution:



```
C:\Users\kourosh 581558>ssh learning@192.168.23.128  
learning@192.168.23.128'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.  
  
0 updates can be applied immediately.  
  
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.  
  
learning@kourosh-581558:~$
```

Screenshot successful execution SCP command:

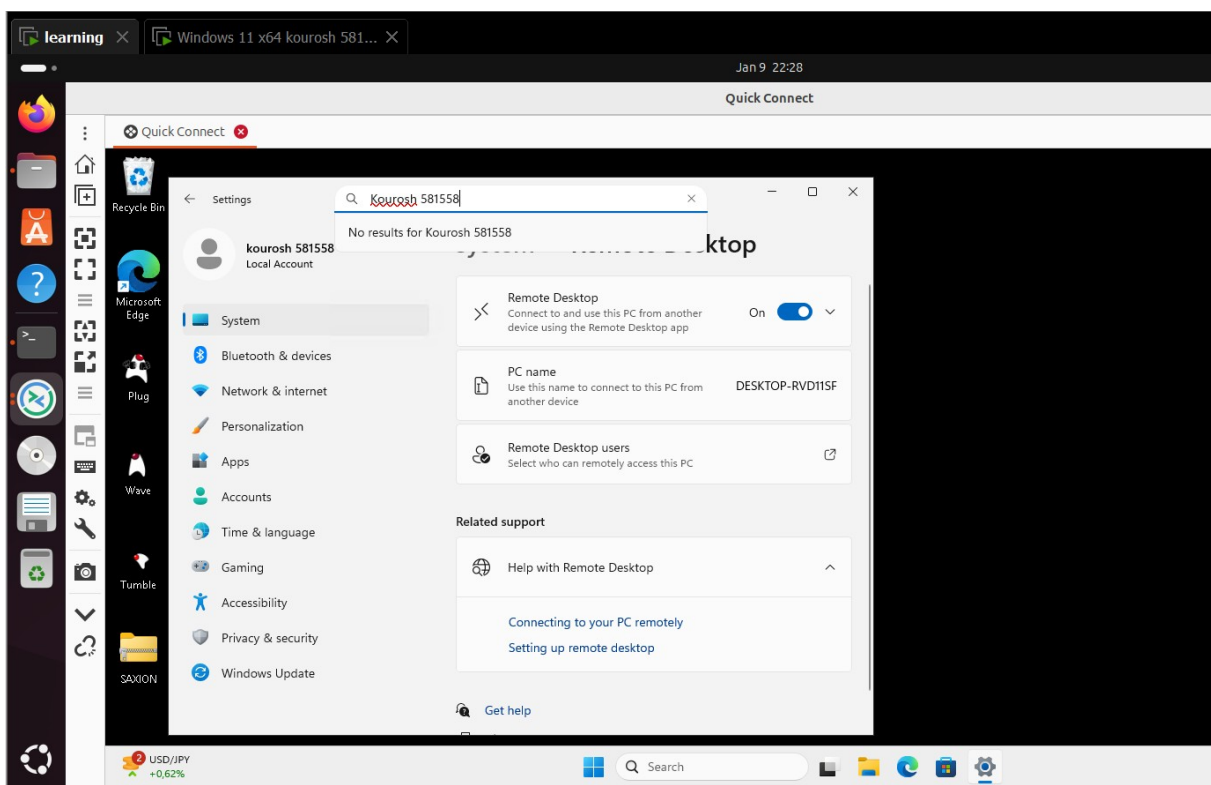
```
C:\Users\kourosh 581558>scp testfile.txt learning@192.168.23.128:/home/learning
learning@192.168.23.128's password:
testfile.txt                                100%  21    5.1KB/s

C:\Users\kourosh 581558>
```

```
learning@kourosh-581558:~$ ls
archive.tar.gz  Documents  hello  Pictures  snap  Templates  Videos
Desktop        Downloads  Music  Public   stuff.txt  testfile.txt
learning@kourosh-581558:~$
```

Screenshot remmina:

```
learning@kourosh-581558:~$ sudo apt install remmina remmina-plugin-rdp -y
```



Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:

```
learning@kourosh-581558:~$ nslookup amazon.com
Server:      127.0.0.53
Address:     127.0.0.53#53

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

learning@kourosh-581558:~$ nslookup google.com
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 142.251.39.142
Name:   google.com
Address: 2a00:1450:400e:801::200e

learning@kourosh-581558:~$ nslookup one.one.one.one
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   one.one.one.one
Address: 1.1.1.1
Name:   one.one.one.one
Address: 1.0.0.1
Name:   one.one.one.one
Address: 2606:4700:4700::1001
Name:   one.one.one.one

learning@kourosh-581558:~$ nslookup dns.google.com
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   dns.google.com
Address: 8.8.4.4
Name:   dns.google.com
Address: 8.8.8.8
Name:   dns.google.com
Address: 2001:4860:4860::8844
Name:   dns.google.com
Address: 2001:4860:4860::8888

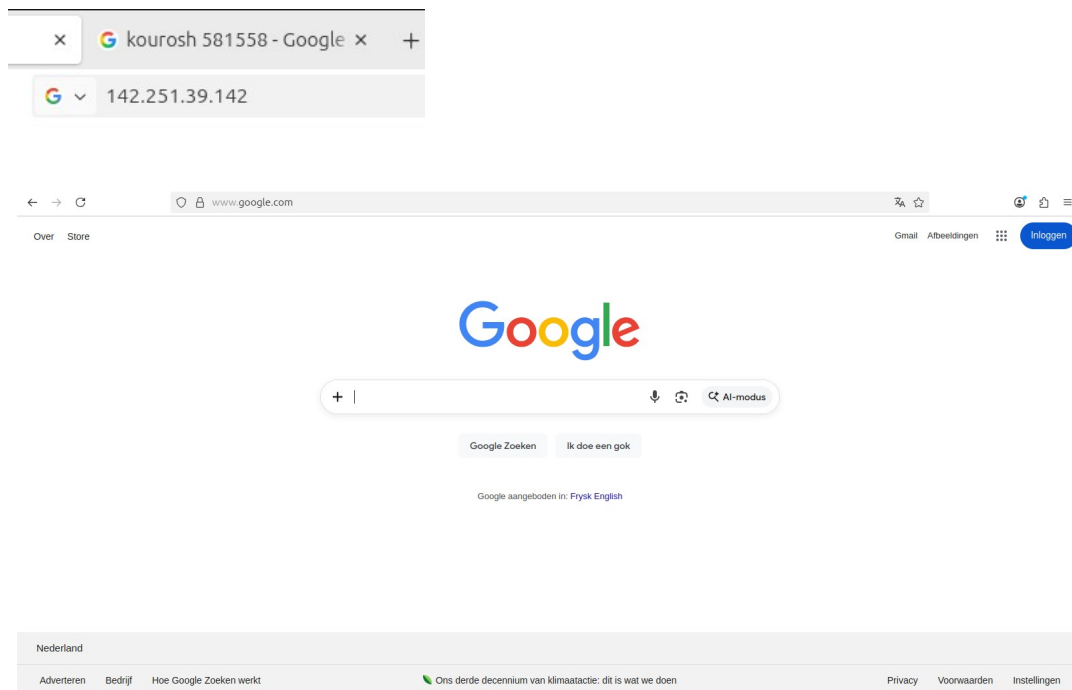
learning@kourosh-581558:~$ nslookup bol.com
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   bol.com
Address: 79.170.100.62

learning@kourosh-581558:~$ nslookup w3schools.com
Server:      127.0.0.53
Address:     127.0.0.53#53

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

Screenshot website visit via IP address:



Assignment 6.3: subnetting

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

128 available IP. Its has 25 bits for the network and the rest for the host. IPv4 addresses have 32 bits, so $32 - 25 = 7$ bits is for the host. $2^7 = 128$ is the total number of possible IPs.

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

First IP → 192.168.110.128 (Not usable, it identifies the network itself)

Last IP → 192.168.110.255 (Not usable, it is sent to all nodes within the network)

So the usable range is 192.168.110.129 to 192.168.110.254.

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

Address: 192.168.110.128

Netmask: 255.255.255.128 = 25

Network: 192.168.110.128/25

Broadcast: 192.168.110.255

HostMin: 192.168.110.129

HostMax: 192.168.110.254

Hosts/Net: 126

Explain the above calculation in your own words.

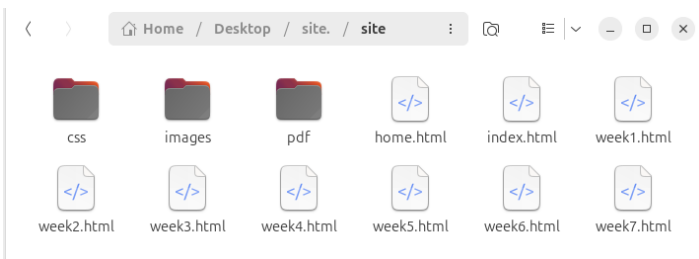
Explained.

Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

```
learning@kourosch-581558:~/Desktop/site$ 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:fd:0b:e7 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.23.128/24 brd 192.168.23.255 scope global dynamic noprefixroute ens33
        valid_lft 1129sec preferred_lft 1129sec
    inet6 fe80::20c:29ff:fe7d:be7/64 scope link
        valid_lft forever preferred_lft forever
```

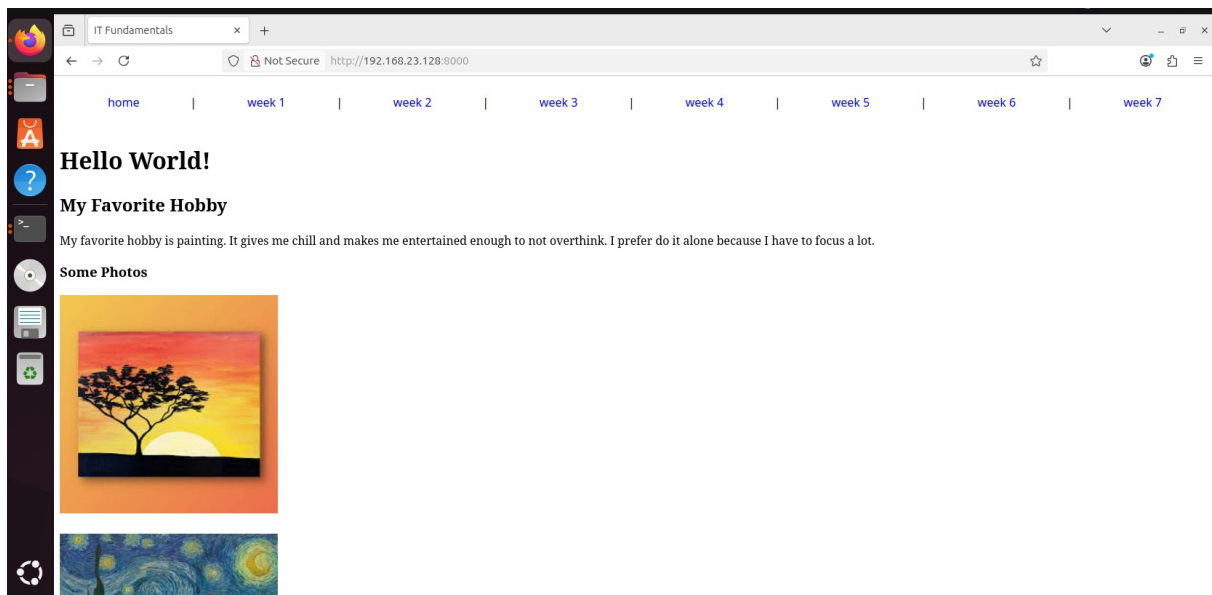
Screenshot of Site directory contents:



Screenshot python3 webserver command:

```
learning@kourosch-581558:~/Desktop/site$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

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("Kourosh 581558\nEnter an integer number: ");
        int number = scanner.nextInt();

        System.out.println("\nSelect an operation:");
        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. Calculate network segment from IP and subnet");
        System.out.print("Enter your choice (1-4): ");

        int choice = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        switch (choice) {
            case 1:
                checkIfOdd(number);
                break;
            case 2:
                checkIfPowerOfTwo(number);
                break;
            case 3:
                printTwosComplement(number);
                break;
            case 4:
                System.out.print("Enter IP address (e.g., 192.168.1.100): ");
                String ip = scanner.nextLine();
                System.out.print("Enter subnet mask (e.g., 255.255.255.224): ");
                String subnet = scanner.nextLine();
                calculateNetworkSegment(ip, subnet);
                break;
            default:
                System.out.println("Invalid choice.");
        }

        scanner.close();
    }

    public static void checkIfOdd(int n) {
        if ((n & 1) == 1) {
            System.out.println("The number " + n + " is ODD.");
        } else {

```

```

        System.out.println("The number " + n + " is EVEN.");
    }
}

public static void checkIfPowerOfTwo(int n) {
    if (n > 0 && (n & (n - 1)) == 0) {
        System.out.println("Yes, " + n + " is a power of 2.");
    } else {
        System.out.println("No, " + n + " is NOT a power of 2.");
    }
}

public static void printTwosComplement(int n) {
    int result = ~n + 1;
    System.out.println("The two's complement of " + n + " is: " + result);
}

public static void calculateNetworkSegment(String ip, String subnet) {
    String[] ipParts = ip.split("\\.");
    String[] subnetParts = subnet.split("\\.");

    int[] ipInts = new int[4];
    int[] subnetInts = new int[4];
    int[] networkInts = new int[4];

    for (int i = 0; i < 4; i++) {
        ipInts[i] = Integer.parseInt(ipParts[i]);
        subnetInts[i] = Integer.parseInt(subnetParts[i]);
        networkInts[i] = ipInts[i] & subnetInts[i];
    }

    System.out.println("\nIP Address: " + toBinaryString(ipInts));
    System.out.println("Subnet Mask: " + toBinaryString(subnetInts));
    System.out.println("-----");
    System.out.println("Network Addr: " + toBinaryString(networkInts));

    System.out.println("Network Address (decimal): " +
        networkInts[0] + "." + networkInts[1] + "." +
        networkInts[2] + "." + networkInts[3]);

    int hostBits = 0;
    for (int i = 0; i < 4; i++) {
        int maskPart = subnetInts[i];
        hostBits += Integer.bitCount(~maskPart & 0xFF);
    }
    int totalHosts = (int) Math.pow(2, hostBits);
    int lastHost = networkInts[3] + totalHosts - 1;
}

```

```

        System.out.println("Range of this network segment: " +
            networkInts[0] + "." + networkInts[1] + "." +
            networkInts[2] + "." + networkInts[3] + " to " +
            networkInts[0] + "." + networkInts[1] + "." +
            networkInts[2] + "." + lastHost);
    }

    private static String toBinaryString(int[] arr) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < arr.length; i++) {
            sb.append(String.format("%8s", Integer.toBinaryString(arr[i])).replace(' ', '0'));
            if (i < arr.length - 1) sb.append(".");
        }
        return sb.toString();
    }
}

```

```

97 System.out.println("Range of this network segment: " +
98     networkInts[0] + "." + networkInts[1] + "." +
99     networkInts[2] + "." + networkInts[3] + " to "
100     +
101     networkInts[0] + "." + networkInts[1] + "." +
102     networkInts[2] + "." + lastHost);
103 }
104 private static String toBinaryString(int[] arr) {
105     StringBuilder sb = new StringBuilder();
106     for (int i = 0; i < arr.length; i++) {
107         sb.append(String.format("%8s", Integer
108             .toBinaryString(arr[i])).replace(' ', '0'));
109         if (i < arr.length - 1) sb.append(".");
110     }
111 }

```

Select an operation:
 1. Is number odd?
 2. Is number a power of 2?
 3. Two's complement of number?
 4. Calculate network segment from IP and subnet
 Enter your choice (1-4): 4
 Enter IP address (e.g., 192.168.1.100): 122.23.23.455
 Enter subnet mask (e.g., 255.255.255.224): 255.255.255.223

IP Address: 01111010.00010111.00010111.11100011
 Subnet Mask: 11111111.11111111.11111111.11011111

 Network Addr: 01111010.00010111.00010111.11000111
 Network Address (decimal): 122.23.23.199

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