

# Template Week 6 – Networking

Student number:568261

## 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?

128 IP addresses

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

192.168.110.129 – 192.168.110.254

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

Network: 192.168.110.128

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.

A /25 subnet has 7 host bits, giving  $2^7 = 128$  total IPs. The first IP is the network address and the last IP is the broadcast, leaving 126 usable addresses (192.168.110.129 to 192.168.110.254) for devices. @

#### Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

```
valid_lmt forever preferred_lmt forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
  link/ether 00:0c:29:f2:f7:8f brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.139.129/24 brd 192.168.139.255 scope global dynamic noprefixroute ens33
```

Screenshot of Site directory contents:

```
omer@ubuntu568261:~/Downloads/site$ ls -R
.:
css      images      pdf      week2.html  week4.html  week6.html
home.html index.html week1.html  week3.html  week5.html  week7.html

./css:
mypdfstyle.css

./images:
photo1.jpg

./pdf:
week1.pdf  week2.pdf  week3.pdf  week4.pdf  week5.pdf  week6.pdf  week7.pdf
```

Screenshot python3 webserver command:

```
Traceback (most recent call last):
  File "<frozen runpy>", line 198, in _run_module_as_main
  File "<frozen runpy>", line 88, in _run_code
  File "/usr/lib/python3.12/http/server.py", line 1314, in <module>
    test()
  File "/usr/lib/python3.12/http/server.py", line 1261, in test
    with ServerClass(addr, HandlerClass) as httpd:
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3.12/socketserver.py", line 457, in __init__
    self.server_bind()
  File "/usr/lib/python3.12/http/server.py", line 1308, in server_bind
    return super().server_bind()
    ^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3.12/http/server.py", line 136, in server_bind
    socketserver.TCPServer.server_bind(self)
  File "/usr/lib/python3.12/socketserver.py", line 473, in server_bind
    self.socket.bind(self.server_address)
    ^^^^^^
```

Screenshot web browser visits your site



Hello World!

My Hobby:being lazy

i like to do absolutely nothing just be lazy and sleep:

#### 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.

```
C:\Users\omerf\.jdks\ms-21.0.8\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Ed
Welcome to the Bit Calculations App!
Select an option:
1. Bitwise number operations (odd, power of 2, two's complement)
2. Calculate network segment from IP and subnet
Your choice: 2
Enter IP address (e.g., 192.168.1.100): |
```

```

import java.util.Scanner;

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

        System.out.println("Welcome to the Bit Calculations App!");

        System.out.println("Select an option:");
        System.out.println("1. Bitwise number operations (odd, power of 2, two's complement)");
        System.out.println("2. Calculate network segment from IP and subnet");
        System.out.print("Your choice: ");

        int choice = 0;
        if (scanner.hasNextInt()) {
            choice = scanner.nextInt();
            scanner.nextLine(); // consume newline
        } else {
            System.out.println("Invalid input! Please enter 1 or 2.");
            scanner.close();
            return;
        }

        switch (choice) {
            case 1:
                performNumberOperations(scanner);
                break;
            case 2:
                calculateNetworkSegment(scanner);
                break;
            default:
                System.out.println("Invalid choice! Please select 1 or 2.");
        }

        scanner.close();
    }
}

// -----
// Bitwise number operations
// -----
public static void performNumberOperations(Scanner scanner) {
    System.out.print("Enter an integer number: ");
    int number;
    if (scanner.hasNextInt()) {
        number = scanner.nextInt();
    } else {
        System.out.println("Invalid input! Please enter an integer.");
        return;
    }
}

```

```

}

System.out.println("Select an operation:");
System.out.println("1. Is the number odd?");
System.out.println("2. Is the number a power of 2?");
System.out.println("3. Two's complement of the number");
System.out.print("Your choice: ");

int choice = scanner.nextInt();

switch (choice) {
    case 1:
        System.out.println(number + (isOdd(number) ? " is odd." : " is even."));
        break;
    case 2:
        System.out.println(number + (isPowerOfTwo(number) ? " is a power of 2." : " is NOT a power
of 2."));
        break;
    case 3:
        System.out.println("Two's complement of " + number + " is: " + twosComplement(number));
        break;
    default:
        System.out.println("Invalid option! Please choose 1, 2, or 3.");
}
}

public static boolean isOdd(int number) {
    return (number & 1) == 1;
}

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

public static int twosComplement(int number) {
    return ~number + 1;
}

// -----
// Network segment calculation
// -----
public static void calculateNetworkSegment(Scanner scanner) {
    System.out.print("Enter IP address (e.g., 192.168.1.100): ");
    String ipStr = scanner.nextLine();

    System.out.print("Enter subnet mask (e.g., 255.255.255.224): ");
    String subnetStr = scanner.nextLine();
}

```

```

int[] ip = parseIP(ipStr);
int[] subnet = parseIP(subnetStr);

if (ip == null || subnet == null) {
    System.out.println("Invalid IP or subnet format!");
    return;
}

int[] network = new int[4];
for (int i = 0; i < 4; i++) {
    network[i] = ip[i] & subnet[i];
}

System.out.println("\nIP Address: " + toBinaryString(ip));
System.out.println("Subnet Mask: " + toBinaryString(subnet));
System.out.println("-----");
System.out.println("Network Addr: " + toBinaryString(network));
System.out.println("\nNetwork Address in decimal: " + network[0] + "." + network[1] + "." +
network[2] + "." + network[3]);

int hostBits = 0;
for (int i = 0; i < 4; i++) {
    hostBits += Integer.bitCount(~subnet[i] & 0xFF);
}

int numberofHosts = (int) Math.pow(2, hostBits);
int[] broadcast = network.clone();
broadcast[3] += numberofHosts - 1;

System.out.println("Usable IP range: " + network[0] + "." + network[1] + "." + network[2] + "." +
(network[3]+1)
+ " - " + broadcast[0] + "." + broadcast[1] + "." + broadcast[2] + "." + (broadcast[3]-1));
}

public static int[] parseIP(String ipStr) {
    String[] parts = ipStr.split("\\.");
    if (parts.length != 4) return null;
    int[] ip = new int[4];
    try {
        for (int i = 0; i < 4; i++) {
            ip[i] = Integer.parseInt(parts[i]);
        }
    } catch (NumberFormatException e) {
        return null;
    }
    return ip;
}

```

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

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