Template Week 6 – Networking

Student number:569527

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?

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

Check your two previous answers with this calculator:
https://www.calculator.net/ip-subnet-calculator.html

Explain the above calculation in your own words.

Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

Screenshot of Site directory contents:

Screenshot python3 webserver command:

Screenshot web browser visits your site

Bonus point assignment – week 6

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.
```

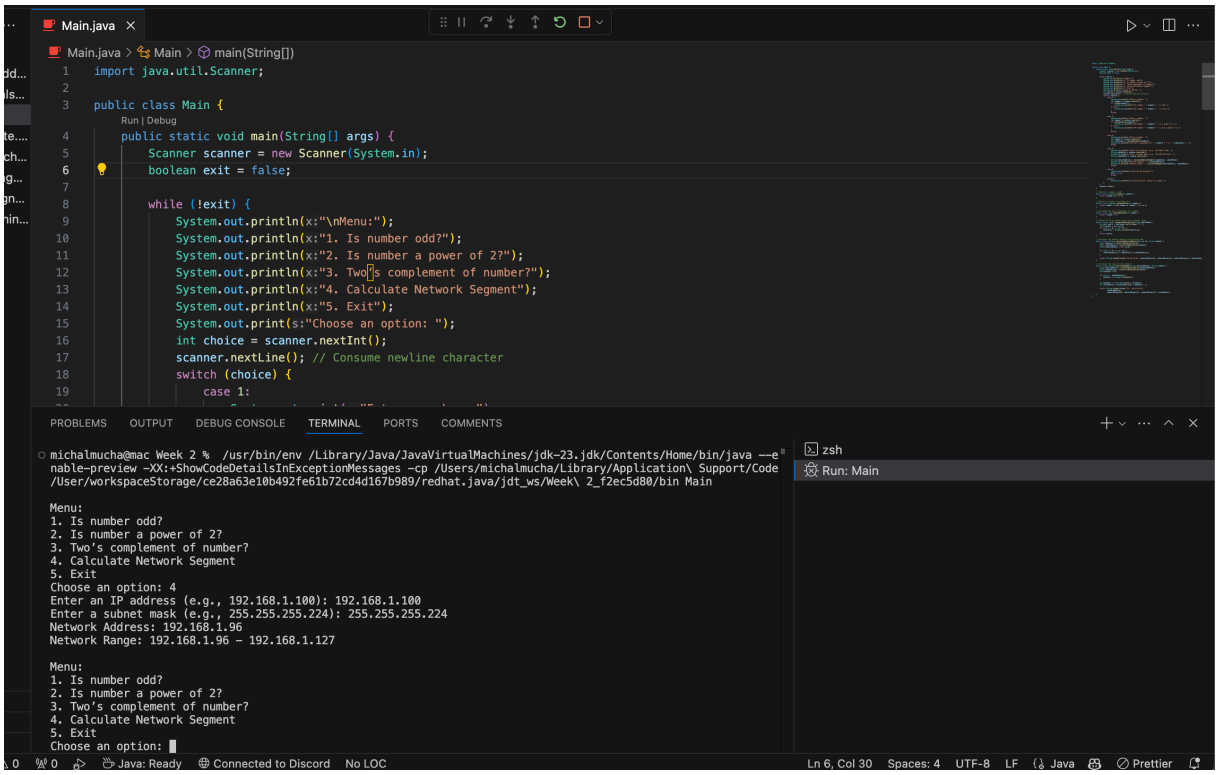
Paste source code here, with a screenshot of a working application.

```java
import java.util.Scanner;

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

```
        while (!exit) {
            System.out.println("\nMenu:");
            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");
            System.out.println("5. Exit");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline character
            switch (choice) {
                case 1:
                    System.out.print("Enter a number: ");
                    int number1 = scanner.nextInt();
                    if (isOdd(number1)) {
                        System.out.println("The number " + number1 + " is odd.");
                    } else {
                        System.out.println("The number " + number1 + " is even.");
                    }
                    break;

                case 2:
                    System.out.print("Enter a number: ");
                    int number2 = scanner.nextInt();
                    if (isPowerOfTwo(number2)) {
                        System.out.println("The number " + number2 + " is a power
of 2.");
                    } else {
                        System.out.println("The number " + number2 + " is not a
power of 2.");
                    }
                    break;

                case 3:
                    System.out.print("Enter a number: ");
                    int number3 = scanner.nextInt();
                    int complement = twosComplement(number3);
                    System.out.println("The two's complement of " + number3 + " is
" + complement + ".");
                    break;

                case 4:
                    System.out.print("Enter an IP address (e.g., 192.168.1.100):
");
                    String ipAddress = scanner.nextLine();
                    System.out.print("Enter a subnet mask (e.g., 255.255.255.224):
");
                    String subnetMask = scanner.nextLine();
```

```
                    String networkAddress = calculateNetworkAddress(ipAddress,
subnetMask);
                    System.out.println("Network Address: " + networkAddress);
                    System.out.println("Network Range: " +
calculateRange(networkAddress, subnetMask));
                    break;

                case 5:
                    System.out.println("Exiting the program.");
                    exit = true;
                    break;

                default:
                    System.out.println("Invalid choice. Please try again.");
            }
        }
        scanner.close();
    }

    // Check if a number is odd
    public static boolean isOdd(int number) {
        return (number & 1) != 0;
    }

    // Check if a number is a power of 2
    public static boolean isPowerOfTwo(int number) {
        return number > 0 && (number & (number - 1)) == 0;
    }

    // Calculate the two's complement of a number
    public static int twosComplement(int number) {
        return ~number + 1;
    }

    // Convert an IP or subnet string to an integer array
    public static int[] convertToBinaryArray(String ipOrSubnet) {
        String[] parts = ipOrSubnet.split("\\.");
        int[] binary = new int[4];
        for (int i = 0; i < 4; i++) {
            binary[i] = Integer.parseInt(parts[i]);
        }
        return binary;
    }

    // Calculate the network address using bitwise AND
    public static String calculateNetworkAddress(String ip, String subnet) {
        int[] ipBinary = convertToBinaryArray(ip);
        int[] subnetBinary = convertToBinaryArray(subnet);
        int[] networkBinary = new int[4];

        for (int i = 0; i < 4; i++) {
```

```
            networkBinary[i] = ipBinary[i] & subnetBinary[i];
        }

        return String.format("%d.%d.%d.%d", networkBinary[0], networkBinary[1],
networkBinary[2], networkBinary[3]);
    }

    // Calculate the range of the network
    public static String calculateRange(String networkAddress, String subnet) {
        int[] networkBinary = convertToBinaryArray(networkAddress);
        int[] subnetBinary = convertToBinaryArray(subnet);
        int hostBits = 32;

        for (int i : subnetBinary) {
            hostBits -= Integer.bitCount(i);
        }

        int maxHosts = (int) Math.pow(2, hostBits);
        int lastAddress = networkBinary[3] + maxHosts - 1;

        return String.format("%s - %d.%d.%d.%d",
                networkAddress,
                networkBinary[0], networkBinary[1], networkBinary[2], lastAddress);
    }
}
```



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