# Assignment 1 [E0321041]

## Create a neural network

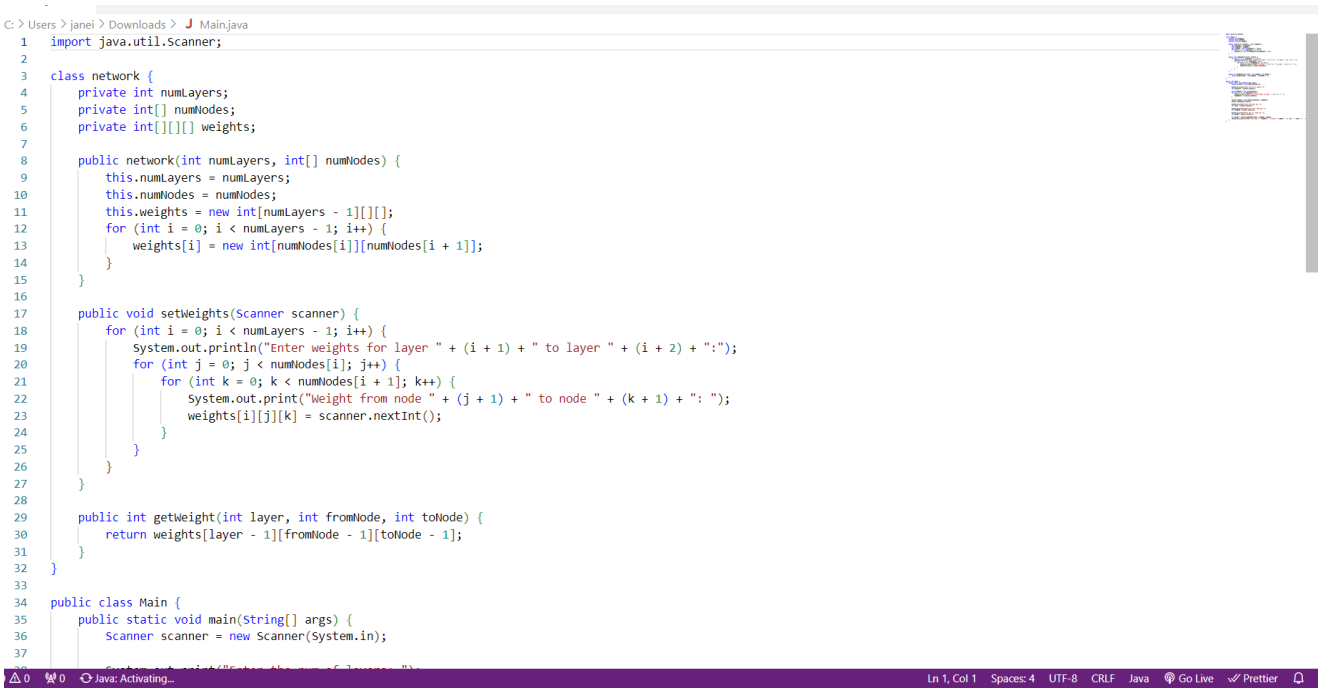
- Question:

A neural network consists of layers. The first layer is the input and the last layer is the output. The layers between the input and output are called hidden layers. The nodes of one layer can be connected to the nodes of the next layer. Each connection between two nodes (called an edge) has a weight.

Write a java code which does the following :

It takes the number of layers, and nodes in each layer as input, and creates a class object.
It sets each edge's weight from user input
It should be possible to query the weight of any edge, given the node.

- Code:



C: > Users > janei > Downloads > Main.java

```java
import java.util.Scanner;

class network {
    private int numLayers;
    private int[] numNodes;
    private int[][][] weights;

    public network(int numLayers, int[] numNodes) {
        this.numLayers = numLayers;
        this.numNodes = numNodes;
        this.weights = new int[numLayers - 1][][];
        for (int i = 0; i < numLayers - 1; i++) {
            weights[i] = new int[numNodes[i]][numNodes[i + 1]];
        }
    }

    public void setWeights(Scanner scanner) {
        for (int i = 0; i < numLayers - 1; i++) {
            System.out.println("Enter weights for layer " + (i + 1) + " to layer " + (i + 2) + ":");
            for (int j = 0; j < numNodes[i]; j++) {
                for (int k = 0; k < numNodes[i + 1]; k++) {
                    System.out.print("Weight from node " + (j + 1) + " to node " + (k + 1) + ": ");
                    weights[i][j][k] = scanner.nextInt();
                }
            }
        }
    }

    public int getWeight(int layer, int fromNode, int toNode) {
        return weights[layer - 1][fromNode - 1][toNode - 1];
    }
}

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

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

        System.out.print("Enter the num of layers: ");
        int numLayers = scanner.nextInt();

        int[] numNodes = new int[numLayers];
        for (int i = 0; i < numLayers; i++) {
            System.out.print("Enter the num of nodes in layer " + (i + 1) + ": ");
            numNodes[i] = scanner.nextInt();
        }

        network network = new network(numLayers, numNodes);
        network.setWeights(scanner);

        System.out.print("Enter the layer num: ");
        int layer = scanner.nextInt();

        System.out.print("Enter the 'from' node num: ");
        int fromNode = scanner.nextInt();

        System.out.print("Enter the 'to' node num: ");
        int toNode = scanner.nextInt();

        int weight = network.getWeight(layer, fromNode, toNode);
        System.out.println("Weight from node " + fromNode + " to node " + toNode + " in layer " + layer + ": " + weight);
    }
}
```

- Output:

```
Enter the num of layers: 3
Enter the num of nodes in layer 1: 2
Enter the num of nodes in layer 2: 3
Enter the num of nodes in layer 3: 4
Enter weights for layer 1 to layer 2:
Weight from node 1 to node 1: 1
Weight from node 1 to node 2: 2
Weight from node 1 to node 3: 3
Weight from node 2 to node 1: 4
Weight from node 2 to node 2: 5
Weight from node 2 to node 3: 6
Enter weights for layer 2 to layer 3:
Weight from node 1 to node 1: 7
Weight from node 1 to node 2: 8
Weight from node 1 to node 3: 9
Weight from node 1 to node 4: 10
Weight from node 2 to node 1: 11
Weight from node 2 to node 2: 12
Weight from node 2 to node 3: 13
Weight from node 2 to node 4: 14
Weight from node 3 to node 1: 15
Weight from node 3 to node 2: 16
Weight from node 3 to node 3: 17
Weight from node 3 to node 4: 18
Enter the layer num: 2
Enter the 'from' node num: 2
Enter the 'to' node num: 4
Weight from node 2 to node 4 in layer 2: 14
PS C:\Users\janei>
```

----------X-----X----------