Fun title for your project

ALGORITHMS AND DATA STRUCTURES
PRACTICAL 1

Group groupnr

Student1	s0000001
Student2	s0000002

This is a template for practical 1, all text in blue and the pseudocode example may be removed. The report may have at most 6 pages in the current format. The current sections and subsections must be present in the report, additional subsections may be added. Additionally, you may add examples, figures or pseudocode snippets (see Algorithm **??**).

1 Explanation of the Algorithm

1.1 Initialization

How do you read in the data and initialize the required data structures?

1.2 Algorithm Logic

Explain what your approach is, and what the algorithm does. Explain which data structures did you use and why. In case you adapt existing algorithms, explain how and why you modified and incorporated it.

Algorithm 1 Pseudocode example.

```
function FUNCTION(arguments)
    Step 1
    while Condition 1 does not hold do F
        if Condition 2 then
            return Yes
        else
            Step 2
    return No
```

1.3 Returning the Answer

When does the algorithm terminate and what is returned?

1.4 Optimizations

Did you use any special tricks to make the algorithm faster in special cases based on the problem statement?

2 Correctness Analysis

Why is your algorithm correct? Provide precise arguments, supported by formal reasoning whenever needed. Think of an expanation to a critical colleague why your algorithm is correct. For examples, take a look at correctness proofs in the textbook (Algorithms Illuminated) or solutions to the weekly exercises.

3 Complexity Analysis

What is the big O complexity of your algorithm (including reading in the data and making the data structure)? The analysis should be written in terms of the given input parameters. For example, if the input includes sets V of vertices and E of edges the complexity analysis should result in a big-Oh complexity using V and E. Include an explanation of your answer.

4 Reflection

Briefly reflect on the project. For example, you can write about how easy it was to find a suitable algorithm? Or if your algorithm is not always correct on DOMjudge, why do you think you get wrong answers? Or if there are of additional ways to speed up the algorithm?