University of Porto DCC-FCUP

Design and Analysis of Algorithms (CC2001) 2024/2025

Exercise Sheet 4

Programming practice – Mooshak contest DAA2425_C2

Graphs / BFS / DFS / Connected and Strongly connected components
Shortest path (w.r.t. number of edges) / Notion of topological order of DAG

Exercises for submission

(C2-A) Construção de mapa / Map Construction (weight 10%) **(main exercises)**
(C2-B) Halloween (weight 30%) **(main exercises)**
(C2-C) Pai Natal insuflável com escada / Santa Claus with ladder (weight 30%) **(main exercises)**
(C2-D) Ilhas e ilhéus / Islands and Islets (weight 30%) **(main exercises)**
(C2-E) Linhas Coloridas / Color Lines (weight 25%)
(C2-F) Sociologia / Sociology (weight 35%)
(C2-G) Quantas faltam no mínimo? / How many remain at least? (weight 35%)

Submission Deadline: March, 29 (submit to DAA's Mooshak)
(The exercises will be available for submission after that deadline, but do not count towards your grade)

You are encouraged to talk to the lecturers and to your colleagues if you have difficulties. However, any more direct help you have received from other colleagues or AI tools, like ChatGPT, should be acknowledged in the comments of the code you submit.

There will be at least 8 problem contests for classes during the semester, each one is worth at most 12.5 per cent of the grade for the exercise component. You can get full mark in that component (i.e., 1 point at 20) even if you haven't done everything, but recall that **experience in problem solving and programming** is crucial in the practical tests of DAA and for your background. For a problem to count, you have to get all the tests right (i.e. be Accepted). The maximum for each problem set is 100 per cent, even if you solve all the problems. It is guaranteed that the main exercises on each contest will sum to at least 100% of its grade.

In this class, you should use the code given in file **DAA2425_C2_code.tgz** to support your implementation.

1. Problem A -"Construção de mapa"

Reconstruct a **directed graph** from a sequence of routes.

Analyse the graph data structure defined in the examples given in **DAA2425_C2_code.tgz** and change the implementation to be able to read and process the input in a different format. Note also that the same edge can be used in different routes, but must be added just once to the graph.

2. Problem B -"Halloween"

In this problem, the nodes of the graph $G = (V, E)$ will keep a value, that is the *number of pumpkins*.

It must be solved in $\mathcal{O}(k(|V| + |E|))$ time, being k the number of queries. Use the BFS or DFS implementation given in **DAA2425_C2_code.tgz**, with the necessary adjustments (for input reading and search).

3. Problem C -"Pai Natal insuflável com escada"

This problem must be solved in $\mathcal{O}(|V| + |E|)$ time, by BFS. Use the code in **DAA2425_C2_code.tgz** as a basis, and modify it to find the **minimum distance** (minimum number of edges) from the source node to every other reachable node (if needed). Discard nodes that are too far away, in view of the constraints given.

4. Problem D -"Ilhas e ilhéus"

This problem must be solved in $\mathcal{O}(|V| + |E| + Q)$ time, based on BFS or DFS, being Q the number of queries.

In a pre-processing phase, we must find **the connected components**, starting from the node v that has the highest identifier.. All nodes in its component must be marked with value v. Then, we proceed in the same way for the nodes not yet visited.

After that, we are ready to answer each query in $\mathcal{O}(1)$

5. Problem E -"Color Lines"

Can be solved by BFS or DFS.

The game state (graph) must be represented by a **matrix**.

Cell (i, j) can have at most 4 neighbours, found in directions $(1, 0)$, $(0, 1)$, $(-1, 0)$, $(0, -1)$.

6. Problem F - "Sociologia"

This problem must be solved in $\mathcal{O}(|V| + |E|)$.

The graph is **directed**.

We must find its **strongly connected components**.

Start by writing a function to compute the **transpose** of the graph.

Adapt algorithm by Kosaraju-Sharir.

7. Problem G - "Quantas faltam no mínimo?"

The problem can be solved in $\mathcal{O}(|V| + |E|)$, either by BFS or DFS.

We can adapt the implementation of the data structure for graphs to support weighted directed graphs.

The edges of the graph are the tasks and define a **precedence relation** implicitly. We are looking for the set of tasks that will always be after **after** the given task in all **all topological orders** of the tasks. But, we do not need to compute a topological order.