**Descrição do Problema e da Solução**

Segundo o enunciado, o 2º projeto de ASA 2021/2022 consiste, dado um grafo dirigido e dois dos seus vértices, em determinar se o grafo forma uma árvore genealógica válida e, caso forme, o conjunto de ancestrais comuns mais próximos entre os dois vértices. A nossa solução, implementada em linguagem C++, recorre a uma classe personalizada composta por um inteiro (número de vértices), um vetor de vetores de inteiros (lista de adjacências) e uma matriz de inteiros (lista de adjacências do grafo transposto), para a representação do grafo dirigido.

A resposta ao primeiro problema é facilmente determinada. Por definição, apenas são impostas duas restrições para um grafo dirigido poder formar uma árvore genealógica: todo o vértice **(1)** tem, no máximo, dois progenitores e **(2)** não é progenitor de nenhum dos seus ancestrais. Traduzindo estas limitações para teoria de grafos, é simples compreender que, num grafo dirigido, **(1)** o número de progenitores se trata do grau de entrada de um vértice e **(2)** um arco de um nó para um dos seus ancestrais forma um caminho fechado no grafo, por outras palavras, forma um ciclo. Logo, um grafo dirigido constitui uma árvore genealógica se **(1)** nenhum dos seus vértices apresentar um grau de entrada superior a 2 e se **(2)** não contiver ciclos (DAG). Para garantir que nenhum vértice excede o grau de entrada 2, basta para isso monitorizar, na formação do grafo dirigido, cada arco adicionado e as respetivas atualizações dos graus de entrada dos seus vértices associados. Já para a deteção de ciclos é suficiente confirmar que, numa floresta DFS do grafo dirigido, não existem arcos para trás em nenhuma das suas árvores.

**Análise Teórica**

Análise teórica da complexidade total e das várias etapas da solução proposta.

Inserir aqui um pseudo código de muito alto nível a indicar a complexidade de cada etapa.

Exemplo:

* Leitura dos dados de entrada: simples leitura do input, com ciclo(s) a depender de linearmente/quadraticamente/… de V/E/V+E/… Logo, Θ(V)
* Processamento da instância para fazer alguma coisa. Logo, O(??)
* Aplicação do algoritmo X para fazer algo. Logo, O(?X?X)
* Transformação dos dados com uma dada finalidade. O(?Y?Y?)
* Apresentação dos dados. O(???)

Complexidade global da solução: O(!??!)

**Avaliação Experimental dos Resultados**

Descrição do tipo experiências feitas e gráfico demonstrativo da avaliação de tempos associados.

Gerar pelo menos 10 instâncias (e indicar quais) de tamanho incremental e cálculo dos tempos para cada instância.

Gerar o gráfico do tempo (eixo do YYs) em função do tamanho da instância de entrada (eixo dos XXs) como exemplificado abaixo. Indicar a informação dos eixos.



Concluir se o gráfico gerado está concordante com a análise teórica prevista.