Trabalho Prático – Parte 2

Programação Funcional

Enunciado

Data Limite de Entrega: 4 de junho de 2023

1. Introdução

Este trabalho aborda a gestão e o escalonamento de horários e salas para a realização de exames, no âmbito de uma instituição de ensino superior.

Na primeira fase do trabalho foi solicitado o desenvolvimento de soluções que permitam a gestão e visualização da informação disponível. Esta informação é relativa aos alunos, às unidades curriculares consideradas e à inscrição de alunos nas diversas unidades curriculares.

Na segunda fase do trabalho é solicitado o desenvolvimento de soluções que permitam a proposta do escalonamento de horários e salas para a realização dos exames das diversas unidades curriculares.

2. Tarefas

Conceba um programa desenvolvido em Haskell, que permita a leitura de três ficheiros de texto com o formato disponibilizado na primeira fase do trabalho (ucs.txt, inscricoes.txt e listaalunos.txt), e que, recebendo do utilizador, através do terminal, dois valores:

- o número total de dias em que os exames podem ocorrer

- o número de salas disponíveis para exame (assumindo que se o número de salas é igual em todos os dias),

seja capaz de:

1. Criar um novo ficheiro que contenha um escalonamento dos exames (apenas para uma época de exames) para todas as UCs, considerando o dia e sala em que cada exame ocorrerá. Deverá ser apresentado um aviso no terminal caso o número de salas e dias disponíveis sejam insuficientes para acomodar todos os exames necessários, sendo que não poderá ocorrer mais que um exame na mesma sala no mesmo dia (5 valores).

2. Criar uma solução de escalonamento de exames que garanta a restrição de não existir mais do que um exame de UCs do mesmo ano no mesmo dia (2 valores).

3. Apresentar no ecrã (terminal) as incompatibilidades entre cada par de UCs. Por incompatibilidades entenda-se o número de alunos que estão inscritos a cada par de UCs. Por exemplo, um par de UCs que não apresente nenhum aluno inscrito simultaneamente em ambas terá uma incompatibilidade de zero. Por outro lado, um par de UCs que apresente um número elevado de alunos inscritos em ambas simultaneamente, terá um valor elevado de incompatibilidades (3 valores).

4. Adicionar no ficheiro que contém a proposta de escalonamento dos exames a informação relativa ao número total de incompatibilidades que essa solução apresenta, ou seja, o número de alunos que terão exame de mais de uma UC no mesmo dia (2 valores).

5. Apresentar uma solução de escalonamento de exames que considere a minimização das incompatibilidades, ou seja, que tente apresentar um escalonamento de exames que permita reduzir, tanto quando possível, o número de incompatibilidades (5 valores)

6. Apresentar uma solução de escalonamento de exames que considere que cada sala apresenta uma lotação limitada. O valor da lotação de cada sala deve ser pedido ao utilizador. A lotação das salas deverá ser tida em conta no novo escalonamento dos exames, considerando que um exame de uma UC que tenha mais alunos inscritos do que a lotação da respetiva sala, terá de ser dividido em duas salas, reduzindo, deste modo, o número total de exames que podem ocorrer no respetivo dia (3 valores).

Nota: Deverá ser considerado que os três ficheiros de input podem ser alterados em termos de dimensão dos dados (não de formato), sendo, portanto necessário, que o programa desenvolvido esteja preparado para lidar com um volume maior ou menor de informação.

Sugestão: A representação das incompatibilidades entre cada par de UCs pode ser feita através de um grafo, onde cada nó representa uma UC e as ligações entre os nós representam o número de alunos inscritos a ambas UCs. Em Haskell estes grafos podem ser representados através de uma matriz, ou uma lista de listas de valores, onde cada posição das listas corresponde a uma UC e cada valor corresponde ao número de inscritos a cada par de UCs. Deste modo, preenchendo estas listas com as contagens das inscrições a cada par de UCs, é depois possível iterar as listas e identificar as incompatibilidades necessárias.

Sugestão: A minimização das incompatibilidades pode ser feita através da implementação de uma heurística, por exemplo implementando um método iterativo que começará por identificar os pares de UCs que apresentam o maior número de incompatibilidades, alocando-as a dias diferentes enquanto tal for possível.

3. Ficheiros disponibilizados

Os ficheiros disponibilizados são os três ficheiros de texto com o formato disponibilizado na primeira fase do trabalho (ucs.txt, inscricoes.txt e listaalunos.txt), podendo estes ficheiros sofrer alterações ao nível da quantidade de dados (não no formato).

4. Instruções

Os trabalhos devem ser realizados por **grupos de até 3 elementos**.

A entrega do trabalho deverá ser feita pelo Moodle, até às 23h59 do dia 4 de junho de 2023. A entrega deverá consistir num único ficheiro ZIP, que incluirá:

- Ficheiros fonte do código desenvolvido
- Breve relatório em formato PDF que incluirá a identificação dos elementos do grupo (nome e número de aluno), e a explicação sucinta da solução proposta

Os trabalhos deverão ser demonstrados e explicados ao Professor das aulas PL durante a semana seguinte à entrega do trabalho, sendo que esta demonstração terá uma duração de 5 minutos por grupo.