Trabalho Prático – Parte 1

Programação Funcional

Enunciado

Data Limite de Entrega: 08 de novembro de 2024

1. Introdução

Após longas horas de estudo e várias noites sem dormir, e com a ajuda de uma ferramenta de análise e interpretação de texto, o Professor de arqueologia, Indiana Jobs, finalmente decifrou o código. O código que permite operar uma nave alienígena que a sua amiga, a arqueóloga Laila Croft, encontrou numa das suas expedições.

Agora só falta implementar esse código e executá-lo no computador de bordo da nave. Devido à exigente necessidade de robustez contra falhas, e como grandes adeptos que são de linguagens formais, e em especial de programação funcional, resolveram desenvolver uma solução em Haskell.

2. Tarefas

Ajude os nossos exploradores, concebendo **um programa em Haskell** que permita executar um conjunto de ações que podem ser usadas para operar a nave alienígena.

Considere que a **localização** de uma nave se representa através de uma tripla **(x, y, z)** que contém as **coordenadas** do local, em 3 dimensões.

Considere que o **estado** de uma nave se representa através de uma dupla **(localização, ligado),** onde **ligado** é um valor booleano que indica se a nave está ligada ou não.

Considere que um **movimento** da nave se representa através de uma tripla **(x, y, z)** que contém o número de posições que a nave se deverá mover, em cada uma das 3 dimensões.

Desenvolva as seguintes funções, incluindo as respetivas assinaturas.

1. **atualiza_acao** : que recebe uma dupla com a ação (ligar ou desligar) e o estado atual da nave, e devolve o estado atualizado após a aplicação dessa ação;

2. **move** : que recebe um movimento e o estado atual da nave, e devolve o estado atualizado após a aplicação desse movimento;

3. **move_lista** : que recebe uma lista de movimentos e o estado atual da nave, e devolve o estado atualizado após a aplicação de todos os movimentos da lista;

4. **move_varios** : que recebe uma **lista de naves** e uma lista de igual tamanho com os respetivos estados iniciais, e devolve uma lista de duplas **(estado_final, ID)**, em que o **ID** é um identificador da nave e o **estado_final** o estado atualizado da mesma nave após a aplicação dos respetivos movimentos. Cada elemento da **lista de naves** que entra como argumento nesta função é uma dupla, em que o primeiro elemento é uma lista de movimentos, e o segundo elemento, o **ID** da respetiva nave. lista de naves (movimentos,ID)

5. **verifica_embates** : que recebe um possível novo estado de uma nave, e uma lista de estados de todas as naves, e devolve um valor booleano que indica que o novo estado origina um embate com alguma das naves. Por embate entenda-se partilharem as mesmas coordenadas.

6. Crie uma nova versão, atualizada, da função **move_varios** que apenas move uma nave caso esse movimento não origine um embate.

NOTA: Na função Move_varios falta colocar a condição de eles so se mexerem se as naves tiverem ligadas

7. Represente os conceitos utilizados nestas funções através da definição de tipos de dados sinónimos, e atualize as funções e respetivas assinaturas de acordo.

Podem ser utilizadas as funções auxiliares consideradas necessárias para o desenvolvimento das funções solicitadas.

Só deve ser considerada para a implementação do trabalho a matéria abordada nas aulas PL até ao final da ficha PL2.

3. Instruções

Os trabalhos devem ser realizados por grupos de 3 elementos.

A entrega do trabalho deverá ser feita pelo portal InforEstudantes, até às 23h59 do dia 08 de novembro de 2024. 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.