Trabalho Prático

Programação Funcional CC / Tópicos de Matemática

LCC 1º ano (2006/2007)

O trabalho deve ser realizado em grupo (no máx. de 3 alunos) e terá que ser entregue na última semana de aulas (em horário a anunciar). O trabalho deve ser acompanhado de um relatório com a justificação das soluções apresentadas para os problemas propostos. A listagem do programa deverá ser um anexo do relatório. A alíneas de valorização não são obrigatórias. Das várias alíneas de valorização, são de particular interesse as alíneas 13 e 14.

Relações Binárias

Pretende-se que construa um programa em Haskell que implemente a noção de relação binária sobre um conjunto e que permita manipular e testar estas relações. Sugerimos que use o tipo Haskell `[a]` para implementar a noção de conjunto.

Vamos limitar o problema da representação de relações binárias às **relações binárias sobre conjuntos finitos**.

1. Defina um tipo de dados em Haskell que lhe permita representar a noção de *relação binária*.

2. Defina uma função que lhe permita testar se um determinado termo do tipo que definiu na alínea anterior é de facto uma relação binária bem construída (*bem formada*).

3. Declare constantes que representem relações binárias concretas. Por exemplo, as seguintes relações binárias sobre o conjunto $\{1, 2, 3, 4, 5, 6\}$:

   (a) $\{(1,1),(1,4),(2,3),(1,2),(4,2)\}$

   (b) $\{(1,1),(2,2),(2,3),(2,4),(3,2),(3,3),(3,4),(4,2),(4,3),(4,4),(5,5),(6,6)\}$

   (c) $\{(1,1),(1,2),(2,2),(2,3),(3,3),(3,2),(4,5),(4,4),(5,4),(6,6)\}$

   (d) $\{(1,1),(1,2),(2,2),(2,3),(3,2),(2,4),(3,3),(4,3),(4,4),(5,5),(6,6)\}$

   (e) $\{(1,1),(1,2),(1,3),(1,4),(1,5),(2,2),(2,3),(2,4),(2,5),(3,4),(3,5)\}$

   Use estas relações (e outras que lhe pareçam pertinentes) para testar o bom funcionamento das funções que lhe vamos propor.

4. Defina funções para testar se uma relação binária é:

   (a) *relexiva*

   (b) *simétrica*

   (c) *transitiva*

5. Defina uma função que teste se uma relação binária é uma *relação de equivalência*.

6. Defina uma função que, dados uma relação de equivalência sobre um conjunto e um elemento do conjunto, calcule a *classe de equivalência* desse elemento.

7. Defina uma função que calcule as classes de equivalência geradas por uma relação de equivalência.

8. Relembre a noção de *partição de um conjunto*.

   (a) Defina uma função que, dados um conjunto A e um conjunto de subconjuntos de A, teste se este é uma partição de A.

   (b) Defina uma função que, dada uma partição de um conjunto, construa a relação de equivalência induzida por essa partição.

9. Relembre a noção de *relação de ordem*.

   (a) Defina uma função para testar se uma relação binária é *anti-simétrica*.

   (b) Defina uma função para testar se uma relação binária é uma relação de ordem.

10. Defina uma função que receba duas relações binárias sobre um mesmo conjunto e calcule a *composição* das duas relações.

11. Defina uma função que, dada uma relação binária, calcule a sua relação *complementar*.

12. Defina funções para construir, separadamente, os seguintes *fechos* de uma relação:

    (a) *fecho reflexivo*

    (b) *fecho simétrico*

    (c) **(Valorização)** *fecho transitivo*

13. **(Valorização)** Lançamos agora o desafio de construir uma versão "avançada" das funções que definiu nas alineas 4 e 9a. O que se pretende é que estas funções de teste apresentem uma justificação (um contra-exemplo) no caso do teste em causa falhar. Assim,

    (a) Defina um novo tipo de dados que permita captar as situações de: sucesso, falha da reflexividade apresentando um contra-exemplo, falha de simetria apresentando um contra-exemplo, falha de anti-simetria apresentando um contra-exemplo e falha da transitividade apresentando um contra-exemplo.

    (b) Defina funções que testem, separadamente, a reflexividade, a simetria, a anti-simetria e a transitividade, de uma relação binária, e que apresentem um contra-exemplo em caso de falha.

    (c) Declare o tipo de dados que definiu na alínea (a) como instância da classe `Show`, de modo a que a apresentação do resultado das funções de teste da alínea anterior seja uma justificação, por palavras, do sucesso ou da falha do teste em causa.

14. **(Valorização)** Defina "versões avançadas" de funções que testem se uma relação binária

    (a) é uma relação de equivalência

    (b) é uma relação de ordem

    indicando, em caso de falha, a/as propriedades que não se verificam.

15. **(Valorização)** Manipulação de ficheiros.

    (a) Defina uma função que permita guardar uma relação binária num ficheiro.

    (b) Defina uma função que permita carregar uma relação binária a partir de um ficheiro.

16. **(Valorização)** Construa uma aplicação interactiva, com menus, que integre as funções que definiu, e que permita definir relações binárias de forma interactiva (ou ler as relações a partir de ficheiros) e depois testar as suas propriedades.