Aula Teórico-Prática 2

Programação Funcional

LEI 1º ano

1. Defina uma função `max2` que calcula o maior de dois números inteiros. Comece por definir a assinatura da função.

2. Defina uma função que calcula o maior de três números inteiros. Para isso apresente duas definições alternativas: recorrendo ou não à função `max2` definida na alínea anterior.

3. Utilisando as funções `ord::Char->Int` e `chr::Int->Char` defina as seguintes funções:

   (a) `isLower :: Char -> Bool`
   (b) `isDigit :: Char -> Bool`
   (c) `isAlpha :: Char -> Bool`
   (d) `toUpper :: Char -> Char`
   (e) `intToDigit :: Int -> Char`
   (f) `digitToInt :: Char -> Int`

   Obs: todas estas funções já estão definidas no módulo `Char`.

4. Num triângulo verifica-se sempre que a soma dos comprimentos de dois dos lados é superior (ou igual) à do terceiro. A esta propriedade chama-se *desigualdade triangular*. Defina uma função que, dados três números, teste se esses números correspondem aos comprimentos dos lados de um triângulo.

5. Vamos representar um ponto por um par de números que representam as suas coordenadas no plano cartesiano.

   ```
   type Ponto = (Float,Float)
   ```

   (a) Defina uma função que recebe 3 pontos e verifica se esses ponto constituem os vértices de um triângulo.
   (b) Defina uma função que recebe 3 pontos que são os vértices de um triângulo e devolve um tuplo com o comprimento dos seus lados.

6. Defina uma função que recebe os (3) coeficientes de um polinómio de 2º grau e que calcula o número de raízes (reais) desse polinómio.

7. Usando a função anterior, defina uma função que, dados os coeficientes de um polinómio de 2º grau, calcula a lista das suas raízes reais.

8. As funções das duas alíneas anteriores podem receber um tuplo com os coeficientes do polinómio, ou receber os 3 coeficientes separadamente. Defina a versão alternativa ao que definiu acima.

9. Vamos representar horas por um par de números inteiros:

   ```
   type Hora = (Int,Int)
   ```

   Assim o par `(0,15)` significa *meia noite e um quarto* e `(13,45)` *duas menos um quarto*. Defina funções para:

   (a) testar se um par de inteiros representa uma hora do dia válida;
   (b) testar se uma hora é ou não depois de outra (comparação);

(c) converter um valor em horas (par de inteiros) para minutos (inteiro);

(d) converter um valor em minutos para horas;

(e) calcular a diferença entre duas horas (cujo resultado deve ser o número de minutos)

(f) adicionar um determinado número de minutos a uma dada hora.

10. Analise a seguinte definição e apresente uma definição alternativa que use concordância de padrões em vez dos *if*s.

```
opp :: (Int,(Int,Int)) -> Int
opp z = if ((fst z) == 1)
        then (fst (snd z)) + (snd (snd z))
        else if ((fst z) == 2)
             then (fst (snd z)) - (snd (snd z))
             else 0
```