PFL_TP1_G08_01

Representação interna de polinómios

Neste trabalho, decidimos utilizar uma lista de tuplos para representar os polinómios. Cada tuplo representa um monómio e é constituído por 2 elementos: o primeiro corresponde ao coeficiente (negativo ou positivo) do monómio e o segundo, à parte literal do mesmo. É ainda de referir que, o 2º elemento de cada tuplo é novamente uma lista de tuplos. Estes tuplos correspondem a cada variável que integre o monómio, associada ao seu expoente.

- Polynomial = [Monomial]
- Monomial = (Int, Vars)
- Vars = [(Char, Int)]

Justificação

Com esta representação, conseguimos facilmente alcançar as variáveis e os respetivos expoentes, acedendo ao primeiro e segundo elemento dos tuplos, com chamadas recursivas das funções, de modo a iterar a lista de `Vars`. Da mesma forma, conseguimos aceder a cada monómio que constitui o polinómio, iterando a lista de monómios.

Funcionalidades

Normalização

Uma vez que esta funcionalidade pode implicar diversos pormenores e modificações (dependendo do polinómio introduzido), decidimos criar uma função para cada uma delas e chamá-las na função principal que aplica todas as alterações, `normalizePolynomial :: String -> String`.

1. A função recebe um polinómio ainda no formato de string, tal como o utilizador o inserir e, de seguida, transforma a string no tipo `Polynomial` que criámos
2. Se a variável do tipo `Polynomial` estiver vazia,é retornada a mesma variável
3. Se não estiver vazia, em primeiro lugar chama-se a função que soma os coeficientes associados a partes literais iguais (ex: `2*x^2 + 3*x^2 = 5*x^2`)
4. De seguida, é chamada uma função, que recursivamente, para cada monómio, verifica se alguma das variáveis pode ser junta com outra, somando os seus expoentes (ex: `2*x^2x^3 = 2*x^5`)
5. Por fim, o polinómio é ordenado e convertido de novo numa String

Adição

Para esta funcionalidade aproveitámos a função que criámos para a funcionalidade anterior, apenas concatenando os 2 polinómios a somar na função principal `addMonomials :: Monomial -> Monomial -> Monomial`, previamente a chamar a função de normalização, uma vez que ao fazê-lo, obtem-se um polinómio que pode ser normalizado.

1. A função recebe dois polinómios ainda no formato de string, tal como o utilizador os inserir e transforma-os no tipo `Polynomial`
2. Se algum dos polinómios inseridos estiver vazio, o output será o polinómio que não for nulo ou. Se ambos forem nulos, o output é um polinómio nulo
3. Se não, a soma resulta da chamada da função de normalização, pelas razões já mencionadas.

Multiplicação

Para a multiplicação de polinómios, decidimos percorrer recursivamente a lista de monómios do primeiro polinómio inserido e multiplicar cada um por todos os monómios do segundo polinómio.

1. A função recebe dois polinómios ainda no formato de string, tal como o utilizador os inserir e transforma-os no tipo `Polynomial`
2. Se algum ou ambos os polinómios inseridos estiverem vazios, o output será um polinómio nulo.
3. Se não, aplica-se a função que multiplica o primeiro monómio do primeiro polinómio pelo segundo polinómio, concatenando-se esse resultado com a chamada recursiva da função principal para os restantes monómios do primeiro polinómio.

Derivação

Por fim, para a derivação, criámos também uma função para derivar cada monómio e, como auxiliares, uma função para verificar se a variável pela qual se pretende derivar o polinómio está presente em cada monómio e uma função para baixar os expoentes das variáveis aquando da derivação. Todas elas acabam por ser chamadas com a chamada da função principal `derivPoly :: Polynomial -> Char -> String`.

1. A função recebe um polinómio e um caracter que representa a variável pela qual se quer derivar o polinómio.
2. Se o polinómio for nulo, é retornado também um polinómio nulo
3. Se não, o output resulta da derivação e posterior normalização do mesmo.

Exemplos de utilização das funcionalidades

Normalização

`0*y^2 + 4*x*x^3 + 2*x^4`
`2*y^2 - 4*y^2 + 2*y^0 + 4*y^1*`
`y^0x^0 + 1 + 2*x^2 + x^3`

Adição

`4*x + 3*x^2 + 2*x^3` e `x^3 - x^2 + x`
`x + 2*x` e `-2x - x`
`2*x^2 + 2*y^2` e `2*xx + 2*z^2`

Multiplicação

`2*x^2` e `4*x + 4*yz`
`2*x^2 + x + 1` e `4x + y + 2*y^2`
`x - 1` e `-x + 1`

Derivação

`4*x^2 + x + y + 1` e `x`
`2*y + z` e `x`
`-x^2 + x + 1` e `x`