Problem Set 1 - Haskell

Computação Quântica 2018/2019

February 5, 2019

1. Write an Haskell function that:
   (a) Calculates the perimeter of a circle given its radius.

```
perimeter_circle :: Float -> Float
```

   (b) Calculates the area of a circle given its radius.

```
area_circle :: Float -> Float
```

2. Write an Haskell function to calculate the factorial of a number n. Remember that by convention, $fac(0) = 1$.

```
factorial :: Int -> Int
```

3. Prime numbers can be very useful for many purposes such as cryptography.
   (a) Write a function that returns all numbers from 2 to a given number n.

```
all_numbers_up_to_n :: Int -> [Int]
```

   (b) Implement a function to eliminate the multiples of a number n from a list

```
eliminate_multiples :: Int -> [Int] -> [Int]
```

   (c) Admit the Erastothenes sieving algorithm: starting by a list of numbers from 2..n, the algorithm iteratively takes out the multiples of the prime elements. In each iteration the prime element to use always lies in the next position of the element used in the previous iteration.

```
Initial list: [2, 3, 4, 5, 6, 7, 8, 9, 10]
Iteration 1: [2, 3, 5, 7, 9] // take out the multiples of 2
Iteration 2: [2, 3, 5, 7] // take out the multiples of 3
Iteration 3: [2, 3, 5, 7] // take out the multiples of 5.
```

```
(...)
Iteration n: [2, 3, 5, 7] // in the end only the primes remain
```

Implement a function that returns all primes up to a number n.

```
prime_list :: Int -> [Int]
```

(d) Implement a function that tests the primality of a given number n.

```
is_prime :: Int -> Bool
```

(e) Implement a function to factorize a number into prime factors, based on the previous functions. Example: $15 = 3 \times 5$.

```
factorize :: Int -> [Int]
```

4. Matrices are an invaluable tool in quantum information. Here we will try to implement some of the primitive operations involving matrices. Consider that the definition for matrices in Haskell is given by lists of lists `[[Int]]`. For instance, the representation of the Pauli X matrix reads as follows:

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \mapsto \texttt{[[0,1],[1,0]]}.$$

Implement functions to:

(a) Calculate the transpose of a matrix A^T

(b) Multiply matrices $A.B$

(c) Calculate the tensor product $A \otimes B$

(d) Calculate projection matrices $A \otimes A^T$

5. An n-qubit quantum state can be represented by a column vector with 2^n entries. Implement a function that takes a vector representing a 2-qubit state and applies a CNOT operation to it. Note that a CNOT can be represented by a 4×4 matrix:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$