Aula Prática 2

Instruções:

1 - Os exercícios práticos devem ser realizados individualmente e enviados por e-mail com o assunto [IF686EC] AP 2 para monitoria-if686-ec-l@cin.ufpe.br até as 23:59 de sexta-feira (05.07.2018).

2 - As resoluções dos exercícios devem estar em arquivos diferentes (não compactar), um arquivo por exercício com os nomes no formato Q[número da questão].hs.

3 - O arquivo com a resposta de cada questão deve conter a função solicitada no formato dado em negrito após o enunciado de questão. Os tipos de entrada e saída explicitados, assim como o nome da função, devem ser respeitados.

1) Uma linguagem de programação baseada em pilha possui apenas uma pilha (stack) onde ficam os dados/operandos e todas as instruções são apenas de empilhar, desempilhar ou fazer operações consumindo (lendo) os dados no topo da pilha e deixando o resultado final o topo da pilha.
Dados os tipos de dados abaixo e os exemplos, escreva um interpretador que executa as instruções com o comportamento abaixo:

```
data Instrucao = PUSH Int -- empilha um valor inteiro
               | POP -- desempilha (remove) um valor do topo da pilha
               | ADD -- remove (lê) os dois valores no topo da pilha e deixa a soma deles no topo da pilha.
               | SUB -- remove (lê) os dois valores no topo da pilha e deixa a soma deles no topo da pilha.
               | DUP -- repete o mesmo valor no topo da pilha (duplica ele)

type Pilha = [Int]
```

2) Um robô é controlado por 4 comandos:

1. Left, para girar sua direção à esquerda 90 graus;
2. Right, para girar sua direção à direita em 90 graus;
3. Forward seguido de um número N, que indica um avanço de N metros.
4. Backward seguido de um número N, que indica um retrocesso de N metros.

Supondo que o robô comece na posição (0,0) (coordenadas) e direcionado para norte (i.e. para o posição (0,1)), faça uma função destination que informe a localização do robô após uma sequência de comandos.

[BÔNUS] Determine o tipo das funções abaixo mostrando os passos até obter o
resultado. Caso não seja possível determinar o tipo, explique por quê.

a) (.) thrice map

b) swap map thrice

c) tail . head

Dados:

```
( . ) :: (b -> c) -> (a -> b) -> a -> c
map :: (a -> b) -> [a] -> [b]
thrice :: (a -> a) -> a -> a
thrice f x = f (f (f x))
swap :: a -> (a -> b) -> b
swap f g = g f
```