Diversão com Robôs

Setup inicial

```haskell
module Semana11 where

import Parser
```

Com o início da exploração de petróleo no oceano, diversas empresas tem interesse em explorar eventuais reservas de minerais no subsolo marítmo. Você foi contratado por uma destas empresas para desenvolver uma pequena linguagem de programação para o controle de um robô que irá realizar a exploração do fundo do oceano. Neste conjunto de exercícios, você deverá desenvolver um conjunto de funções para implementar esta pequena linguagem para controlar este robô explorador.

Por questões de simplicidade, representaremos o robô utilizando o seguinte tipo de dados Haskell:

```haskell
type Fuel = Int

type Distance = Int

type Position = (Int, Int)

data Robot
   = Robot {
      energy :: Fuel,          -- actual energy
      distance :: Distance,    -- distance travelled
      current :: Position      -- current position
   } deriving (Eq, Ord)
```

onde os sinônimos `Fuel`, `Distance` e `Position` representam o valor atual de energia, a distância percorrida e a posição atual do robô. Como exemplo, considere a seguinte representação de um robô que se encontra na posição (0,0), possui 10 unidades de energia e percorreu uma distância de 5 unidades.

```haskell
example :: Robot
example = Robot {
            energy = 10,
            distance = 5,
            current = (0,0)
          }
```

Posições são representadas por pares ordenados no plano cartesiano. O mapa a ser considerado pelo robô é definido por um retângulo no plano cujos extremos são a origem (isto é, o ponto (0,0)) e um ponto (x,y), onde $x, y \geq 0$. Uma representação de exemplo de um mapa de tamanho 2×2 unidades 'e mostrada na figura a seguir.

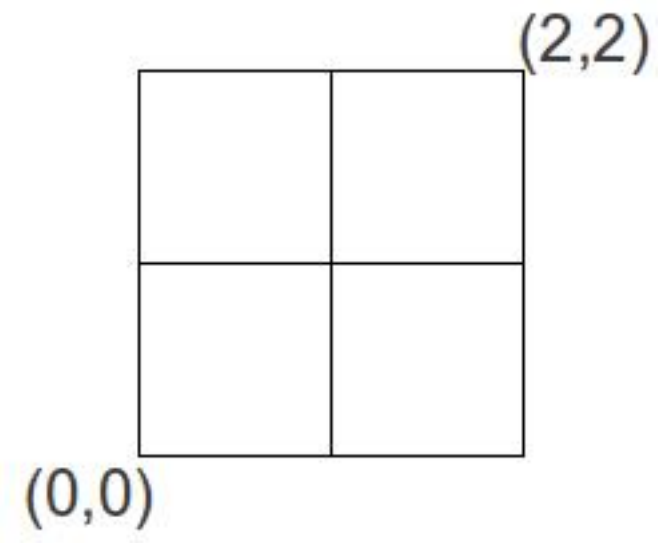

Figure 1: Mapa de exemplo

A linguagem de comandos para o robô possui apenas 6 comandos. Sendo quatro para locomoção (deslocar o robô uma unidade para o norte, sul, leste ou oeste), um para recarga de energia e um para coleta de materiais. Instruções podem ser representadas pelo seguinte tipo de dados Haskell:

```haskell
data Instr
    = North       -- walk north
    | South       -- walk south
    | East        -- walk east
    | West        -- walk west
    | Collect     -- collect material
    | Recharge    -- stay and recharge
    deriving (Eq, Ord)
```

As instruções de locomoção fazem com que o robô se desloque uma unidade na direção especificada. Cada instrução de locomoção consome uma quantidade de 5 unidades de energia e incrementa por 1 a distância percorrida. A instrução de coleta consome 10 unidades de energia e a instrução de recarga obtêm 20 unidades de energia. Nenhuma destas duas últimas instruções altera a distância percorrida pelo robô. Cada um dos comandos do robô são representados apenas por uma letra, conforme a tabela seguinte.

Comando	Constructor Haskell	Mneumônico
Mova para o norte	North	N
Mova para o sul	South	S
Mova para o leste (esquerda)	East	E

Mova para o oeste (direita)	West	W
Colete material	Collect	C
Recarregar	Recharge	R

Um programa completo para o robô especifica o limite superior do mapa (limite superior do retângulo no plano), a posição inicial do robô no plano e uma sequência de instruções a serem executadas. Um programa é representado pelo seguinte tipo de dados Haskell:

```haskell
data Program
    = Program {
        limit :: Position, -- extreme right map position
        init  :: Position, -- initial position of the robot
        instrs :: [Instr]  -- the program
    } deriving (Eq, Ord, Show)
```

Textualmente, programas são especificados de acordo com a seguinte gramática apresentada a seguir.

$$
\begin{array}{rcl}
\textit{programa} & \rightarrow & \textit{posição posição comandos} \\
\textit{posição} & \rightarrow & \texttt{[}\textit{inteiro}\texttt{,}\textit{inteiro}\texttt{]} \\
\textit{comandos} & \rightarrow & \textit{comando comandos} \\
 & | & \lambda \\
\textit{comando} & \rightarrow & \texttt{N} \mid \texttt{S} \mid \texttt{E} \mid \texttt{W} \mid \texttt{R} \mid \texttt{C}
\end{array}
$$

Note que entre cada um dos elementos da gramática não devem haver espaços. A seguinte string, constitui um programa nesta linguagem:

```
[20,20][5,3]NNWWSSEECNNCNN
```

em que o ponto que delimita o mapa 'e $(20, 20)$, o robô inicia na posição $(5, 3)$ e a sequência de comandos executados pelo robô 'e NNWWSSEECNNCNN.

Com base no que foi apresentado, desenvolva o que se pede.

1. Implemente instâncias da classe Show para os tipos de dados Robot e Instr. A instância de Show para o tipo Robot deve ser tal que para o valor:

```haskell
example = Robot {
              energy = 10,
              distance = 5,
              current = (0,0)
          }
```

a seguinte string deve ser produzida: `"Energy:10\nDistance:5\nPosition:\n\tx:0\n\ty:0"`. Por sua vez, para o tipo Instr, você deverá produzir strings correspondentes aos mnemônicos de cada instrução.

2. Implemente um parser para a gramática de programas para o robô. Seu parser deve retornar como resultado um valor do tipo de dados Program.

```haskell
parseProgram :: Parser Char Program
parseProgram = undefined
```

3. Implemente uma função que dada uma lista de instruções e a configuração inicial de um robô, retorne a configuração do robô após a execução de todas as intruções.

```haskell
runRobot :: [Instr] -> Robot -> Robot
runRobot = undefined
```

Dica: A maneira mais simples de se implementar esta função é utilizando a mônada de estado.