Faculdade de Computação
Programação Funcional (BCC/BSI) - 1° Período

Aula Prática: Ordenação II

1) Sejam as funções *minimo* e *menor* para retornar o menor elemento de uma lista:

```
minimo::(Ord a) => [a]->a
minimo [] = undefined
minimo [x] = x
minimo (x:xs)
  | x <= (minimo xs) = x
  | otherwise = minimo xs

menor::(Ord a) => [a]->a
menor [] = undefined
menor [a] = a
menor (a:b:as)
   | (a<b) = menor (a:as)
   | otherwise = menor (b:as)
```

Na ordenação por seleção, os seguintes passos devem ser executados:

1. Encontrar o menor elemento da lista.
2. Inserir o menor elemento encontrado numa nova lista, e removê-lo da lista de origem.
3. Repetir as ações 1 e 2 acima até que a lista de origem seja vazia.

Dada a função abaixo para ordenação por seleção, implemente a função remove:

```
selecao:: (Ord a) => [a]->[a]
selecao [] = []
selecao xs = [x] ++ selecao (remove x xs)
                    where x = minimo xs
```

Execute passo a passo as chamadas:

```
> selecao [2,0,1,-2,4,5,3]

> selecao "cajbfckadle"

> selecao [("Marcos",13),("Ana",56),("Lucas",34),("Jose",26),("Pedro",10),
("Roberto",22),("Claudia",32)]

> selecao [[6,9],[2,13],[6,8,14,9],[10,7],[5]]
```

2) O método de ordenação por bolha é bastante conhecido e baseia-se em trocas de elementos que se encontram em posições consecutivas:

> *"A ideia é percorrer o vetor diversas vezes, a cada passagem fazendo flutuar para o topo o maior elemento da sequência. Essa movimentação lembra a forma como as bolhas em um tanque de água procuram seu próprio nível, e daí vem o nome do algoritmo."* *(http://pt.wikipedia.org/wiki/Bubble_sort)*

Exemplo:
Lista Inicial: [3,7,2,6,1,4]

1ª Troca: [3,2,6,1,4,7]
2ª Troca: [2,3,1,4,6,7]
3ª Troca: [2,1,3,4,6,7]
4ª Troca: [1,2,3,4,6,7]

a) A função de ordenação por Bolha pode ser implementada utilizando-se recursão em cauda. Assim, implemente as funções *bolhaOrd* e *troca* para a função *bolha* abaixo:

```
bolha [] = []
bolha lista = bolhaOrd lista (length lista)
```

b) Faça a implementação de outra função de ordenação por Bolha, usando a função genérica *iterate* e a mesma função *troca* do item (a):

```
iterate :: (a -> a) -> a -> [a]
A função iterate recebe uma função e um valor inicial. Em seguida,
aplica a função ao valor inicial e continua aplicando a função ao
resultado obtido, e assim por diante.

Exemplos:
> iterate (+1) 1
[1,2,3,4,5,6,7,8,9,10 …
> take 5 (iterate (+1) 1)
[1,2,3,4,5]
```

c) Como podemos implementar a função *iterate*?