Introdução à Programação I, 28 de Janeiro de 2002

Duração máxima 2 horas e 30 minutos; Com consulta

Nome (legível): _____ **Número** _____

Problema 1 **(7.5 valores)**

Uma ideia que não parece muito eficiente para ordenar os elementos de uma lista, supostos valores numéricos, consiste no seguinte:

- identificar o valor mais pequeno da lista a ordenar;
- criar uma lista ordenada, juntando (como primeiro elemento) o valor identificado à lista composta pelos restantes valores, que também serão ordenados...

1.1 Complete o procedimento ***retira-elemento***.

Exemplos:

```
> (retira-elem 1 (list 3 1 78 3 9))
(3 78 3 9)
> (retira-elem 15 (list 3 1 78 3 9))
(3 1 78 3 9)

; devolve uma lista composta por todos os elementos de lis, à qual se retira a
; primeira ocorrência de elem, se este existir em lis. Se não existir, devolve lis.
(define retira-elemento
  (lambda (elem lis)
```

1.2 Complete o procedimento ***ordena-em-ascendente***, sabendo que se baseia na ideia acima exposta e que utiliza o procedimento ***retira-elemento***.

Exemplos:

```
> (ordena-em-ascendente (list 13 10 5 7 8 9 10))
(5 7 8 9 10 10 13)
> (ordena-em-ascendente (list))
()
(define ordena-em-ascendente
  (lambda (lis)
    (if (null? lis)
        '()
        (let ((val-min (apply min lis)))   ; identifica o valor mais pequeno
```

Problema 2 — Parte A (7.5 valores)

Há uns anos atrás encontrava-se com alguma frequência o *Jogo dos Furos*, associado à venda, por exemplo, de chocolates. A base do jogo era uma folha de cartolina, com pequenos furos organizados em matriz, escondidos com uma folha de papel colada à cartolina; apesar de escondidos, a posição dos furos era bem visível, o que já não acontecia com a pequena esfera colorida, que existia em cada um deles. As crianças (e não só), munidas de algumas moedas, "faziam furos", pagando por cada um deles uma certa quantia. Pela parte da frente da cartolina, empurravam com um ponteiro ou lápis, sobre os furos pretendidos. Do outro lado, por cada furo, aparecia uma esfera, cuja cor definia o prémio a que tinham direito.
Vamos supor os seguintes códigos de cores:

Vermelho — Caixa de chocolates grande

Azul — Caixa de chocolates média

Verde — Chocolate grande

Amarelo — Chocolate médio

Castanho — Chocolate (muito) pequeno.

Como é fácil de imaginar, a cor que saía com maior frequência era o castanho e, muito raramente, o vermelho... Já agora, como curiosidade, o aliciante para que o jogo se completasse era associar ao último furo, para além do prémio respectivo, o mostruário de prémios, com um exemplar de cada prémio.

O programa que se pretende desenvolver é uma recriação do *Jogo dos Furos*, adaptando-o adequadamente.
Vamos imaginar uma sessão de utilização deste programa:

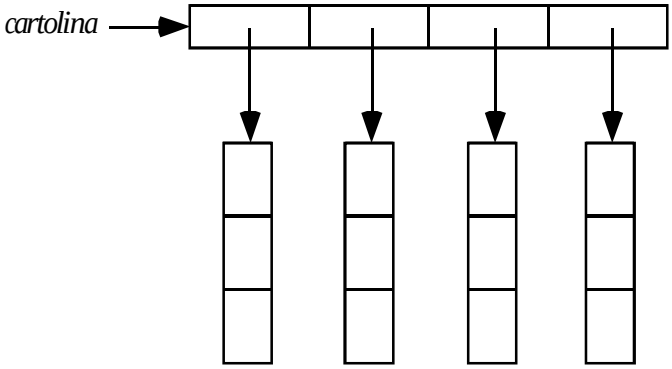
```
> (jogo-dos-furos 10 8)       (escolha de uma cartolina de 8 linhas de 10 colunas)
1  2  3  4  5  6  7  8  9  10
11  12  13  14  15  16  17  18  19  20
21  22  23  24  25  26  27  28  29  30
31  32  33  34  35  36  37  38  39  40
41  42 ...
Furo: 4
verde - Chocolate grande

1  2  3  5  6  7  8  9  10
11  12  13  14  15  16  17  18  19  20
21  22  23  24  25  26  27  28  29  30
31  32  33  34  35  36  37  38  39  40
41  42 ...
Furo: 4                              (tentativa de furar um número já furado...)
Furo: 29
castanho - Chocolate pequeno

Após muitos furos...

21
42 ...
Furo: 42
castanho - Chocolate pequeno

21
Furo: 21
amarelo - Chocolate medio
E ainda uma coleccao de premios.
```

Vamos supor que cada cartolina está organiza em linhas e colunas e que a posição dos prémios é aleatória. O número de bolas coloridas tem a seguinte distribuição: castanho — 50%; amarelo — 25%; verde — 13%; azul — 7%; vermelho — as restantes.

Imagine que é dada a abstracção ***jogo de furos***, com:

- ***faz-jogo-furos***, um construtor com os parâmetros ***col*** e ***lin***, dois inteiros que definem, respectivamente, o número de colunas e linhas de uma *cartolina* correspondente a um jogo de ***col*** x ***lin*** furos, preenchido nas proporções acima referidas.
- ***sel-jogo-furos***, um selector com os parâmetros ***jogo***, ***col*** e ***lin***, que devolve o código associado à esfera escondida no furo da cartolina ***jogo***, situado na linha ***lin*** e na coluna ***col***. Se a célula já tiver sido "furada", devolve um código adequado.
- ***mod-jogo-furos!***, um modificador com os parâmetros ***jogo***, ***col***, ***lin*** e ***codigo***, que altera o valor do furo da cartolina ***jogo***, situado na linha ***lin*** e na coluna ***col***, para o código ***codigo***.
- ***visu-jogo-furos***, um selector com o parâmetro ***jogo***, que visualiza o número dos furos ainda não escolhidos, no formato indicado na sessão de utilização do programa anteriormente apresentado. Devolve o número de furos que ainda estão para furar.

Imagine que a estrutura de dados mais adequada para a entidade *cartolina* da abstracção do *jogo de furos* é um vector de vectores. Por exemplo, para uma *cartolina* de 4 colunas por 3 linhas, a estrutura teria, graficamente, o seguinte aspecto:



Cada elemento dos vectores representados verticalmente poderá assumir os seguintes valores: vazio, castanho, amarelo, verde, azul, vermelho, furado.

2.1 Escreva em Scheme o selector ***sel-jogo-furos***.

2.2 Escreva em Scheme o modificador ***mod-jogo-furos!***.

Problema 2 — Parte B (5.0 valores)

2.3 Delineando, em *Scheme*, o corpo principal do programa, identifique os principais procedimentos em que ele se organiza. Bastará indicar OS NOMES desses procedimentos e os seus PARÂMETROS, e explicar o OBJECTIVO de cada um deles.

2.4 Escrever em *Scheme* o selector ***visu-jogo-furos***.

(Fim.)