Lista de Exercícios 7 – INF05008

Siga as instruções sobre elaboração de exercícios de INF05008.

Nesta lista, para todas as **funções recursivas** você deve incluir modelo da solução. Esse modelo, em forma de comentários, deve explicitar como o algoritmo vai funcionar para o(s) caso(s) base e passo(s) da definição da função recursiva. Este modelo deve ter o seguinte formato, e pode ser colocado antes ou permeado ao código:

```
;;  SE < ...é o caso base da def. de lista... > ENTÃO < ...resolver o problema ... >
;;  SE < ... não é o caso base da def. de lista... >  <=== pode haver mais de um desses casos, e
                                                            pode-se usar SENÃO, se for o último caso
;;     ENTÃO  < ...combinar soluções... >
;;              < ...fazer algo com... >  < o primeiro elemento da lista >
;;              < ...solucionar o problema para... >  < o resto da lista >
```

Você pode usar as seguintes funções pré-definidas que envolvem listas: `empty`, `cons`, `empty?`, `first` e `rest`. Não use outras funções, pois o objetivo é aprender a trabalhar com listas. Além destes, nas definições de constantes do tipo lista, pode usar o construtor `list` ao invés de `cons`, como no exemplo abaixo. Mas não use `list` na construção dos programas (nesta lista de exercícios). Usar `list` nos dá uma forma mais concisa para descrever listas, mas internamente elas são construídas sempre com `cons` e `empty`:

a lista `(cons 1 (cons 2 (cons 3 empty)))` pode ser representada por `(list 1 2 3)`

Abra o template da lista 7 e use as definições de dados dos conjuntos `Retangulo`, `Triangulo`, `Estrela`, `Forma` e `Figura` do template. Além destes tipos, você deve definir todos outros tipos que forem necessários para resolver os exercícios (decidir quais tipos são necessários e defini-los faz parte da solução do problema).

1. Defina um tipo de dado chamado `ListaDeFormas`, que contém todas as possíveis listas (finitas) de formas que podem ser retângulo, triângulo e estrelas, conforme as definições do template. Dê, pelo menos, 5 exemplos de listas de formas, sendo que pelo menos 2 listas devem conter no mínimo 5 elementos. Defina, também, 2 constantes do tipo `Figura`.

2. Construa a função `quantas-vezes` que, dados um dada uma lista de strings e uma palavra (string), nesta ordem, devolve quantas vezes a palavra aparece na lista.

3. Defina a função `lista-cores` que, dada uma lista de formas, devolve uma lista com todas as cores das formas da lista, na ordem na qual essas cores aparecem nas formas da lista original (se houver mais de uma forma com a mesma cor, essa cor aparecerá mais de uma vez na lista).

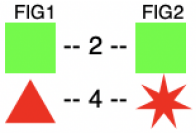
4. Desenvolva uma função, `desenha-lista-formas`, que, dada uma lista de formas, monta uma imagem com todas as formas lado a lado. Para as estrelas, usar 3 para o intervalo de conexão (ver no template o modo de usar a função pré-definida `star-polygon`, que desenha estrelas).

5. Defina a função chamada `lista-formas-cor` que, dados uma lista de formas e uma cor, nesta ordem, devolve uma lista com formas desta cor da lista original. Se não houver nenhuma forma da cor, devolver a mensagem `"Nenhuma forma encontrada."`. *Obs.: Cuidado com o tipo da saída.*

6. Construa a função `remove-repetidas` que, dada uma lista de strings, elimina strings repetidas na lista seguindo a regra: sempre que houver strings repetidas, somente a última string deve ser mantida na lista.

7. Desenvolva a função `seleciona-formas-cor` que, dadas uma figura e uma cor, nesta ordem, retorna uma figura com o mesmo nome da original, mas contendo apenas as formas da cor selecionada da figura original.

8. Crie a função `compara` que, dadas duas figuras, gera uma imagem contendo os nomes das figuras, lado a lado. Abaixo, devem ser exibidos pares de formas que são de cores iguais nas posições iguais nas duas listas. Os pares devem estar um abaixo do outro, indicando as posições nas quais a coincidência acontece. Um exemplo de saída é a imagem abaixo:



9. (Desafio - Ponto extra) Desenvolva a função `quantas-cada-cor` que, dada uma figura, devolve uma lista de elementos do tipo `Par` (que deve ser definido), onde cada par é composto por (i) uma cor que aparece na lista de formas da figura e (ii) o número de formas desta cor na lista de formas da figura. A lista resultante deve conter apenas um par para cada cor que aparece na lista de formas da figura. *Dica: Pense bem em como usar as funções definidas nos exercícios anteriores para executar esta tarefa.*