Programação Funcional
Recursão generativa

Marco A L Barbosa malbarbo.pro.br

Começando

1) O que é recursão generativa?

2) De que forma o processo para projeto de funções recursivas generativas é diferente do processo para funções recursivas estruturais.

Praticando

3) Projete uma função que converta um número natural para string usando a seguinte estratégia: se o número é menor que 10, converta o número diretamente para uma string (pode ser usando seleção com 10 casos!), senão **gere** dois números separando o último dígito do número (usando resto da divisão por 10) dos demais dígitos (usando divisão inteira por 10), converta recursivamente os dois números para string e junte as strings resultantes. Por exemplo, separando 451 obtemos 45 e 1, convertendo para strings (recursivamente) obtemos `"45"` e `"1"`, juntando as strings obtemos `"451"`. Faça a análise do tempo de execução.

4) Projete uma função que inverta os caracteres de uma string usando a seguinte estratégia: se a string é vazia ou se só tem um caractere, então a string invertida é a mesma, senão **gere** uma nova string removendo o primeiro e o último caractere da string, inverta essa string recursivamente e junte com o primeiro (no final) e com o último (no início). Por exemplo, removendo o primeiro e o último da string `"roma"` obtemos `"om"`, que invertida é `"mo"`, juntando com o o primeiro e o últimos obtemos `"amor"`. Faça a análise do tempo de execução.

5) Projete uma função que encontre o máximo de uma lista não vazia de números usando a seguinte estratégia: se a lista tem apenas um elemento, então ele é o máximo, senão, **gere** duas listas dividindo a lista de entrada na "metade", encontre o máximo de cada metade recursivamente e depois determine o máximo entre os dois resultados das chamadas recursivas. Por exemplo, dividindo a lista 4, 1, 3, 2, 7, 3 na metade obtemos as listas 4, 1, 3 e 2, 7, 3, o máximo da primeira lista (determinado recursivamente) é 4, o máximo da segunda lista (determinado recursivamente é 7), o máximo entre 4 e 7 é 7, portanto o máximo da lista é 7. Faça a análise do tempo de execução.

6) Projete uma função que ordene uma lista de números usando a seguinte estratégia: se a lista é vazia, então já está ordenada, senão selecione os elementos mínimos da lista e **gere** uma nova lista sem os mínimos, ordene recursivamente a lista sem os mínimos e junte com a lista com os mínimos. Por exemplo, para a lista 5, 1, 4, 1, 2, 5, 1 os elementos mínimos são 1, 1, 1 e a lista sem os mínimos é 5, 4, 2, 5. Ordenando (recursivamente) a lista sem os mínimos obtemos 2, 4, 5, 5. Juntando com a lista dos mínimos obtemos a lista ordenada 1, 1, 1, 2, 4, 5, 5. Faça a análise do tempo de execução.

Desafios

7) Projete uma função que gere todas as permutações de uma lista de elementos distintos. Por exemplo, para a entrada `[1,2,3]` a saída deve ser `[[1,2,3], [1,3,2], [2,1,3], [2,3,1], [3,1,2], [3,2,1]]`. Faça a análise do tempo de execução.

8) Escreva uma função que conte de quantas formas diferentes podemos dar um valor de troco a partir de uma lista de valores de moedas. Por exemplo, existem 3 formas de dar o troco para o valor 4 se você tiver moedas de 1 e 2: $1+1+1+1$, $1+1+2$, $2+2$.