Estrutura de Dados 1

Lista 2 - Recursão

Todos os códigos DEVEM ser escritos na linguagem de programação JAVA.

Data de Entrega: 17/mar

O objetivo desta lista de exercícios será o de implementar algumas funções recursivas. Relembrando: uma função é dita recursiva quando ela é definida em termos dela mesma, tendo um caso base que serve de condição de parada e uma chamada à própria função, com um caso simplificado do problema, para posterior junção dos vários resultados.

1) Um problema comum em computação é a verificação se uma cadeia de caracteres de uma variável armazena um valor numérico, ou não. Quer dizer, se em todas as posições desta variável só existem caracteres entre 0 e 9. Crie uma função chamada `isNumero()` numa classe `ApenasCaracteres.java` que recebe como argumento um parâmetro do tipo `String` e retorna um `boolean`, sendo `true` quando existirem apenas números e `false` se em alguma posição existir algo que não for um número. **Essa função deve ser implementada de forma recursiva.**

   Caso 1:
   Parâmetro de entrada na função: 123456
   Resultado retornado: true

   Caso 2:
   Parâmetro de entrada na função: 123456A
   Resultado retornado: false

   Caso 3:
   Parâmetro de entrada na função: A983B
   Resultado retornado: false

2) Todo número decimal possui também uma representação binária, sendo o seu valor apresentado apenas com os dígitos "0" e "1". Por exemplo, o número 12 em base decimal possui como representação binária o valor 1100 (pois $1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 0 * 2^0$). Implemente a função `baseBinaria()` da classe `Binaria.java` que recebe como parâmetro um número inteiro decimal positivo e devolve a sua representação binária. **Essa função deve ser recursiva.**

   Caso 1:
   Parâmetro de entrada na função: 12
   Resultado retornado: 1100

   Caso 2:
   Parâmetro de entrada na função: 15

Resultado retornado: 1111

Caso 3:
Parâmetro de entrada na função: 91
Resultado retornado: 1011011

3) A sequência de Collatz, também conhecida como problema 3n + 1, é uma sequência de números naturais gerada a partir de um número **inicial n** e que siga as seguintes duas regras:

Se n for par, deve-se dividir n por 2.
Se n for ímpar, multiplique n por 3 e some 1.
Termina com os últimos números da sequência sendo 4, 2, 1.

Crie uma função recursiva chamada `imprimeSeqCollatz` que receba um número inteiro positivo e imprima todos os números desta sequência.

Caso 1:
Parâmetro de entrada na função: 6
Resultado retornado: 6, 3, 10, 5, 16, 8, 4, 2, 1.

Caso 2:
Parâmetro de entrada na função: 10
Resultado retornado: 10, 5, 16, 8, 4, 2, 1.

Caso 3:
Parâmetro de entrada na função: -100
Resultado retornado: Erro. Considere apenas números estritamente positivos.

Caso 4:
Parâmetro de entrada na função: 100
Resultado retornado: 100, 50, 25, 76, 38, 19, 58, 29, 88, 44, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.