Exercício 2

Recursão

Unidade Curricular de
Laboratórios de Programação
2021/2022

Objetivos

- Aplicação do conceito de recursão a problemas concretos.

Antes de Começar

De modo a poder realizar este projeto deverá já ter lido o guião sobre recursão e recordado o que foi ensinado em AED sobre o assunto.

O que fazer

- Descarregar o arquivo `studentsExercise2.zip` disponível na página de LabP e descompactá-lo. Esse arquivo contém:
    1. O ficheiro `RunRecursive.java`
    2. O ficheiro `TestsRecursive.java`, contendo vários testes `JUnit`
    3. O ficheiro de texto `expectedOutput.txt` contendo o resultado esperado da execução de `RunRecursive`
- No Eclipse, criar um novo projeto Java e, em seguida:
    a) copiar para dentro da pasta `src` desse projeto os dois ficheiros dados com código Java;
    b) Configurar o `Build Path` desse projeto Java para incluir, na `classPath`, a biblioteca `JUnit5` (veja como o fazer na página 14 do tutorial sobre o Eclipse IDE).
- Criar, na pasta `src`, uma classe `Recursive.java` com **implementações recursivas** dos métodos abaixo indicados.

Para testar as suas implementações pode usar a classe `RunRecursive`, comparando depois o ficheiro produzido `output.txt` com o ficheiro dado `expectedOutput.txt`, e também usar a classe `TestsRecursive` de testes `JUnit`.

Segue-se uma descrição detalhada de cada um dos métodos a implementar, que deverão ser todos métodos públicos e métodos de classe (métodos `static`):

1. `double harmonicSum(int k)` que calcula a soma dos primeiros `k` termos da série harmónica. A série harmónica é a série

$$1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\cdots+\frac{1}{n}+\cdots=\sum_{n=1}^{\infty}\frac{1}{n}$$

2. Considere que uma empresa marca as caixas em que envia as encomendas para os seus clientes com uma etiqueta, que é uma sequência de algarismos obtida da seguinte forma:
   a) Cada tipo de produto contido na caixa é representado por uma sequência de 4 algarismos, sendo os 2 primeiros o código desse tipo de produto e os 2 últimos a quantidade contida na caixa desse tipo de produto.
   b) A etiqueta de uma caixa é a concatenação de todos os códigos (cada um deles com 4 algarismos) que representam o conteúdo da caixa.

   Por exemplo, a etiqueta `12046511` seria usada numa caixa que contém 4 unidades do produto com código `12` e 11 unidades do produto com código `65`.

   Implemente o método `int containedQuantity(String boxCode, String productCode)` que, dados o código (ou etiqueta) de uma caixa e o código de um produto, devolve a quantidade desse produto existente na caixa. Deve devolver 0 se o produto não existe na caixa.

3. `boolean allInInterval(int[] values, int x, int y)` que, assumindo que `values != null`, verifica se todos os elementos de `values` pertencem ao intervalo fechado [`x, y`].

4. `int[] sandwich(int[]v, int a, int b)` que recebe um vetor de inteiros `v` e devolve um vetor no qual entre cada dois elementos de `v` foi inserido o valor `a` ou o valor `b`, alternadamente. Por exemplo, se `w` for `{2,5,7,8}`, `sandwich(w,0,-1)` deve devolver `{2,0,5,-1,7,0,8}`. Assuma que `v != null` e `v.length >= 1`.

5. `int[] merge(int[]v, int[]w)` que recebe dois vetores de inteiros, `v` e `w`, que se assume estão ambos ordenados por ordem crescente, e devolve um novo vetor, também ordenado, contendo todos os elementos de `v` e de `w`. Por exemplo, sendo `v` o vetor `{-4,2,10}` e `w` o vetor `{2,4,12,20,22}`, `merge(v,w)` deve devolver `{-4,2,2,4,10,12,20,22}`.

6. `long negativeLucasNumber(int n)` que, para um dado $n \leq 0$, calcula o elemento de ordem `n` da sequência `L` tal que:

   $L_0 = 2$;

   $L_{-1} = -1$;

   $L_n = L_{n+2} - L_{n+1}$, para $n \leq -2$.

   A sequência em causa é a seguinte (por ordem decrescente de `n`):

   `2, -1, 3, -4, 7, -11, 18, …`

Deve apresentar implementações que sejam recursivas (direta ou indiretamente), mas na medida do possível eficientes.

Antes de Entregar

Antes de entregar, certifique-se da correção da formatação e inclua comentários *javadoc* para todos os métodos.

Na classe `Recursive` o cabeçalho javadoc deve incluir a *tag* `@author` com o seu nome e número de aluno.

Entrega

Deve submeter um ficheiro `T2fcxxxxx.zip`, onde `xxxxx` é o seu número de aluno, contendo os ficheiros `Recursive.java` e `RunRecursive.java`.