This third Java programming work consists of the partial implementation of a binary search tree to store integers, which is described in the book Estruturas de Dados e Algoritmos em Java, by António Adrego da Rocha, published by FCA in 2011..

It is intended to implement the ***ABPTree*** data type using defensive programming with exceptions thrown in case of abnormal data situations that violate the preconditions and/or the postconditions of the methods. For this purpose, two exception classes called *EmptyTreeException* and *RepeatedElementException* are provided to indicate, respectively, when a tree is empty and when an element already exists in the tree. The other exception class used *NoSuchElementException* already exists in the Java language.

1. Start by reading the ***ABPTree*** class description (**ABPTree.html** document). Then implement the missing methods: *recSumTotal*; *recCountEnd5*; *recSumEven*; *recCountMult3*; e *recCountLefties*. Under no circumstances should you change the other methods. **Take in considerations that this class gives a compiler error because the signature (prototype) of the last function is incomplete.**
2. You can use the **SimABPInteger.java** simulation program to test the algorithms of insertion, deletion and searching elements in a binary tree.
3. Use the **TestABPInteger.java** unitary testing program to verify that your methods are giving the correct results.

The work must be delivered by June 30, 2022. Only one of the students in the group must send by email to adrego@ua.pt a zip file containing only the final file of the class: **ABPTree.java**.

For any questions about Java you can consult the book Estruturas de Dados e Algoritmos em Java. You can and should also contact me by email with any doubts about the work.