# Relatório Projeto 3.1 AED 2020/2021 Versão 1.0

Nome: António Marques Maria Nº Estudante: 2017265346

TP (inscrição): 4 *Login* no *Mooshak*: 2017265346

Nº de horas de trabalho: 10 *H* *Aulas Práticas de Laboratório:* 2 *H* *Fora de Sala de Aula:* 8 *H*

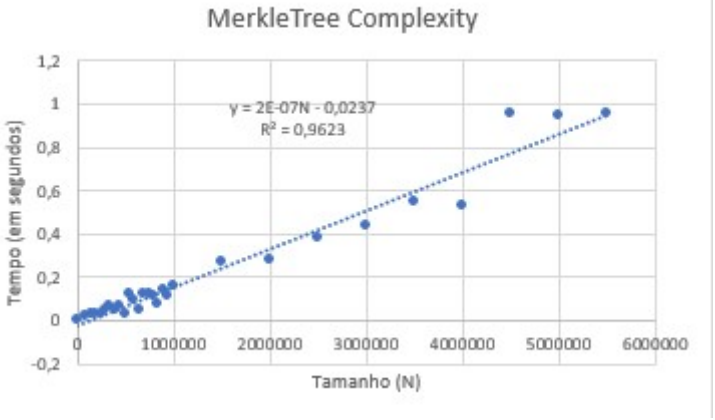
**(A Preencher pelo Docente) CLASSIFICAÇÃO:**

**Comentários:**

## 1. Análise Empírica de Complexidade

Correr a implementação do projeto 3.1 para um número crescente de transações e obter os tempos de execução (excluindo tempo de leitura e impressão de resultados). Produzir tabela, gráfico e regressão relevantes.

| N | Tempo(em segundos) |
|---|---|
| 100000 | 0.015 |
| 200000 | 0.031 |
| 300000 | 0.047 |
| 400000 | 0.049 |
| 500000 | 0.031 |
| 600000 | 0.094 |
| 700000 | 0.125 |
| 800000 | 0.109 |
| 900000 | 0.141 |
| 1000000 | 0.157 |
| 1500000 | 0.269 |
| 2000000 | 0.281 |
| 2500000 | 0.377 |
| 3000000 | 0.437 |
| 3500000 | 0.549 |
| 4000000 | 0.532 |
| 4500000 | 0.953 |
| 5500000 | 0.954 |




A expressão f(N) está de acordo com o esperado? Justifique.

Sim, uma vez que o algoritmo tem de percorrer a árvore inteira para a construir, ou seja, como a árvore tem $2^{log2(N)+1} - 1$ nós, isso significa que a construção da MerkleTree tem complexidade O( $2^{log2(N)+1} - 1$) = O( $2^{log\ (N)} * 2$) = O( $2^{log2(N)}$) = O(N). Neste caso prático, a expressão fIN) = 2E-07N - 0,0237 que corresponde a O(N).

O projeto 3.1 pode ser implementado seguindo uma abordagem iterativa e uma recursiva.

Explique sucintamente o essencial das duas implementações em termos de estruturas de dados utilizadas e da propagação dos *hashcodes* na árvore

A implementação iterativa permite que a HashTree seja representada por um array, o que faz com que a propagação das hashcodes aconteça do final do array até ao início. A implementação recursiva permite que seja representada por uma estrutura em árvore, fazendo com que a propagação das hashcodes aconteça de baixo para cima, ou seja, da base da árvore até à raiz.