Relatório Projeto 2 AED 2023/2024

Nome: Nuno Batista Nº Estudante: 20222 16127
PL (inscrição): 8 Email: nunomarquesbatista@gmail.com

IMPORTANTE:

- Os textos das conclusões devem ser manuscritos... texto que não obedeça a este requisito não é considerado.
- Texto para além das linhas reservadas, ou que não seja legível para um leitor comum, não é considerado.
- O relatório deve ser submetido num único PDF que deve incluir os anexos. A não observância deste formato é penalizada.

1. Planeamento

	Semana 1	Semana 2	Semana 3	Semana 4	Semana 5
Árvore Binária	**X**				
Árvore Binária Pesquisa	**X**				
Árvore AVL		**X**	**X**		
Árvore VP				**X**	
Finalização Relatório					**X**

2. Recolha de Resultados *(tabelas)*

Tempos de inserção da árvore binária

Time (µs) / Num of elements	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
Increasing	1099251	4522595	10182888	18371329	29255661	42140596	56736357	73484421	94451887	115451375
Decreasing	1096080	4539683	10059207	18656949	29255278	41557179	56788185	74177070	95100838	114403985
Random	2015419	8183392	18436142	33464898	52795346	75752976	102863823	133585801	171057060	205233892
Random (90% repeats)	203651	796793	1851138	3326820	5192647	7430250	10016538	12764280	16765115	20519097

Tempos de inserção da árvore binária de pesquisa

Time (µs) / Num of elements	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
Increasing	75584	328322	730220	1329467	2102773	2978719	4052855	5195853	6546428	8150103
Decreasing	78916	324945	724376	1283615	2036173	2985570	3943820	5135501	6731330	8157903
Random	1320	3008	4901	7121	10471	11238	13587	15683	18368	21109
Random (90% repeats)	693	1448	2384	3663	4992	5947	7223	8050	9475	10768

Tempos de inserção da árvore AVL

Time (µs) / Num of elements	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
Increasing	2805	5678	9036	12732	16362	19735	23971	28643	30790	35309
Decreasing	2335	5215	8145	10940	14585	17441	20616	25028	29115	30820
Random	3911	8677	13505	19563	27996	32313	36502	41314	48497	56332
Random (90% repeats)	2065	4834	7504	11052	15416	18745	21300	24186	27711	31486

Tempos de inserção da árvore vermelha e preta

Time (µs) / Num of elements	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
Increasing	920	1892	3021	4275	5186	6509	7732	8764	9738	11235
Decreasing	868	1817	2995	3854	4902	5997	7024	9484	9629	10258
Random	1408	3134	5022	7339	10255	11986	13478	16324	18382	20581
Random (90% repeats)	891	1848	2961	4462	5757	7131	8536	9704	11408	12293

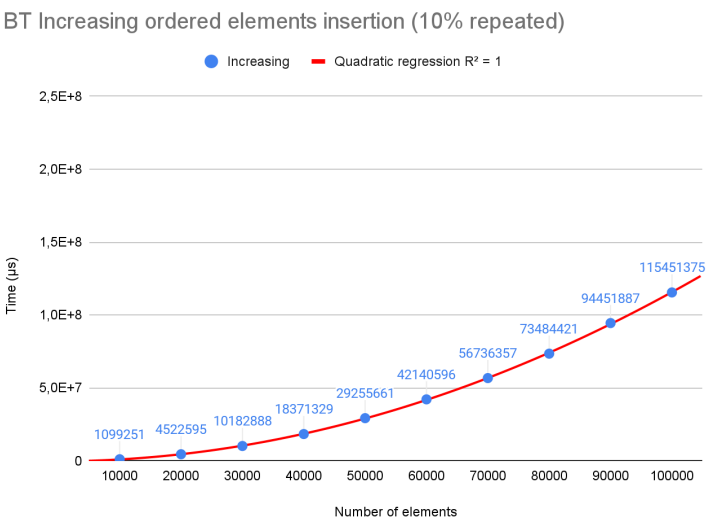
Número de rotações da árvore AVL

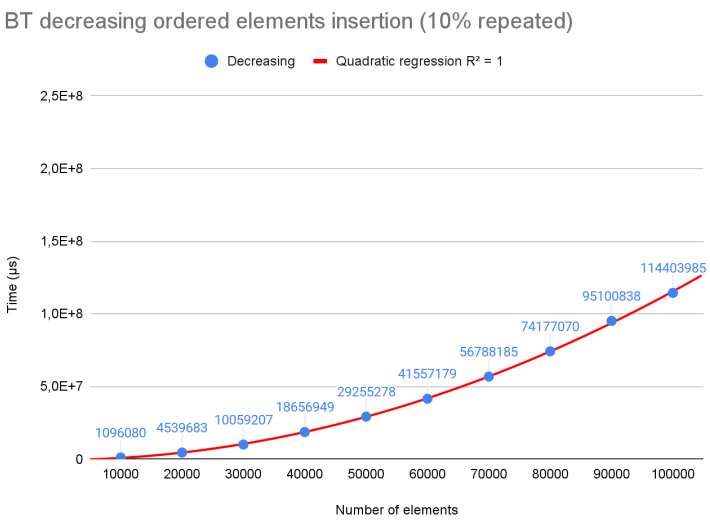
Rotations / Num of elements	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
Increasing	6020	12030	18188	24068	30080	36137	42167	48204	54301	60246
Decreasing	6000	12072	18150	24135	30102	36095	42292	48327	54332	60532
Random	11273	22999	34421	45400	57337	68281	79788	90572	101939	113628
Random (90% repeats)	1235	2275	3467	4702	5704	6955	8080	9432	10720	11854

Número de rotações da árvore vermelha e preta

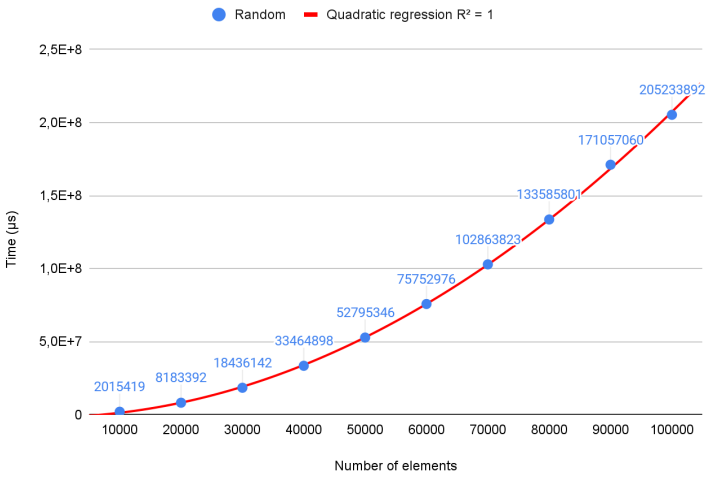
Rotations / Num of elements	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
Increasing	6010	12019	18176	24056	30068	36124	42154	48191	54288	60233
Decreasing	5990	12061	18138	24123	30090	36082	42279	48314	54319	60519
Random	5832	11699	17572	23156	29340	35018	40861	46450	52542	58129
Random (90% repeats)	602	1159	1754	2256	2878	3443	4004	4744	5215	5797

3. Visualização de Resultados *(gráficos)*

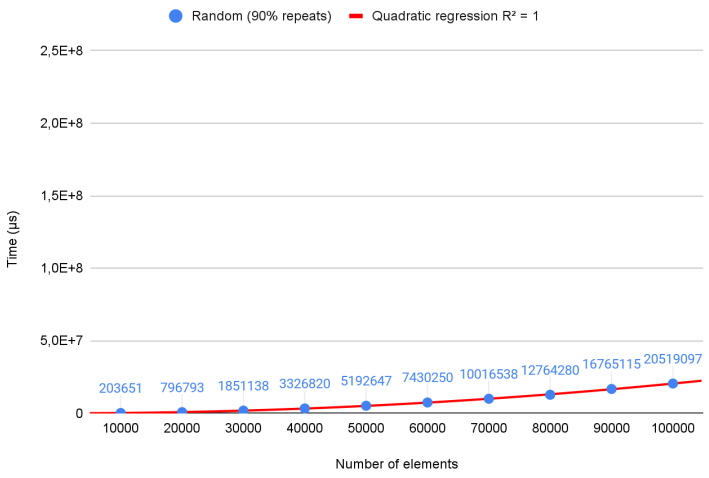




BT random unordered elements insertion (10% repeated)

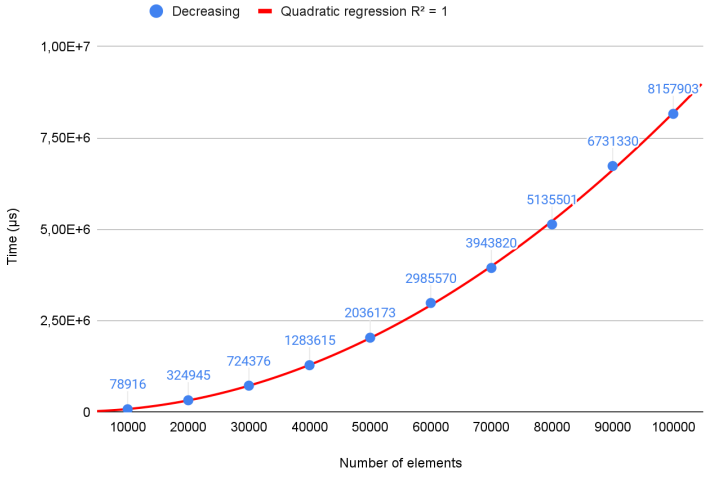


BT random unordered elements insertion (90% repeated)

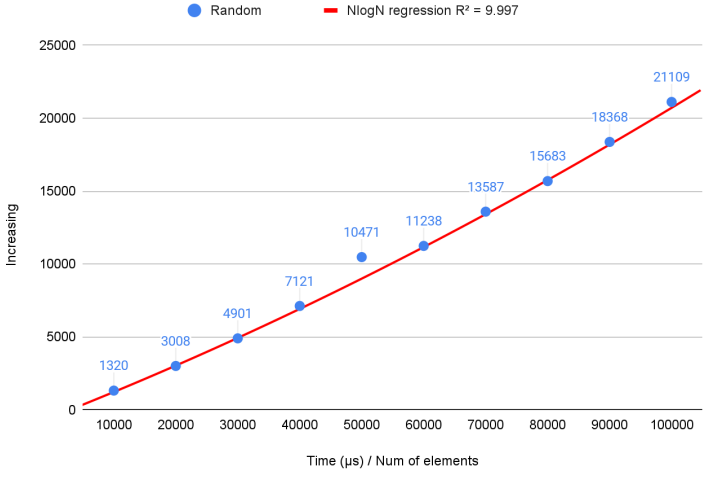


BST increasing ordered elements insertion (10% repeated)

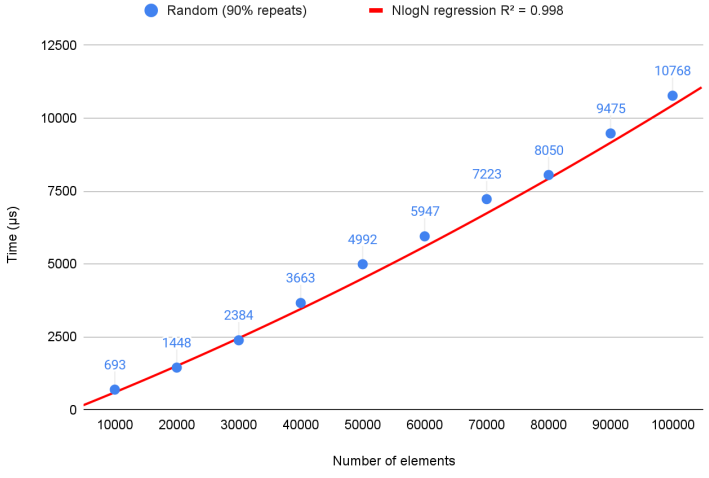
BST decreasing ordered elements insertion (10% repeated)



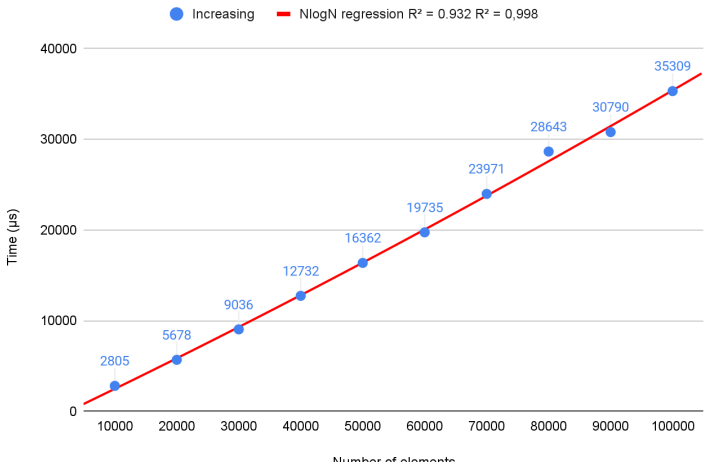
BST random unordered elements insertion (10% repeated)



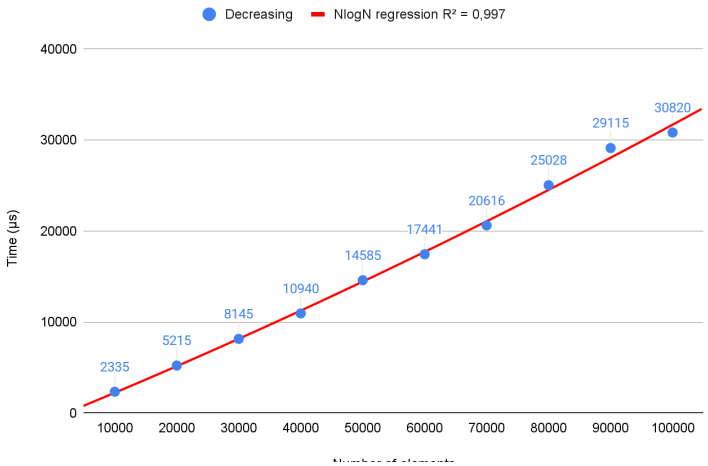
BST random unordered elements insertion (90% repeated)



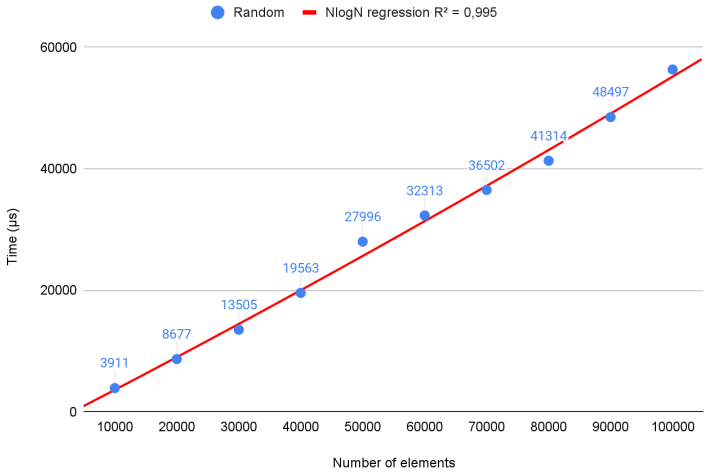
AVL increasing ordered elements insertion (10% repeated)



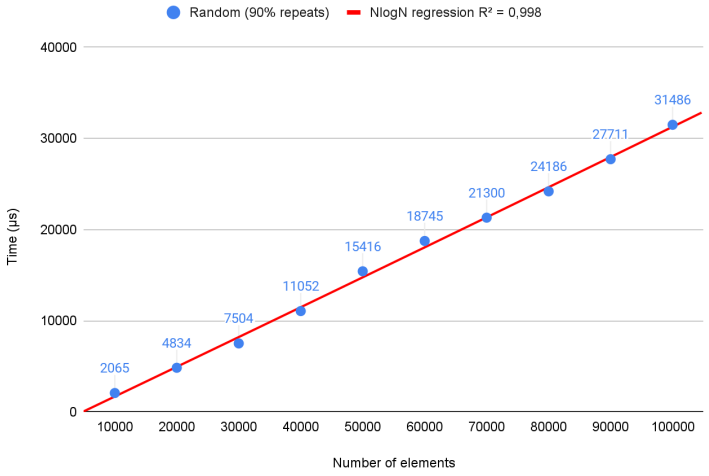
AVL decreasing ordered elements insertion (10% repeated)



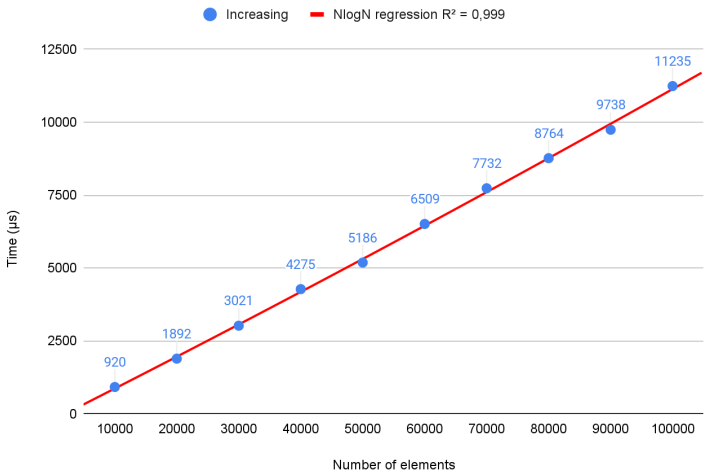
AVL random unordered elements insertion (10% repeated)



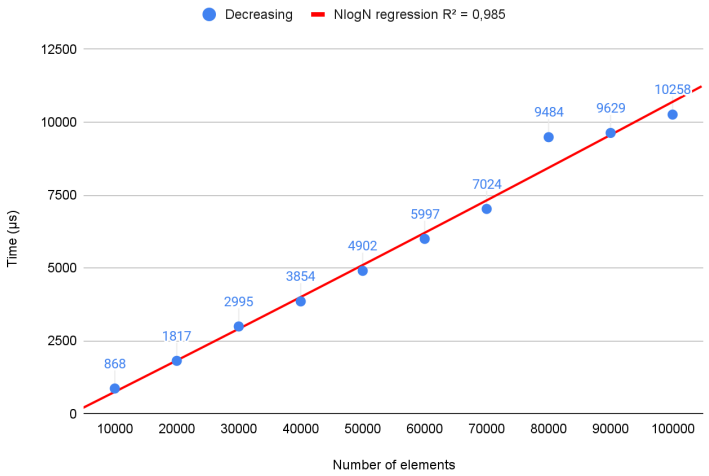
AVL random unordered elements insertion (90% repeated)

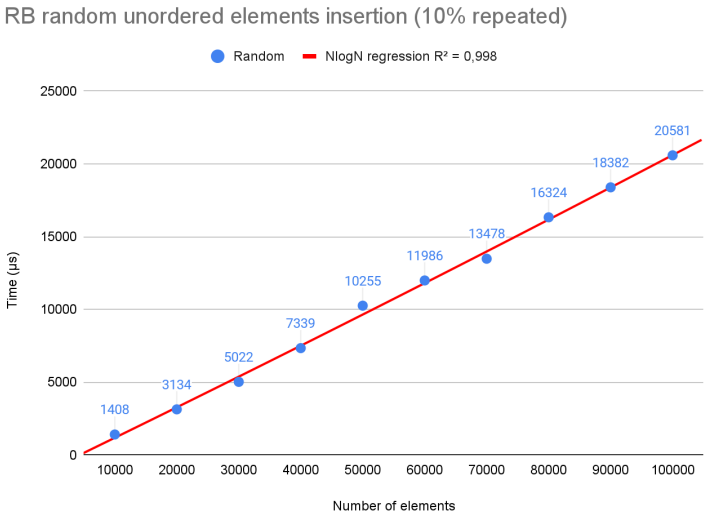


RB increasing ordered elements insertion (10% repeated)



RB decreasing ordered elements insertion (10% repeated)





4. Conclusões *(as linhas desenhadas representam a extensão máxima de texto manuscrito)*

4.1 Tarefa 1 BT

A complexidade de inserção de um elemento na árvore binária, assumindo que já estão N elementos é O(N), uma vez que, antes da inserção propriamente dita, é necessário percorrer a árvore completa para verificar se o elemento já existe. Se não existir, é necessário percorrer a árvore novamente para inserir o nó cujo pior caso também é O(N), quando a árvore é equivalente a uma lista ligada. Assim, a complexidade de inserção de N elementos é $O(N^2)$. Os melhores resultados foram obtidos para o conjunto de elementos aleatórios com 90% de repetições pois a verificação inicial é mais rápida.

4.2 Tarefa 2 BST

A complexidade de inserção de um elemento na árvore binária de pesquisa, no pior caso é O(N), este caso acontece nos conjuntos ordenados pois a árvore escalva por ser equivalente a uma lista ligada. assim, a complexidade total de inserção é $O(N^2)$. No entanto, no caso médio, a complexidade é O(logN) (O(NlogN) para N elementos), isto acontece quando a árvore está balanceada e é necessário descer apenas logN níveis para encontrar o lugar correto de inserção, este resultado é espectável no conjunto de aleatórios. Os melhores resultados foram obtidos no conjunto de aleatórios com 90% de repetidos, uma vez que, para além da árvore ter mais tendencia para ser balanceada, também vai ter menos elementos no

total, o que se traduz para menos níveis a descer.

4.3 Tarefa 3 AVL

A complexidade de inserção na AVL é O(logN) (O(NlogN para N elementos), uma vez que a AVL está sempre balanceada, em cada inserção, é necessário descer logN níveis para encontrar o lugar de inserção, bem como subir o mesmo número de níveis de modo a verificar o balance factor, então, a complexidade é O(NlogN)+O(NlogN)=O(NlogN). Apesar da complexidade ser igual à da VP, os resultados foram piores para a AVL pois esta é mais estritamente balanceada (balance factor não pode ser -2 ou 2). Os piores resultados foram obtidos para o conjunto aleatório, uma vez que este conjunto gera muitos mais casos com rotações duplas. Apesar de ser mais lenta que a VP a inserir, este balanceamento rigido leva a melhores tempos de pesquisa, o que, em teoria, faz com que a AVL devesse ter um tempo total do conjunto aleatório de 90% repetidos melhor que a VP, uma vez que a AVL deteta mais rapidamente os elementos repetidos.

4.4 Tarefa 4 VP

A complexidade de inserção na VP é O(logN)(O(NlogN) para N elementos) pois a VP é balanceada, mesmo que não de maneira tão rígida, é necessário descer logN níveis. Os resultados obtidos foram melhores que os da AVL pois a VP é menos estritamente balanceada, o que significa que não é necessário fazer tantas rotações. Claro que apesar de ser mais rápida a inserir, este balanceamento menos rígido vai levar a piores tempos de pesquisa. É de notar que apesar de, em média, a VP requerer menos rotações que a AVL, no caso dos conjuntos ordenados, o número de rotações necessárias é semelhante, uma vez que tanto as regras da VP como as de AVL são violadas a quase todas as rotações.

Anexo B - Delimitação de Código de Autor

Classe Node:

- Tudo

Árvore binária:

- Todas as funções

Árvore binária de pesquisa:

- Todas as funções

Árvore AVL:

- Função de inserção
- Função de balance factor e height

Árvore VP:

- Função de inserção

Anexo C - Referências

Árvore AVL:

- Rotações com base nos exemplos do geeksforgeeks.org + slides das aulas teóricas

Árvore VP:

- As rotações são modificações das rotações da AVL

Representação intuitiva das árvores:

- Ideia tirada do primeiro comentário do post: https://stackoverflow.com/questions/36802354/print-binary-tree-in-a-pretty-way-using-c

Função de fix insertion da árvore VP:

- Com base no exemplo do programiz.com + slides das aulas teóricas

Geração de inputs:

- Copiado de: https://stackoverflow.com/questions/57025341/how-to-understand-c11-random-number-generator