Projeto 3 – Estruturas de Dados I
SCC-0202 – Relatório

Marcus Vinicius C. B. Martins – 11219237
Pedro Guerra Lourenço – 11218747

1. No tipo de estrutura 2, qual seria a influência no tempo de inserção e de busca caso fosse possível utilizar um ponteiro auxiliar que apontasse para a última posição da lista?

R.:
Caso houvesse um ponteiro auxiliar apontando para a última posião da lista, a complexidade do algoritmo de inserçã seria O(1) em vez de O(n) - que é complexidade atual, percorrendo todos os itens da lista até chegar no último para nserir -, pois sempre teríamos acesso direto ao local onde iríamos inserir o novo item. Já para o algoritmo de busca, ão haveria diferença em sua complexidade, pois, como a lista é desordenada, não há diferença em começar pelo iníci ou pelo final da lista. A complexidade para a busca continuaria sendo O(n).

==========

2. Dos tipos de estruturas baseadas em lista, qual delas tem o maior e menor tempo de inserção? Justifique sua resposta

R.:
A estrutura que possui maior tempo de inserção é a lista ordenada (1). Era de se esperar que a estrutura (1) ou a (2) f ssem a de maior tempo, dado que a complexidade da inserção é O(n) em ambas, sendo maior que todas as outras estr turas. No caso do conjunto de dados apresentado, a ordem específica de inserção fez com que houvessem mais comp rações na inserção ordenada do que na sequencial (final). Isso pode acontecer em casos específicos, por exemplo no aso em que os dados já estão ordenados (T(n) = 2n na implementação do grupo para a ordenada e T(n) = n para a des rdenada).
Já a estrutura que possui o menor tempo de inserção é a lista desornada com inserção na primeira posição, apresenta do complexidade de inserção O(1), pois há um ponteiro apontando para o ínicio da lista, onde será inserido o novo v lor.

==========

3. Como os valores de entrada fornecidos para este projeto influenciam no tempo de busca em estruturas baseadas em listas? A ordem dos valores inseridos e buscados importa? O tamanho dos valores importa? A quantidade de valores importa?

R.:
Primeiramente, vale ressaltar que quanto menor a quantidade de valores inseridos, mais rápido será o programa - tan o para inserir quanto para buscar valores - independentemente da estrutura baseada em lista utilizada. Além disso, o amanho das entrada deve respeitar o limite inferior e superior suportado pelo tipo de dado utilizado na lista (nesse pr jeto: o tamanho de um inteiro).
Tendo isso em vista, para as listas com inserção desordenada (no início e no fim), os valores de entrada não infuenci m em praticamente nada, pois a inserção é sempre efetuada em um local pré-definido (começo ou fim da lista) e a bu ca sempre passa por todos os itens da lista, no pior caso. Logo, a complexidade dos algoritmos de busca e de inserçã permanece inalterada independentemente dos valores de entrada. A única mudança que poderia ocorrer é: a presenç de muitos itens de mesmo valor faria com que a complexidade da busca diminuísse significativamente, pois o valor

uscado logo seria encontrado devido à repetição de valores na lista.
Para a lista ordenada, os valores de entrada influenciam, principalmente, no algoritmo de inserção. A inserção é feita de forma ordenada, ou seja, o novo valor é inserido antes de primeiro valor maior que ele (após o maior valor menor que o novo item). Sendo assim, conforme a sequência dos valores de entrada aproxima-se de uma sequência decresc nte, o algoritmo de busca aproxima-se da complexidade O(1), pois os novos valores são inseridos próximos do ínici da lista (quando um valor na lista é maior que o novo, este já é inserido - só percorremos uma pequena parte da lista para então inserir). Já no caso dos dados serem já próximos a uma lista ordenada crescente, a complexidade se aprox ma de O(n), pois no pior caso, que os dados já estão ordenados crescentemente, cada inserção percorerrá a lista intei a para inserir no fim.

==========

4. Existe alguma diferença significativa no tempo de busca entre os tipos de estruturas
baseadas em árvore? Justifique sua resposta.

R.:
Não, apesar de haver uma pequena diferença no tempo de busca, a distribuição dos dados inseridos não caiu nos cas s degenerados em que a árvore se aproxima da forma de uma lista. Manteve-se a ABB próxima o suficiente de uma rvore balanceada (o que a AVL faz forçadamente, justificando o maior tempo de inserção nela e a garantia de uma b sca eficiente).

==========

5. De acordo com valores de entrada fornecidos para este projeto, a estrutura tipo 5
melhora o tempo de busca em relação ao tipo 4 ? Justifique sua resposta.

R.:
Sim, mas a melhora é bem pequena, pois essa sequência específica dos valores de entrada faz com que a árvore biná ia de busca tenha uma boa distribuição (equilibrada) de valores para suas subárvores direita e esquerda em cada nó, u seja, a ABB aproxima-se de uma AVL (árvore balanceada). Assim, o algoritmo de busca da ABB irá aproximar-se de O(log n), mas isso ainda não é garantido, pois a diferença entre as alturas das subárvores direita e esquerda poder ser mais do que 1 em algum nó. A estrutura 5 (AVL) garante exatamente isso (subárvores direta e esquerda com dif rença de altura = 1). Consequentemente, seu algoritmo de busca será O(log n) em qualquer caso.
Caso a sequência de entradas fosse outra, a complexidade da busca de ABB poderia aproximar-se de O(n), enquanto a a busca na AVL continuaria O(log n).

==========

6. De todas estruturas implementadas, qual é a mais recomendada em ambos os casos
de inserção e busca para este projeto? Justifique sua resposta.

R.:
Ponderando entre inserção e busca, descarta-se as listas, pois estas possuem tempo demasiadamente grande para um dessas operações. Entre ABB e AVL, levando em conta a natureza dos dados, que não desbalanceia muito a ABB, p de-se dizer que a ABB possui vantagens, com aproximadamente 75% do tempo de inserção da AVL, possuindo ape as 6% a mais que a AVL de tempo na busca. Portanto, a ABB é recomendada quase que certamente, exceto no caso ue a busca é feita com muito mais frequência que a inserção, de modo que os 6% superem os 75% devido à quantida e de operações.
Evidencia-se que, caso outra sequência de entrada fosse fornecida ou case a busca procurasse por outros valores, talv

z a AVL fosse uma melhor escolha.

==========

7. Na sua opinião, quais foram os desafios e aprendizados durante a implementação do projeto?

R.:
Dificuldades:

- Implementação da rotação: caso especial em que a raiz é afetada
- Vazamento de memória no programa fornecido

Aprendizados:

- Função clock e cálculo de "delta time"
- Leitura de arquivos com fopen e fscanf
- Trabalhar com enums para distinguir os tipos de coleção
- Uso de funções macro, como a função max definida no colecao.c
- Prática da implementação das árvores ABB e AVL.
- Trabalho em equipe e versionamento de código, utilizando o GitHub.