Avaliação 02

Descritivo de Implementação

O presente documento tem por objetivo servir de guia para a elaboração dos trabalhos da **segunda** unidade da componente curricular Estruturas de Dados Básicas II. Ao fim desta unidade, é esperado que o aluno tenha conhecimento sobre a teoria relativa ao conteúdo abordado, e consiga replicá-la em um ambiente computacional.

Como ficou definido na aula inicial, as atividades avaliativas irão constar de implementações computacionais dos pressupostos teóricos vistos em sala de aula. Tais implementações deverão seguir algumas regras básicas.

(a) As implementações deverão ser feitas em grupo de até 5 participantes (discentes matriculados na turma);

(b) A linguagem a ser adotada é de livre escolha. Contudo, as implementações devem abranger a teoria vista integralmente. O uso de bibliotecas auxiliares é permitido, porém, não para as atividades fim dos algoritmos. Em outras palavras, as estruturas de dados e pseudocódigos devem ser implementados em sua integralidade;

(c) Apesar de ser uma atividade em grupo, a avaliação ocorrerá de forma individual, de forma que todos devem ter conhecimento dos códigos e funções implementadas pelo grupo.

(d) Não custa lembrar que a presença física nas aulas também é objeto de avaliação, contribuindo para a construção da nota individual. Então, evitem faltar.

(e) A data para entrega (virtual) é o dia **07 de janeiro de 2025.**

Deverá ser entregue:

i. **O código utilizado, para inspeção:** Os códigos deverão estar comentados, em português, detalhando o funcionamento de cada função/procedimento, incluindo o formato da entrada e qual a saída esperada (pode ser GitHub).

ii. **Um vídeo curto**, explicando o funcionamento/estrutura geral do programa, IDE utilizada e os resultados encontrados ao executar os algoritmos.

iii. **Uma lista de atividades desenvolvidas por integrante**, detalhando a participação efetiva em cada implementação, seja em sua concepção, implementação, revisão ou teste.

Para a avaliação, deverá ser considerado o seguinte:

1. Implementação Computacional (Heap)

1.0. Descreva o ambiente computacional utilizado (*Software* e *Hardware*).

1.1. Implemente as funções de *Heap* (máximo e mínimo). Seu código deve constar das funções de (**a) Alteração de Prioridade; (b) Inserção; (c) Remoção (da raiz) e (d) Construção das *Heaps***. Para este item, apresente as saídas do programa com os exemplos vistos em sala de aula.

1.2. Descreva em forma de pseudocódigo e implemente o algoritmo de ordenação *Heapsort*.

1.3. Crie listas aleatórias, contendo inteiros variando entre *0* e *10.000 (dez mil)*, e com o tamanho (da lista) *10.000, 100.000* e *1.000.000* (salve as listas em arquivo txt)

1.4. Compare os resultados obtidos com o algoritmo de ordenação *Heapsort* com os algoritmos implementados no trabalho da primeira unidade (*BubbleSort, MergeSort* e *QuickSort*, iterativo e recursivo) nas listas, salve o resultado da lista ordenada em um arquivo .txt, e compute o tempo de processamento para cada caso. Faça o tabelamento dos resultados, e realize uma análise detalhada. Esta análise deve ser escrita e entregue em um arquivo PDF.

2. Implementação Computacional (Árvores Binárias)

2.0. Descreva o ambiente computacional utilizado (*Software* e *Hardware*).

2.1. Defina (descreva em formato de fluxograma e implemente) uma estratégia para a **criação** de árvore binária a partir de uma lista de dados de entrada. (de preferência, uma estratégia que não deixe a árvore inicial muito desequilibrada) .

2.2. Implemente as funções de **busca**, **inserção** e **remoção** em uma árvore binária. Teste nos exemplos vistos em sala de aula.

2.3. Implemente as funções de percurso em árvore binária: **(a) Pre-Ordem; (b) Ordem-Simétrica; (c) Pós-Ordem** e **(d) Em-Nível**. Teste nos exemplos vistos em sala de aula.

3. Implementação Computacional (Árvores AVL)

3.0. Descreva o ambiente computacional utilizado (*Software* e *Hardware*).

3.1. Implemente as funções de **(a) rotação; (b) busca** *(mesmo que na árvore binária)***; (c) inserção e (d) remoção** em uma árvore AVL. Teste nos exemplos vistos em sala de aula.

3.2. Crie uma árvore AVL, usando a função de inserção, com os seguintes valores inteiros: L={15, 18,20, 35, 32, 38, 30, 40, 32, 45, 48, 52, 60, 50}. Imprima a árvore em nível (ou usando alguma biblioteca de representação gráfica de árvores).

4. Implementação Computacional (Árvores Rubro-Negras)

4.0. Descreva o ambiente computacional utilizado (*Software* e *Hardware*).

4.1. Faça um algoritmo em forma de fluxograma para a função de exclusão de um nó em uma árvore Rubro Negra. (deve ser entregue em formato PDF)

4.2. Implemente as funções de **(a) rotação** *(mesmo que na árvore AVL)***; (b) busca** *(mesmo que na árvore binária)***; (c) inserção e (d) remoção** em uma árvore RN. Teste nos exemplos vistos em sala de aula.

4.3. Crie uma árvore RN, usando a função de inserção, com os seguintes valores inteiros: L={15, 18,20, 35, 32, 38, 30, 40, 32, 45, 48, 52, 60, 50}. Imprima a árvore em nível (ou usando alguma biblioteca de representação gráfica de árvores).