Avaliação 01

Descritivo de Implementação

O presente documento tem por objetivo servir de guia para a elaboração dos trabalhos da primeira 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 conteudo abordardo, 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 algorimos. Em outras palavras, as estruturas de dados e pseudocódigos devem ser implementados em sua integralidade;

(c) Nesta primeira unidade, os algoritmos deverão "rodar" em computador local ou servidor web de livre escolha. Nas próximas unidades, poderemos ter um servidor próprio para executar as atividades. Então, evitem implementações muito específicas de plataformas.

(d) 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.

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

(f) A data para entrega (virtual) é o dia 31 de outubro de 2024.

Deverá ser entregue:

i. **Um arquivo PDF:** As análises teóricas (questão 1, listada ao fim desse arquivo) deverão ser entregues em um arquivo PDF, com o detalhamento do que foi pedido.

ii. **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.

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

iv. **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. Análise Teórica

1.0. Deve ser feita a descrição das funções/procedimentos utilizados e uma explicação do funcionamento geral para o algoritmo (pseudocódigo), buscando justificar o resultado esperado em todos os casos possíveis (explicar porque o seu algoritmo funciona).

1.1. Estabeleça pseudocódigos para o algoritmo de ordenação ***BubbleSort*** em versões iterativa e recursiva e analise a complexidade em relação as notações *Big O, ômega e Theta*. Para a versão recursiva, utilize os 4 métodos vistos.

1.2. Estabeleça pseudocódigos para o algoritmo de ordenação ***MergeSort*** em versões iterativa e recursiva e analise a complexidade em relação as notações *Big O, ômega e Theta*. Para a versão recursiva, utilize os 4 métodos vistos.

1.3. Estabeleça pseudocódigos para o algoritmo de ordenação ***QuickSort*** em versões iterativa e recursiva e analise a complexidade em relação as notações *Big O, ômega e Theta*. Para a versão recursiva, utilize os 4 métodos vistos.

2. Fase de análise de algoritmo

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

2.1. Função iterativa

2.1.A. Implemente os códigos *idadeRep* e *idadeRep2*, vistos em sala de aula;

2.1.B. Crie instâncias aleatórias com idades (inteiros) variando entre 0 e 100, com tamanhos *n=100, n=1.000, n=10.000*

2.1.C. Compute os tempos de processamento e compare com os resultados da análise assintótica vista em sala.

2.2. Função recursiva

2.2.A. Implemente os códigos *buscaBinaria* e *bBinRec*, vistos em sala de aula;

2.2.B. Crie instâncias aleatórias com idades (inteiros) variando entre 0 e 100, com tamanhos *n=100, n=1.000, n=10.000*

2.2.C. Compute os tempos de processamento e compare com os resultados da análise assintótica vista em sala.

2.3. Implemente os algoritmos de ordenação baseados nos pseudocódigos que você sugeriu nos itens **1.A**, **1.B** e **1.C**, em suas versões iterativa e recursiva.

2.4. Crie listas aleatórias, com inteiros variando entre *0* e *1000*, de tamanho *1.000, 10.000* e *100.000* (salve as listas em arquivo txt)

2.5. Execute os algoritmos de ordenação (*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 constar no arquivo PDF enviado.