GOVERNO DO ESTADO DO RIO GRANDE DO NORTE
UNIVERSIDADE ESTADUAL DO RIO GRANDE DO NORTE
DISCIPLINA: LABORATÓRIO DE ESTRUTURA DE DADOS
PROF.: FRANCISCO CHAGAS DE LIMA JÚNIOR

ALUNO 1:_____

ALUNO 2:_____

DATA DA ENTREGA: Até 15/02/2024
OBSERVAÇÕES:

1. A entrega desta atividade, corretamente implementada e no prazo previsto, vale 40% da 2ª nota.
2. A defesa individual desta lista, em forma de seminário, vale 60% da 2ª nota
3. Todos os códigos escritos nesta atividade serão corrigidos considerando a sintaxe da linguagem de programação C ou C++.

ATIVIDADE AVALIATICA II

A. ÁRVORE BINÁRIA DE BUSCA:

1. Implemente uma árvore binária de acordo com a seguinte sequência de números:
   a) 20, 5, 12, 36, 27, 45, 9, 2, 6, 17, 40.
   b) A partir da árvore obtida no item "a)", remova os nós: 9, a seguir o 5, e finalmente o 20.

   Observação: Esta implementação deve conter as seguintes funções:
   F1.: Inserção
   F2.: Listagem
   F3.: Remoção
2. Escreva um algoritmo recursivo que encontre o maior valor armazenado em uma árvore binária de busca.

3. Dada uma árvore binária de busca, onde cada nó é constituído pelas seguintes informações: NOME, SEXO ('M' ou 'F'), IDADE e PESO. Sabendo que a árvore foi construída com a chave NOME e que já existe um ponteiro chamado RAIZ que aponta para o nó raiz da árvore, construir um algoritmo que, a partir desta árvore, gere duas listas ordenadas por NOME, uma para homens e outra para mulheres.

4. Adapte os algoritmos de inserção e remoção em árvores binárias de busca de forma a tratar a ocorrência de conteúdos-chave repetidos, mantendo um contador de ocorrências em cada nó.
5. Escreva uma função que verifique se uma árvore binária de busca é cheia. Uma árvore é dita cheia se todos os nós que não são folhas têm os dois filhos, isto é, não pode existir nó com apenas um filho A função deve retornar 1 no caso de a árvore ser cheia ou 0 no caso de não ser. No caso da árvore ser vazia, a função também deve retornar 1.

B. ÁRVORE AVL:

1. Transforme a árvore implementada no item “3”, em uma árvore AVL.

Observações:

1. Esta implementação deve conter as seguintes funções:
   - F1.: Inserção
   - F2.: Listagem
   - F3.: Remoção
   - F4.: Consulta.
2. Implemente um menu de opções como interface para o programa.

C. GRAFOS:

1. Utilize o tipo abstrato de dados descrito a seguir, para a representação de grafos e implemente funções de gerenciamento de memória (alocação/liberação), inserção e remoção de arcos e listagem do grafo:
   Observação: Leia o grafo de um arquivo “*.txt”.

```
typedef struct grafo {
   int ponderado;
   int V;
   int A;
   int **adj;
   int *grau;
   float **pesos;
}Grafo;
```

2. Pesquise sobre o algoritmo de **Floyd-Warshall** e o implemente, para determinar o caminho mínimo em grafos, utilizando a estrutura de dados “**`typedef struct grafo`**”, já implementada no item anterior.

BOM TRABALHO