Aluno(a):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Matrícula:\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| 1a) | 2,5 |  |
| 2a) | 2,5 |  |
| 3a) | 2,5 |  |
| 4a) | 2,5 |  |
|  | 10.0 |  |

**LEIA COM CUIDADO**

1. A prova é individual e sem consulta.
   1. **Qualquer tentativa de “cola” resultará na anulação da prova do aluno ou dos alunos envolvidos**.
   2. Os aparelhos celulares deverão permanecer desligados e guardados fora do alcance durante toda a prova. **Aparelhos celulares ligados ou de alguma forma visíveis serão tratados como tentativa de “cola”**.
2. A interpretação faz parte da questão.
   1. **Não há perguntas durante a prova.**
   2. Em caso de dúvida escreva a dúvida e a sua interpretação na resposta.
3. O tempo de prova é 1:45 h.
4. **Após o início da prova, não será possível sair e voltar à sala.**
5. As respostas devem seguir as questões. Caso precise de rascunho use o verso da folha.
6. A prova pode ser feita a lápis.

**Questão 1 (2,5 pontos)**

a) **(1,5 pontos)** Defina em pseudo-código um algoritmo não recursivo que recebe como entrada um ponteiro *r* para a raiz de uma árvore binária de busca e uma chave *K* e retorna um ponteiro *n* para o nó *N* da árvore tal que a chave armazenada em *N* seja a *menor chave armazenada na árvore que é maior ou igual* a *K*. Se *N* não existir, o algoritmo deve retornar NULL. Por simplicidade, assuma que os valores de chave são inteiros.

Assuma a seguinte estrutura para os nós de uma árvore binária de busca:

typedef struct \_abb Abb;

struct \_abb {

int chave;

Abb\* pai;

Abb\* esq;

Abb\* dir;

};

b) **(1,0 ponto)** Argumente porque a sua implementação está correta.

***Resposta***

**Questão 2 (2,5 pontos)**

a) **(1,5 pontos)** Implemente em C *de forma não recursiva e eficiente* uma função que calcula a altura de uma árvore AVL, com o seguinte protótipo:

int avl\_height(AvlNode\* node)**;**

visitando o menor número possível de nós. Utilize a estrutura:

typedef struct \_avl\_node AvlNode;

struct \_avl\_node {

void\* info;

int bf; /\* balance factor \*/

AvlNode\* parent;

AvlNode\* left;

AvlNode\* right;

};

b) **(1,0 ponto)** Argumente porque a sua implementação está correta e é eficiente.

***Resposta***

**Questão 3 (2,5 pontos)** Considere as árvores abaixo.

(i)

50

30 39 70 90

10 20 37 38 40 60 80 100

(ii)

50

30 33 70 90

10 20 31 32 41 60 81 89 94 100

Para os itens (a) a (c), descreva passo a passo todas as modificações sofridas, redesenhando apenas a parte da árvore modificada a cada passo. Justifique sempre porque não há mais modificações a fazer.

a)(0,5 ponto) Assuma que a árvore (i) é uma árvore B de ordem 3. Insira a chave 36.

b)(1,0 ponto) Assuma que a árvore (i) é uma árvore B de ordem 3. Remova a chave 70 e depois a chave 90.

c) (0,5 ponto) Assuma que a árvore (ii) é uma árvore B de ordem 3. Remova a chave 50.

d) (0,5 ponto) As árvores (i) e (ii) são representadas acima apenas de forma esquemática. Elas podem ser consideradas uma árvore B de ordem *n*, onde *n* > 3? Explique a sua resposta.

***Resposta***

**Questão 4 (2,5 pontos)**

a) **(0,5 ponto)** Considere a tabela de dispersão estendida, *T*, parcialmente representada abaixo. Quantos apontadores em *T* apontam para o bloco *B1* ? Caso haja mais de um apontador, além do ilustrado na figura, em quais posições da tabela *T* eles estarão armazenados? Explique sua resposta.

*T*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *i =3* |  |  |  |
| 0 |  |  |  | *B1* |
| 1 |  |  |  | *j1 =2* | |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
| 6 |  |  |  |  |
| 7 |  |  |  |  |

b) **(0,5 ponto)** Considere a tabela de dispersão estendida representada na figura abaixo. Suponha que cada bloco possa conter até 3 chaves. Mostre como a estrutura ficará após a remoção de *K21*, assumindo que a estrutura deve economizar espaço sempre que possível. Explique sua resposta.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *i =2* |  |  |  |
| 0 |  |  | *K11* | *B1* |
| 1 |  |  | *K12* | *j1 =2* | |
| 2 |  |  |  |  | |
| 3 |  |  |  |  |
|  |  |  | *K21* | *B2* |
|  |  |  | *K22* | *j2 =2* | |
|  |  |  |  |  | |
|  |  |  |  |  |
|  |  |  | *K31* | *B3* |
|  |  |  | *K32* | *j3 =2* | | |
|  |  |  |  |  | | |
|  |  |  |  |  |
|  |  |  | *K41* | *B4* |
|  |  |  | *K42* | *j4 =2* | |
|  |  |  |  |  | |

c) **(0,5 ponto)** Suponha que um bloco *B* esteja completo. Explique todos os casos de uma inserção de uma nova chave *K* em *B*, supondo que possam haver chaves duplicadas.

d) **(1,0 ponto)** Suponha que cada bloco possa conter um mínimo de 2 chaves e um máximo de 3 chaves. Assuma que a tabela possua 8 entradas. Qual o número mínimo de chaves que uma tabela de dispersão estendida com esta configuração pode conter? Explique sua resposta.

***Resposta***