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)** Defina o *comprimento* de um ramo de umaárvore binária (o que incluiu as árvores de busca binárias e as árvores AVL) como o número de nós que fazem parte do ramo.

a) **(0,5 ponto)** Qual o custo, no pior caso, para computar o comprimento do maior ramo de umaárvore binária com *n* nós. Explique a sua resposta.

b) **(1,0 ponto)** Implemente em C uma função que calcula o comprimento do maior ramo de uma árvore AVL com *n* nós de tal forma que a complexidade, no pior caso, da sua função seja *O*(*log*(*n*)). A função deverá ter o seguinte protótipo:

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

e utilizar a estrutura:

typedef struct \_avl\_node AvlNode;

struct \_avl\_node {

void\* info;

int bf; /\* balance factor \*/

AvlNode\* parent;

AvlNode\* left;

AvlNode\* right;

};

c) **(0,5 ponto)** Argumente porque a implementação da função, apresentada no item (b), está correta.

d) **(0,5 ponto)** Argumente porque a implementação da função, apresentada no item (b), de fato possui complexidade, no pior caso, *O*(*log*(*n*)).

**Resposta**

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

50

30 39 70 90

10 20 37 38 40 60 80 100

**Figura 1. Árvore B de ordem 3.**

100

53 77 123 200

10 40 60 70 80 90 110 113 130 138 140 170 230 236 242 245

**Figura 2. Árvore B de ordem 5.**

Realize as seguintes operações, indicando os nós que sofrem modificações (divisão, redistribuição ou concatenação) após cada operação. Justifique sempre porque não há mais modificações a fazer.

a) **(0,5 ponto)** Insira a chave 36 na árvore-B da Figura 1.

b) **(1,0 ponto)** Remova a chave 70 e depois a chave 90 da árvore-B da Figura 1.

c) **(0,5 ponto)** Insira a chave 143 na árvore-B da Figura 2.

d) **(0,5 ponto)** Remova a chave 100 da árvore-B da Figura 2.

**Resposta**

**Questão 3 (2,5 pontos)** Considere uma família de árvores definida de forma semelhante a árvores B, exceto que:

* Os nós possuem tamanho variável entre 128 e 256 bytes
* As chaves possuem tamanho variável, entre 22 bytes e 44 bytes
* Os ponteiros ocupam 4 bytes
* Em cada nó, há um campo a mais, de 4 bytes, indicando o número de chaves que o nó efetivamente armazena

1. **(1,0 ponto)** Qual é o maior número de chaves que uma árvore de altura 2 armazena (uma árvore que só tem a raiz possui altura 0, por convenção)? Explique cuidadosamente sua resposta.
2. **(1,0 ponto)** Qual é o número mínimo de chaves, estando todos os nós completamente preenchidos, que uma árvore de altura 2 armazena? Explique cuidadosamente sua resposta.
3. **(0,5 ponto)** A escolha dos tamanhos mínimo e máximo dos nós é adequada para o tamanho das chaves? Explique sua resposta.

**Resposta**

**Questão 4 (2,5 pontos)** Considere tabelas de dispersão estendidas tais que:

* Após aplicar a função de hash a uma chave, os bits são considerados da direita para a esquerda.
* Cada bloco pode conter um mínimo de 1 e um máximo de 3 chaves.

*T*

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

**Figura 3. Representação parcial de uma tabela de dispersão.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *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* | |
|  |  |  |  |  | |

**Figura 4. Representação esquemática de uma tabela de dispersão.**

a) **(0,5 ponto)** Considere a tabela de dispersão estendida, *T*, da Figura 3. Qual o valor de *j1* para o bloco *B1*? Explique sua resposta.

b) **(0,5 ponto)** Considere a tabela de dispersão estendida, *T*, da Figura 3. Quantos bits estão sendo considerados e qual ou quais são as possíveis combinações dos últimos bits das chaves que são armazenadas no bloco *B1*? Explique sua resposta.

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

d) **(1,0 ponto)** Considere a tabela de dispersão estendida da Figura 4. Mostre como a estrutura ficará após as remoções, sucessivamente, de *K41* e *K42*. Explique sua resposta.

**Resposta**