Aluno(a):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ matrícula:\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| 1a) | 2.0 |  |
| 2a) | 2.0 |  |
| 3a) | 3.0 |  |
| 4a) | 3.0 |  |
|  |  |  |

1. A prova é individual e sem consulta.
2. A interpretação faz parte da questão.
3. O tempo de prova é 1:30 h.
4. As respostas devem seguir as questões.
5. Caso precise de rascunho use o verso da folha.
6. Caso parte da resposta esteja no verso, indique claramente este fato.
7. (**2.0 pontos**) Apresente cada um dos movimentos que ocorrem ao se remover o elemento do topo do seguinte heap armazenado em um vetor.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **posição** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| **elemento** | T | S | P | G | R | O | N | A | E | C | B | I | F |

Resp.:

**2)** **(2.0 pontos)** A seguinte função realiza inserção de chaves em uma tabela de dispersão, utilizando desempate interno de colisões:

#define MAX 8 /\* tamanho da tabela de hash \*/

#define DEC 2

#define VAZIO (-1)

int hash[MAX];

int insere (int x)

{

int k, pos;

pos = x % MAX;

for (k = 0; k < MAX; k++)

{

if (hash[pos] == VAZIO) /\* entrada vazia \*/

{

hash[pos] = x; /\* insere a chave \*/

return (pos);

}

if (hash[pos] == x) /\* chave duplicada \*/

return (pos);

pos = pos - DEC; /\* colisão \*/

if (pos < 0)

pos = pos + MAX;

}

return (-1); /\* não inseriu \*/

}

1. **(1.0 ponto)** Partindo de uma tabela vazia (todas as posições contêm –1) e utilizando a rotina dada, realize a inserção das seguintes chaves: 18, 28, 42, 78 e 32, desenhando a tabela a cada inserção.
2. **(1.0 ponto)** Escreva a expressão matemática da função de dispersão H(x,k) implicitamente utilizada na rotina acima. Por que H(x,k) não é uma boa função de dispersão? Justifique sua resposta.

Resp.:

**3)** **(3.0 pontos)** Considere a árvore B de ordem 4 (ou seja, todo nó tem nó máximo 4 filhos e 3 chaves) abaixo:

100

50 75 120 200

10 40 60 70 80 90 110 115 130 135 140 220 230 240

Realize as seguintes operações, utilizando sempre a árvore resultante da operação anterior. Redesenhe a árvore a cada passo, indicando os nós que sofrem modificações.

a) **(1.0 ponto)** Inserção de 150.

b) **(1.0 ponto)** Remoção de 120 (após a inserção de 150).

c) **(1.0 ponto)** Remoção de 100 (após a inserção de 150 e a remoção de 120).

Resp.:

4) (**3.0 pontos**) Considere o grafo abaixo.

**6**

4

12

22

24

25

10

7

16

28

3

4

5

2

0

1

1. (**1,0 ponto**) Represente o grafo por listas de adjacências (note que o grafo é não dirigido).
2. (**1,0 ponto**) Liste os índices na ordem que eles são visitados em uma busca em profundidade (depth-first search) começando pelo nó 1. Assuma que os vizinhos de um nó são visitados em ordem crescente.
3. (**1,0 ponto**) Calcule as distâncias de todos os nós ao nó 6, calculadas pelo algoritmo de Dijkstra.

Resp.: