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

|  |  |  |
| --- | --- | --- |
| 1a) | 2,0 |  |
| 2a) | 2,0 |  |
| 3a) | 2,0 |  |
| 4a) | 2,0 |  |
| 5a) | 2,0 |  |
|  | 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 (2,0 pontos)** Uma tabela de dispersão (*hash table*) de tamanho 11 é implementada com *encadeamento externo* através da seguinte função de dispersão:   
Nela são inseridas 8 dados que possuem as seguintes chaves de busca (nesta ordem):

7,10,15,14,17,3,21,25

1. (0,5 ponto) Desenhe a estrutura de dados após a inserção destas chaves.
2. (1,0 ponto) Explique, com base no exemplo anterior, quais são os casos que devem ser considerados para implementar a operação de remoção de uma chave x:

remocao(x)

Entrada: um valor x de chave

Saída: NULL, se x não é encontrada

p, ponteiro para o elemento que contém x

1. (0,5 ponto) Qual a complexidade temporal, no pior caso, da operação de remoção? Explique sua resposta.

**Resposta**

**Questão 2 (2,0 pontos)** Considere a seguinte sequencia de inteiros: 90, 60, 30, 15, 45.

* 1. (0,5 ponto) Mostre, passo a passo, como o vetor armazenando um *heap* mínimo é construído pela inserção sucessiva destes 5 elementos, na ordem dada. Comente brevemente cada passo do algoritmo de inserção.
  2. (1,0 ponto) Mostre, passo a passo, como fica o vetor após a remoção apenas do segundo menor elemento do *heap* construído no Item (a).
  3. (0,5 ponto) Sugira um algum algoritmo mais eficiente para construção de um *heap* mínimo com estes mesmos inteiros e comente se ao final o vetor ficará exatamente igual ou não ao do item (a)?

**Resposta**

**Questão 3 (2,0 pontos)** Implemente de forma não recursiva, visitando o menor número possível de nós, uma função que calcule a *largura* de uma árvore de busca binária, definida como a diferença entre o maior e o menor valor dentre os valores de chave na árvore. Por simplicidade, assuma que os valores de chave são inteiros. A função deve ter o seguinte protótipo:

int abb\_largura (Abb\* r)**;**

A função recebe como entrada um apontador para a raiz da árvore e retorna a largura da árvore.

Adote a seguinte estrutura para os nós:

typedef struct \_abb Abb;

struct \_abb {

int chave;

Abb\* esq;

Abb\* dir;

};

**Resposta**

**Questão 4 (2,0 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. (0,5 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.

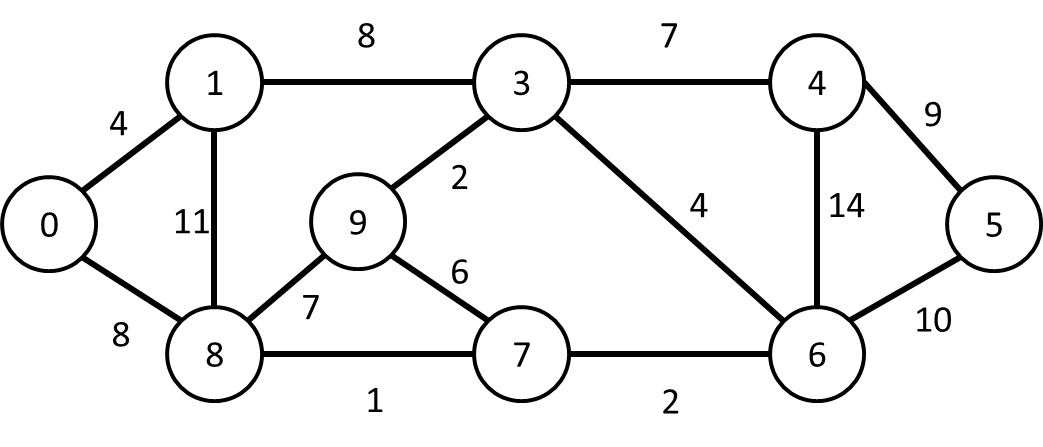
**Resposta**

**Questão 5** **(2,0 pontos)**

(a) (1,0 ponto) Mostre os passos do algoritmo de Kruskal para calcular uma árvore geradora mínima do grafo mostrado na figura abaixo. Considere que o algoritmo adota uma partição dinâmica dos nós do grafo, representada por uma floresta, com a implementação de UNION por altura e FIND com compressão de caminhos.

**Respostas que não utilizarem uma partição dinâmica como pedido não serão consideradas.**

(b) (1,0 ponto) Explique qual a vantagem, em termos de custo do processamento, de adotar uma partição dinâmica dos nós do grafo, representada por uma floresta, com a implementação de UNION por altura e FIND com compressão de caminhos?



**Resposta**