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

|  |  |  |
| --- | --- | --- |
| 1a) | 1.5 |  |
| 2a) | 1.5 |  |
| 3a) | 2.0 |  |
| 4a) | 2.5 |  |
| 5a) | 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 (1.5 pontos)**

a) **(1.0 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 pontos)** Considere a tabela de dispersão estendida, *T*, representada na figura abaixo. Suponha que cada bloco possa conter até 3 chaves. Mostre como a estrutura ficará após a remoção de *K31*, assumindo que a estrutura deve economizar espaço sempre que possível. Explique sua resposta.

*T*

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

Resposta:

**Questão 2 (1.5 pontos)**

1. **(0.5 pontos)** Crie uma árvore 2-3 através da inserção, na ordem, das chaves: 1, 12, 8, 2, 6. Mostre todos os passos.
2. **(1.0 ponto)** Remova a chave 3 da árvore 2-3 abaixo de 3 formas diferentes. Mostre e explique todos os passos.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 3 | 8 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 |  | 6 |  |  | 32 |  |

Resposta:

**Questão 3 (2.0 pontos)** Escreva uma rotina em C que receba como entrada um *min heap* e um float *p* maior do que 1.0, modifique a prioridade *x* do menor elemento do *min heap* para (*p* × *x*) e reorganize a estrutura para que mantenha as propriedades de um *min heap*. A rotina deve retornar 1, se a alteração ocorreu corretamente, e 0, em caso contrário.

A sua implementação deve fazer o menor número possível de comparações e movimentações. A implementação não deve usar as operações sobre *min heaps* pré-definidas, mas poderá usar a função troca definida abaixo.

Assuma que o *min heap* seja definida como abaixo:

struct \_heap {

int max; /\* tamanho maximo do heap \*/

int pos; /\* proxima posicao disponivel no vetor \*/

float\* prioridade; /\* vetor das prioridades \*/

};

typedef struct \_heap Heap;

Assuma ainda que a assinatura da rotina seja:

int heap\_altera(Heap\* heap, float p)

A função troca é definida como:

static void troca(int a, int b, float\* v) {

float f = v[a];

v[a] = v[b];

v[b] = f;

}

Resposta:

**Questão 4** **(2.5 pontos)** Considere o problema de representar a que partido político cada pessoa de um conjunto de pessoas pertence. Assuma que: (1) uma pessoa só pertence a um único partido em um dado instante; (2) dois partidos *A* e *B* podem se fundir (ou seja, para efeitos desta questão, as pessoas que pertencem a *A* ou *B* passam a pertencer ao mesmo partido).

a) **(0.5 pontos)** Qual estrutura de dados, dentre as apresentadas na disciplina, seria mais eficiente para representar a situação acima e implementar as rotinas dos itens (b) e (c)? Explique sua resposta. Defina em C a estrutura de dados escolhida, ou seja, o typedef, struct, etc... necessários às rotinas dos itens (b) e (c).

b) **(1.0 pontos)** Usando a estrutura de dados escolhida no item (a), implemente em C uma rotina para determinar se duas pessoas são do mesmo partido. A rotina não deverá chamar as operações definidas em sala para a estrutura escolhida.

c) **(1.0 pontos)** Usando a estrutura de dados escolhida no item (a), implemente em C uma rotina para fundir dois partidos, como definido acima. A rotina não deverá chamar as operações definidas em sala para a estrutura escolhida.

Resposta:

**Questão 5 (2.5 pontos)** Implemente uma função em C que receba como entrada um grafo não dirigido *G=(V,E)*, com pelo menos 1 nó, e retorne o diâmetro do grafo. A função deve ter o seguinte protótipo:

int diametro(Graph\* G)

Considere que o grafo está representado pela sua matriz de adjacências:

typedef struct graph Graph;

struct graph {

int nv; /\* número de nós no grafo \*/

int\*\* adj; /\* matriz de adjacências do grafo \*/

};

A função deverá se basear no algoritmo de grafos apresentado em sala que melhor se adequa para a tarefa, supondo que o grafo é esparso, ou seja, *G=(V,E)* é tal que |*E*| é muito menor do que |*V*|2.

Não é necessário incluir o algoritmo na solução, mas indique o protótipo do algoritmo adotado e qualquer outra estrutura de dados necessária para a sua implementação. A sua implementação não deverá modificar a estrutura de dados passada como entrada.

Argumente porque a sua implementação da função diametro está correta, prove qual é o custo da sua implementação e argumente porque o algoritmo escolhido é o que melhor se adequa para a tarefa.

Resposta: