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

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

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

**Questão 1** (2.0 pontos): Os algoritmos abaixo são usados para resolver problemas de tamanho *n*. Determine a complexidade, no pior caso, de cada algoritmo. Explique sua resposta.

a) (1.0 ponto):

int funcao\_C(int n) {

for ( i=1; i < n; i \*= 2 ) {

for ( j = n; j > 0; j /= 2 ) {

for ( k = j; k < n; k += 2 ) {

sum += (-j \* k) << i/2;

}

}

}

}

b) (1.0 ponto):

int funcao\_C(int n) {

for( int i = 1; i <= n; i \*= 2 ) {

for( int j = 0; j < i; j++ ) {

for( int k = 0; k < n; k += 2 ) {

sum \*= k;

}

for( int k = 1; k < n; k \*= 2 ) {

sum /= k

}

}

}

}

**Resposta**

**Questão 2** (2.0 pontos): Uma tabela de dispersão de tamanho 11 é implementada com encadeamento interior, utilizando todo o espaço de endereçamento para tratar colisões (ou seja, sem área de overflow). A função de dispersão é a seguinte:

* 1. (0.5 ponto) Desenhe a estrutura de dados após a inserção das chaves (nesta ordem):

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

Explique como cada colisão foi tratada.

* 1. (1.0 ponto) Explique como deve ser implementada uma operação de remoção para que as operações de consulta, inserção e remoção feitas em seguida sejam executadas corretamente. Mostre como ficará a estrutura após a remoção da chave 3.
  2. (0.5 pontos) Na estrutura resultante do item (b), insira a chave 4.

**Resposta**

**Questão 3** (3.0 pontos): Considere a seguinte sequencia de inteiros: 90, 60, 30, 15.

* 1. (0.5 ponto) Mostre, passo a passo, como o vetor armazenando um *heap* mínimo é construído pela inserção sucessiva destes 4 elementos, na ordem dada. Comente brevemente cada passo do algoritmo de inserção.
  2. (1.0 ponto) Mostre, passo a passo, como o vetor armazenando um *heap* mínimo é construído de forma otimizada com estes 4 elementos. Comente brevemente cada passo do algoritmo otimizado de construção.
  3. (0.5 ponto) Considere o vetor construído no item (a). Mostre, passo a passo, como fica o vetor após a remoção do menor elemento e do segundo menor elemento do heap, usando o algoritmo de remoção apresentado em sala. Comente brevemente cada passo destas remoções.
  4. (1.0 ponto) Descreva como você implementaria a operação

remove(n, Heap\* h)

que remove e devolve os n menores elementos de um heap mínimo h, *sem usar um vetor auxiliar de* n *posições*.

**Resposta**

**Questão 4** (3.0 pontos): Considere o problema de representar as turmas de uma disciplina em um dado semestre. Assuma que cada aluno está matriculado em uma única turma. Considere duas operações: (1) determinar se dois alunos estão na mesma turma; (2) juntar duas turmas em uma só.

a) (1,0 ponto) Qual estrutura de dados, dentre as apresentadas em sala, seria mais eficiente para resolver o problema acima? Explique sua resposta.

c) (2,0 pontos) Defina em C a estrutura de dados escolhida no item (a) e implemente uma rotina para determinar se dois alunos estão na mesma turma. A rotina não deverá chamar as operações definidas em sala para a estrutura escolhida.

**Resposta**