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

|  |  |  |
| --- | --- | --- |
| 1a) | 1.0 |  |
| 2a) | 1.0 |  |
| 3a) | 3.0 |  |
| 4a) | 2.0 |  |
| 5a) | 3.0 |  |
|  | 10.0 |  |

1. A prova é individual e sem consulta. Qualquer tentativa de “cola” resultará na anulação da prova de ambos os alunos envolvidos.
2. A interpretação faz parte da questão. Em caso de dúvida escreva a dúvida e a sua interpretação na resposta. NÃO HÁ PERGUNTAS AO PROFESSOR DURANTE A PROVA.
3. O tempo de prova é 1:45 h.
4. As respostas devem seguir as questões. Caso precise de rascunho use o verso da folha.
5. A prova pode ser feita a lápis.

**Questão 1** **(1.0 ponto).** Qual a ordem da complexidade em tempo dos algoritmos A e B abaixo? Explique sua resposta.

**Respostas sem explicações não serão consideradas.**

1. (0.5 ponto)

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;

}

}

}

1. (0.5 ponto)

int B(int n) {

int i,j,sum = 0;

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

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

sum+=i+2\*j;

}

}

return sum;

}

**Resposta:**

**Questão 2** **(1.0 ponto).** Considere uma família de árvores definida de forma semelhante a árvores B, exceto que:

* Os nós possuem tamanho fixo em 72 bytes
* As chaves possuem tamanho variável, entre 4 bytes e 12 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. (0.5 ponto) Qual será o maior número de chaves, todas de menor tamanho, que é possível acomodar em uma árvore de altura 2? Explique cuidadosamente sua resposta.
  2. (0.5 ponto) Qual será o número mínimo de chaves, todas de maior tamanho, que uma árvore de altura 2 armazena? Explique cuidadosamente sua resposta.

**Respostas sem explicações não serão consideradas.**

**Resposta:**

**Questão 3 (3.0 pontos).** Escreva uma rotina em C para recuperar um ponto representado pela curva de Morton. A rotina deve ter o seguinte protótipo:

Ponto\* decodeMorton(int m)

A rotina:

* recebe como entrada um inteiro m, onde
  + m possui 16 bits
  + m codifica as coordenadas de um ponto p, começando com a coordenada x
* aloca e preenche as coordenadas do ponto p codificado em m
* devolve um ponteiro para o ponto p

Assuma, por simplicidade, que um ponto possui coordenadas inteiras, definidas pelas 8 bits mais à direita de números inteiros. O tipo Ponto é definido como

typedef struct ponto Ponto;

struct ponto {

int x; /\* usa apenas os 8 bits mais à direita do int \*/

int y; /\* usa apenas os 8 bits mais à direita do int \*/

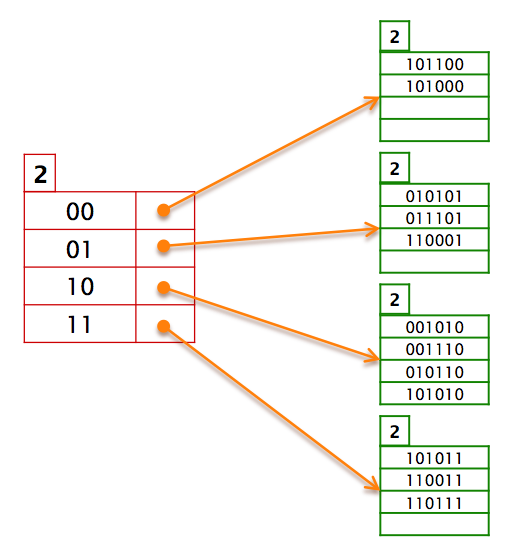
}

**Resposta:**

**Questão 4 (2.0 pontos).**

a) **(1.0 ponto)** Considere a estrutura de hash expansível abaixo. Realize a remoção da chave 49 de duas maneiras diferentes. Indique quais transformações são realizadas. Explique em que cenários cada alternativa seria a melhor.

**Respostas que não indicarem as transformações não serão consideradas.**



b) **(1.0 ponto)** Considere a árvore 2-3 abaixo. Realize a remoção da chave 100 de duas maneiras diferentes. Indique quais transformações são realizadas.

**Respostas que não indicarem as transformações não serão consideradas.**

100

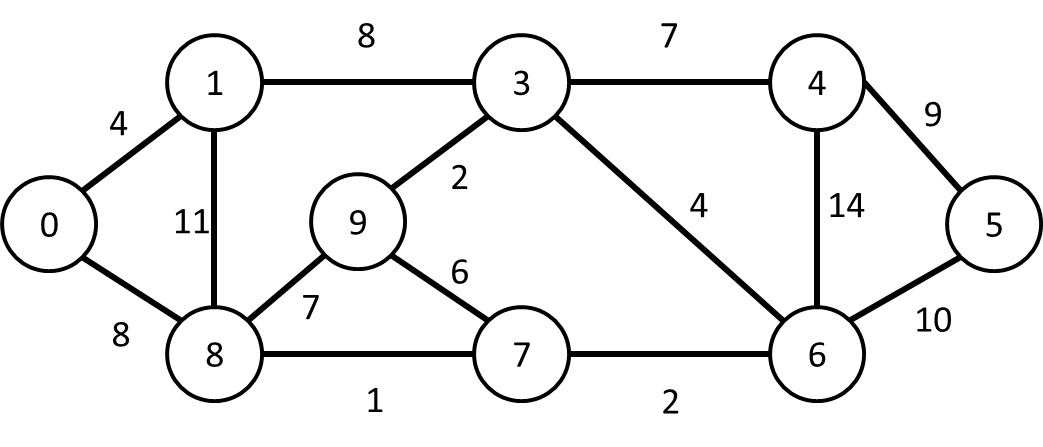
53 77 123 200

10 40 60 70 80 90 110 170 230

**Resposta:**

**Questão 5** **(3.0 pontos).** 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.**



**Resposta:**