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

|  |  |  |
| --- | --- | --- |
| 1a) | 2,5 |  |
| 2a) | 2,5 |  |
| 3a) | 2,5 |  |
| 4a) | 2,5 |  |
|  | 10.0 |  |

1. A prova é individual e sem consulta.
2. A interpretação faz parte da questão. Em caso de dúvida escreva a dúvida e a sua interpretação na resposta.
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.
6. (2.5 pontos)
   1. (0,5 pontos) Explique como determinar se um grafo não dirigido é conectado ou não. Use a figura abaixo para ilustrar a sua solução.

0

1

2

3

4

5

6

7

* 1. (2,0 pontos) Implemente uma função em C que determine se um grafo não dirigido é conectado ou não. A função deve ter o seguinte protótipo:

int connected(Graph\* G);

Considere que o grafo está representado como listas de adjacências:

typedef struct graph Graph;

typedef struct listNode ListNode;

struct listNode {

int vertex;

ListNode\* link;

};

struct graph {

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

int\* vis; /\* vis[i]=1 sse vertice i foi visitado \*/

ListNode\*\* vv; /\* vv[i] aponta p/ lista de vértices adjacentes a i \*/

};

Resposta:

1. Basta iniciar um percurso em profundidade (ou em amplitude) a partir de qualquer nó. O grafo será conectado sse todos os nós forem marcados como visitados.
2. O programa utiliza a rotina dfs apresentada em sala:

void dfs(Graph\* g, int N){

ListNode\* w;

g->vis[N] = 1;

for (w = g->vv[N]; w != NULL; w = w->link)

if (!g->vis[w->vertex]) dfs(g,w->vertex);

}

int conectado(Graph\* g) {

int i;

for (i=1; i<g->nv; i=i+1) g->vis[i]=0;

dfs(g,0);

for (i=1; i<g->nv; i=i+1)

if (vis[i]==0) return 0;

return 1;

}

1. (2.5 pontos) Considere um conjunto de cidades e de estradas que as ligam. Suponha que todas as estradas sejam de terra batida e de mão dupla. O grafo abaixo representa as cidades e estradas, onde o peso de cada aresta indica o comprimento da estrada em quilômetros.

0

1

2

3

4

2

10

7

5

6

4

3

1. (2,0 pontos) Determine quais estradas deveriam ser asfaltadas de tal forma que: (a) seja possível ir de qualquer cidade a qualquer outra cidade rodando apenas em estradas asfaltadas; (b) o menor número possível de quilômetros seja asfaltado. Explique a sua resposta.
2. (0,5 pontos) No caso geral, onde o problema é representado por um grafo não dirigido conectado, a solução do problema sempre será única? Explique sua resposta.

Resp:

a) Basta aplicar o algoritmo de Kruskal ao grafo pois o menor número possível de quilômetros a serem asfaltados corresponde à árvore geradora mínima do grafo.

0

1

2

3

4

2

1

7

5

6

4

3

b) A solução, em geral, não será única pois a árvore geradora mínima de um grafo não é única.

1. (2.5 pontos) Considere o grafo abaixo.

0

1

2

3

4

2

10

7

5

6

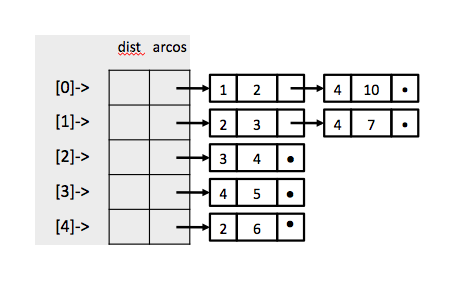
4

3

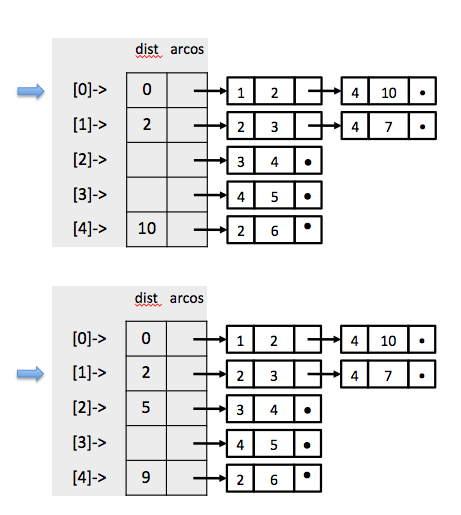
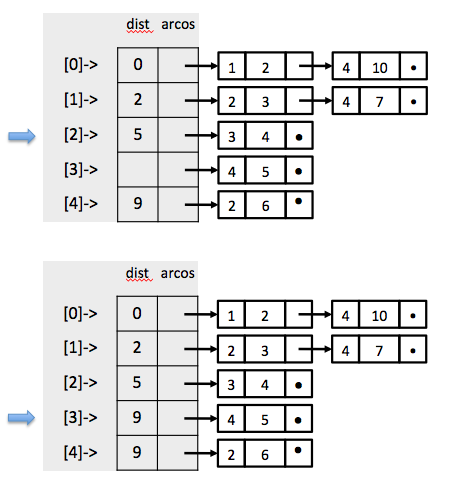
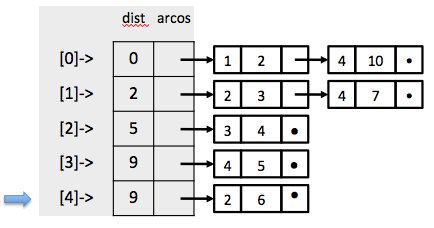
1. (1,0 ponto) Represente o grafo por listas de adjacências (note que o grafo é dirigido).
2. (1,5 ponto) Determine o menor caminho partindo do vértice 0 para o vértice 3 usando o algoritmo de Dijkstra. Mostre passo-a-passo como o algoritmo funciona.

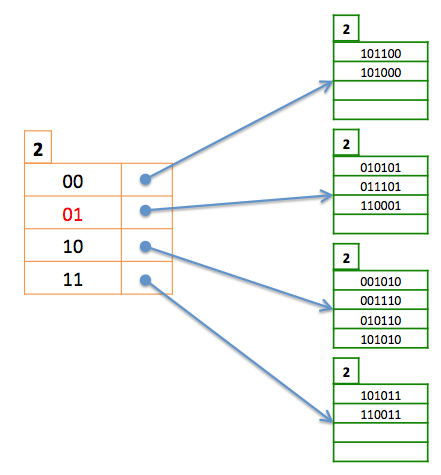
Resp.

(a)



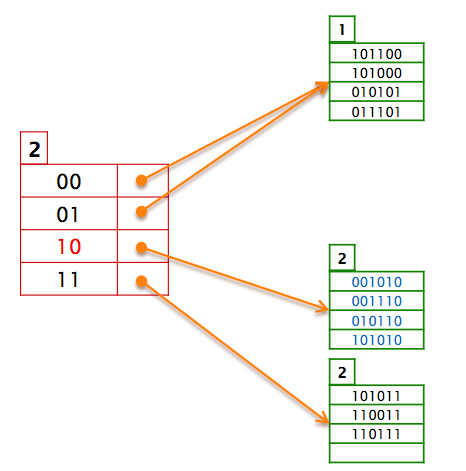
b)

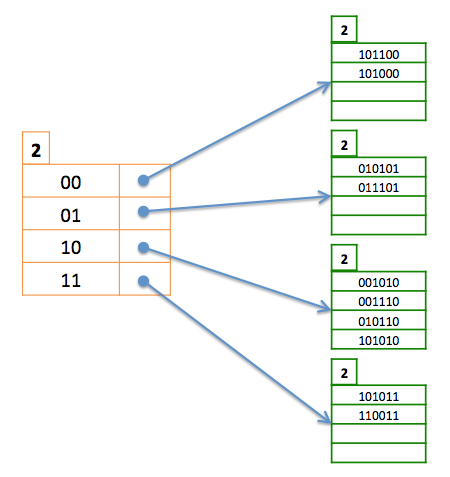


1. (2.5 pontos) Considere a estrutura de hash expansível abaixo:
2. (1.5 ponto) Apresente duas alternativas para remover a chave 110001, mostrando passo-a-passo as transformações na estrutura.
3. (1.0 ponto) Argumente sob que condições cada solução será a melhor.

Resp.:

(a) A chave é removida do segundo bucket e os dois primeiros buckets são combinados. Alternativamente, poderíamos simplesmente remover a chave do segundo bucket, mantendo os dois buckets.





(b) A primeira alternativa seria preferível se o número esperado de remoções for bem maior do que o de inserções. A segunda alternativa seria melhor se o número esperado de inserções e remoções for equilibrado.