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

|  |  |  |
| --- | --- | --- |
| 1a) | 3.0 |  |
| 2a) | 2.0 |  |
| 3a) | 2.0 |  |
| 3a) | 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.
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 (3.0 pontos).** Implemente uma função em C que receba como entrada um grafo não dirigido *G*, com pelo menos 1 nó, e retorne

1 se há um caminho entre quaisquer pares de nós de *G*

0 se não há um caminho entre algum par de nós de *G*

-1 em caso de erro

A função deve ter o seguinte protótipo:

int teste(Graph\* G);

Considere que o grafo está representado por 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 \*/

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

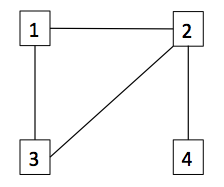
};

**Resposta:**

**Questão 2 (2.0 pontos).** Seja *G* um grafo não dirigido com *n* nós. Seja *MG* a matriz de incidênciade *G*. Considere a sequência *T1*,..., *Tk* ,... de matrizes *n* × *n* definida como:

* *T1* = *MG*
* *Tk* = *Tk-1* × *MG*, para *k >* 1

1. **(1.0 ponto)** Construa a matriz *T2*, definida como acima, para o grafo da Figura 1, considerando os nós na ordem indicada na figura.



**Figura 1**: Exemplo de grafo.

1. **(1.0 ponto)** Sejam *Ni* e *Nj* dois nós de *G*. Seja *T2* = a matriz definida como acima. Se , o que podemos dizer acerca de *Ni* e *Nj* ?

**Resposta:**

**Questão 3 (2.0 pontos).** Considere o grafo dirigido abaixo.

0

1

2

3

4

2

10

7

5

6

4

3

a) **(0.5 ponto)** Represente o grafo por listas de adjacências.

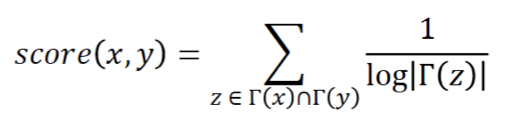
b) **(0.5 pontos)** Liste os nós na ordem que serão visitados se for realizada uma busca em amplitude (breadth-first search) começando pelo nó 0. Assuma que os vizinhos de um nó são escolhidos em ordem crescente. Mostre todos os passos durante o processamento do algoritmo.

c) **(0.5 pontos)** Liste os nós na ordem que serão visitados se for realizada uma busca em profundidade (depth-first search) começando pelo nó 0. Assuma que os vizinhos de um nó são escolhidos em ordem crescente. Mostre todos os passos durante o processamento do algoritmo.

d) **(0.5 pontos)** Discuta se há alguma forma de otimizar a busca em amplitude.

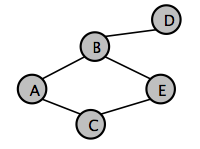
**Resposta:**

**Questão 4 (3.0 pontos).** Dado um grafo não dirigido, o *coeficiente de Adamic-Adar* de dois nós *x* e *y*, *score(x,y)*, é definido como (log na base 10):



onde Γ(*n*) denota o conjunto de nós ligados por arestas a um nó *n* e |*s*| denota a cardinalidade de um conjunto *s*.

a) **(1.0 ponto)** Compute *score(A,D)* e *score(A,E)* para o grafo abaixo.



b) **(2.0 pontos)** Implemente uma função em C que receba como entrada um grafo não dirigido *G* e dois nós, *x* e *y*, do grafo *G* e retorne o coeficiente de Adamic-Adar de *x* e *y*. Considere que o grafo está representado por 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 \*/

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

};

A função deve ter o seguinte protótipo:

float adamic-adar(Graph\* G, int x, int y);

onde x e y são dois inteiros indicando a posição dos nós no vetor vv.

**Resposta:**