# Lista de Exercícios para G2

Entrega no dia da prova (acrescentando 1.0 na nota), escrito à mão.

1. Considere a representação de uma árvore B a seguir, de ordem MAX:

typedef struct arvb ArvB;

struct arvb {

int \*k[MAX-1]; // chaves

ArvB \*sub[MAX]; // filhos

};

Escreva uma função que avalia se a árvore foi construída corretamente, ou seja, para duas chaves **k[i-1]** e **k[i]**, as chaves associadas no nó **sub[i]** devem ser menores que **k[i]** e maiores que **k[i-1]** (lembre-se de considerar o caso especial das extremidades). Neste exercício, considere que todas as chaves possuem valores distintos, e use INT\_MIN e INT\_MAX como o menor e maior valor possível de uma chave. Retorne 1 caso tenha sido construída corretamente, 0 caso contrário.

int earvb (ArvB \*b);

1. Considere cada um dos caracteres da palavra P E S Q U I S A como sendo informações que devam ser armazenadas em uma tabela de dispersão (*hash table*) utilizando a função **h(caractere) = (ordem de caractere no alfabeto) % 7, m = 7**

Desenhe como ficaram as tabelas de dispersão, cada uma com utilizando:

* 1. Encadeamento exterior
  2. Encadeamento interior (utilizando a estrutura do lab)

Caso um caractere já tenha sido inserido na tabela, o mesmo deve ser desconsiderado.

1. Considerando o grafo direcionado abaixo, responda as questões a seguir:

A diagram of a network

Description automatically generated

1. Mostre como seria a sua representação por matriz de adjacências.
2. Mostre como seria a sua representação por lista de adjacências.
3. Liste a ordem de visitação a partir de uma busca em profundidade iniciando no vértice 1 (considerando a ordem crescente de visitação).
4. Liste a ordem de visitação a partir de uma busca em largura iniciando no vértice 1 (considerando a ordem crescente de visitação).
5. Em laboratório, utilizamos a estrutura de união e busca para calcular a árvore geradora mínima:

UniaoBusca\* ub\_cria (int tam);

int ub\_busca (UniaoBusca\* ub, int u);

int ub\_uniao (UniaoBusca\* ub, int u, int v);

void ub\_libera (UniaoBusca\* ub);

Suponha que exista um método que implementa o algoritmo de Kruskal. Porém, ao invés de retornar a árvore geradora mínima, retorna a estrutura de união e busca gerada:

UniaoBusca\* kruskalUniaoBusca (Grafo\* g);

A partir disso, escreva uma função que verifica se o grafo g é conexo. Retorne 1 caso seja conexo, 0 caso contrário.

int econexo (Grafo\* g);

1. Considere que temos um tabuleiro formado por 3x4 posições, cada uma contendo um número:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 4 | 3 | 6 |
| 7 | 8 | 6 | 8 |
| 2 | 3 | 1 | 8 |

O objetivo consiste em deslocar um peão do canto superior esquerdo até o canto inferior direito, através de uma sequência de movimentos para a direita ou para baixo, minimizando o somatório dos pontos correspondentes às quadrículas por onde passou.

A partir disso, formule este problema como um grafo e determine o caminho mínimo. Como ficariam as estruturas auxiliares de custo mínimo e vértice antecessor?