Relatório Atividade 1

Integrante: Klaus de Freitas Dornsbach
Matrícula: 18100536

1 Representação

Foi utilizada a linguagem de programação Python devido a facilidade de implementação. Como estrutura de dados para armazenar o grafo foi utilizado uma lista de adjacência de hash maps, para minimizar o uso de memória e garantir O(1) para buscar a existência de um arco. Foi armazenada a quantidade de vertices numa variavel n_vertices, assim como a quantidade de arestas, o grau do vertice foi armazenado na própria estrutura "Node", assim como o rótulo. Vizinhos é o retorno das chaves do hash map dos nodos, peso é o valor da chave.

2 Buscas

Implementou-se o algoritmo de busca em largura, utilizando a estrutura "deque" do Python como fila. Para imprimir foi criada uma lista de listas com 2 informações: distancia (camada) e que elemento é esse (id, representado pela posição no vértice), então a lista foi ordenada pelo atributo distancia e em seguida imprimida.

3 Ciclo Euleriano

Neste problema, foi utilizado o algoritmo de Hierholzer, ignorando o peso dos arcos e colocando um valor significativo no seu lugar: se estes arcos foram ou não visitados. A implementação foi feita com base no algoritmo visto em sala, somente adaptando a minha estrutura de dados do grafo.

4 Bellman-Ford

Foram feitas iterações e relaxações através de todos os vértices e todos os arcos do grafo e em seguida foi feito o teste de ciclos negativos. Foi utilizado esse algoritmo pela simplicidade e por possuir tolerância a valores negativos nos pesos dos arcos

5 Floyd-Warshall

Foi feito um "try catch" para tolerar chaves não pertencentes aos hash maps dos vértices, assim simplificando a implementação da operação de popular a matriz inicialmente.