Trabalho Prático 2 – Árvore Geradora Mínima

Problema: Rede grande de cabos no DCC, ligando várias salas. Quer-se reduzir a quantidade de fios (visando economia). Para isso, procura-se a melhor forma possível de ligar todas as salas usando a menor quantidade possível de cabos.

Solução: Trata-se de um problema de grafo (vértices são as salas do DCC e arestas são os cabos que ligam cada sala) e a solução é dada por meio da árvore geradora mínima (menor árvore que consegue conectar todos os vértices do grafo).

Algoritmo: Usa-se o algoritmo de Kruskal

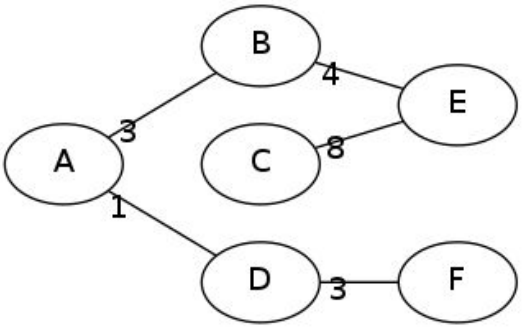
- https://pt.wikipedia.org/wiki/Algoritmo_de_Kruskal
- https://www.youtube.com/watch?v=T_O0pKbGcX0

Entrada: Recebe-se por parâmetro um arquivo conforme mostra a figura 1 na coluna "Entrada". Ela é composta por uma linha inicial contendo o número de vértices (v = 9) e o número de arestas (a = 13). Em seguida, há um número de linhas igual a **a,** representando cada aresta. cada linha é composta pelo valor do primeiro vértice, do segundo vértice e do seu peso, conforme mostra a figura 2.

Figura 1 - Entrada e saída do programa

Entrada	Saída
9 13 0 1 4 0 7 8 1 2 8 2 3 7 2 8 2 2 5 4 3 4 9 3 5 14 4 5 10 5 6 2 6 7 1 6 8 6 7 8 7	9 8 0 1 4 0 7 8 2 3 7 2 5 4 2 8 2 3 4 9 5 6 2 6 7 1

Figura 2 - Grafo com pesos



Saída: A saída deve ser dada em um arquivo contendo, basicamente, as mesmas coisas da entrada: uma linha com o número de vértices e o de arestas e as demais linhas representando cada aresta. Esses dados, contudo, são do novo grafo que será criado: o grafo da árvore geradora mínima para o grafo inicial. Um exemplo de saída é mostrado na figura 1.

Tipo Abstrato de Dados:
Tipo Vértice:

- Valor
- Filho
- Pai
- Peso relativo ao seu pai

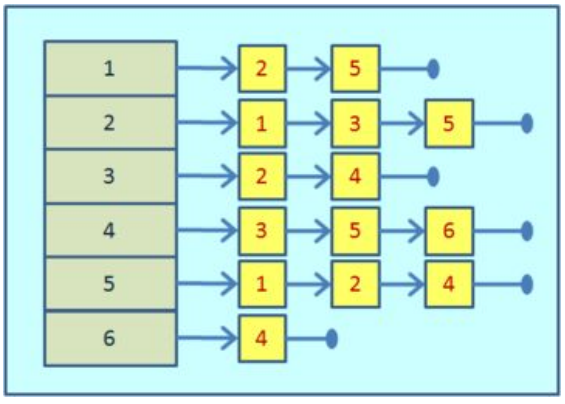
Tipo Aresta:

- Vértice 1
- Vértice 2
- Peso

Representação do Grafo: Lista de adjacências (recomendo falar sobre isso na documentação).

- https://pt.wikipedia.org/wiki/Lista_de_adjac%C3%AAncia
- http://esborniacomputacional.blogspot.com/2010/05/lista-de-adjacencias-em-c.html

Figura 3 - Lista de Adjacências



Desenvolvimento da solução: Crio um vetor de arestas, coloco nele cada aresta lida do arquivo. Em seguida, ordeno pelo peso das arestas (conforme deve ser feito no algoritmo de Kruskal). Depois de ordenadas, eu pego uma por uma e insiro seus vértices na lista de adjacências. Antes de inserir, porém, eu descubro se já existe uma maneira de alcançar um vértice a partir de outro (isso significa que eles pertencem à mesma árvore). Se isso acontecer, eu pulo aquela aresta e olho para a próxima. Do contrário, ou seja, se o par de vértices não pertencer à mesma árvore, eu os insiro na lista de adjacências (insiro o vértice 1 na lista do vértice 2 e insiro o vértice 2 na lista do vértice 1).
Terminado o processo de inserções, eu tenho exatamente as arestas da árvore geradora mínima. Pego cada par de vértices da lista (lembrando que um par de vértices representa uma aresta) e coloco em um novo vetor de arestas. Feito isso, imprimo no arquivo de saída e acabou o trabalho.

Funções e complexidade:
Como cada trabalho é diferente, vou colocar aqui as funções que são realmente importantes e que usei em todos (funções como as de imprimir a lista na tela não são necessárias, coloquei apenas para fins de teste, e não é necessário colocar a complexidade delas).
Obs.: **n** é o número de arestas da árvore

1. Função para pesquisar na lista de adjacências: O(n) no pior caso, pois olha apenas uma vez para cada vértice a cada execução.
2. Função para inserir na lista de adjacências: O(n) no pior caso, pois passa por cada vértice da lista de um vértice somente uma vez (pois já entra ordenando, no caso da Carla e da Karyne). Para o Douglas, a complexidade é O(1), pois insere sem ordenar.
3. Função para ordenar um vetor de arestas pelo peso: O(nlogn), pois usa quicksort, e essa é a complexidade de tal método.
4. No caso da versão do Douglas, utilizo outra função de ordenação para ordenar os vértices. A complexidade é a mesma do quicksort (O(nlogn)).

A complexidade geral do trabalho é a mesma do algoritmo de Kruskal: O(nlogn).

Testes: Usei o exemplo que veio na especificação.

Figura 4 - Entrada e saída do caso de teste