Parte II.
2ª VA

Implementação

Regras:

- O trabalho consistem em implementar 5 algoritmos em grafos em C ou C++.
- Os algoritmos serão i) `kosaraju`, ii) `prim`, iii) `kruskal`, iv) `dijkstra` e v) um algoritmo de sua escolha.
- O último algoritmo escolhido não poderá ser: busca em largura, busca em profundidade e ordenação topológica.
- Cada algoritmo deve ter sua própria pasta, com o código e um makefile.

Árvore Geradora Mínima

Para o Problema da Árvore Geradora Mínima a entrada deve possuir o seguinte formato:

```
Lista de adjacencia de um grafo G com 6 vértice e 8 arestas (3a coluna é o peso da aresta)6
8
1 2 5
1 3 4
1 4 2
1 6 6
2 4 1
2 5 7
3 5 6
4 6 1
```

O nome binário deverá ser o nome do algoritmo (e.g., `prim`). Para o problema da AGM os algoritmos devem possuir os seguintes parâmetros:

```
-h                      : mostra o help
-o <arquivo>            : redireciona a saida para o ‘‘arquivo’’
-f <arquivo>            : indica o ‘‘arquivo’’ que contém o grafo de entrada
-s                      : mostra a solução
-i                      : vértice inicial (para o algoritmo de Prim)
```

Exemplos de execução:

```
Calcula o custo da AGM com o grafo de entrada "arquivo-entrada.dat" e vértice inicial 1.
$ ./prim -f arquivo-entrada.dat -i 1
14
```

```
Imprime a árvore do exemplo anterior
$ ./prim -f arquivo-entrada.dat -i 1 -s
(1,3) (1,4) (2,4) (3,5) (4,6)

Note que a solução não precisa estar em ordem.
(4,1) (6,1) (4,2) (3,4) (5,6)
Também seria uma solução válida.
```

Componentes fortemente conexas

Para o problema de componentes fortemente conexos o algoritmo de kosaraju deve possuir os seguintes parâmetros:

```
 -h                    : mostra o help
 -o <arquivo>          : redireciona a saida para o ‘‘arquivo’’
 -f <arquivo>          : indica o ‘‘arquivo’’ que contém o grafo de entrada
```

Exemplos de execução:

```
Imprime as componentes fortemente conexas do grafo

arquivo-entrada.dat
12 17
1 2
1 4
2 3
3 1
3 7
4 6
5 4
6 7
7 5
8 6
8 11
9 8
10 9
11 10
11 12
12 10
12 7


$ ./kosaraju -f arquivo-entrada.dat
8 9 10 11 12
1 3 2
7 6 4 5
```

Caminhos Mínimos

Para o problema de caminho mínimo o algoritmo de dijkstra deve possuir os seguintes parâmetros:

```
 -h                    : mostra o help
 -o <arquivo>          : redireciona a saida para o ‘‘arquivo’’
 -f <arquivo>          : indica o ‘‘arquivo’’ que contém o grafo de entrada
 -i                    : vértice inicial
```

Exemplos de execução:

```
Imprime a distância do vértice inicial 1 até os demais
$ ./djikstra -f arquivo-entrada.dat -i 1
1:0 2:3 3:4 4:2 5:10 6:3
```

Caso um vértice seja inalcançável a partir do vértice inicial, o algoritmo deve mostrar apresentar o valor -1. No site da disciplina, será disponibilizado um conjunto de arquivos de teste e suas repectivas saidas (Bat1).

Critérios de Avaliação.

- Cada algoritmo será avaliado por uma nota de 0 a 10.
- A nota final será a médias das notas.
- Os 4 primeiros algoritmos serão avaliados por duas baterias de testes Bat1 e Bat2, em que Bat1 será fornecido.

- O 5º algoritmo deve vir acompanhado de um conjunto com pelo menos 20 entradas de tamanhos similares às fornecidas em `Bat1`.
- O algoritmo receberá 10 ou 0 se passar por `Bat1` ou não.
- O algoritmo que passar por `Bat1` terá pontos descontados com os seguintes critérios:
  - Implementação ineficiente (-1 ponto).
  - Não passar por Bat2 (-3 pontos).
  - Não utilização das estruturas de dados corretas (-1 ponto).
  - Não possuir makefile (-1 ponto).
  - Não for estruturado em uma pasta própria (-1 ponto).
- O 4º algoritmo será avaliado de forma análoga só que sem a bateria de teste.