Relatório Trabalho de Desenvolvimento
Programação Linear e Grafos

Geison Machado da Silva
Gabriel Lucas Teixeira Monteiro
Lucsan Rosa Machado

1 Análise do Problema

Podemos considerar cada cidade como um vértice e o caminho entre elas como as arestas. Assim para calcular a menor distância entre duas cidades basta aplicar o algorítimo de Floyd na matriz iterando 19 vezes, porque são 20 cidades.

2 Resultados Obtidos

A matriz de até 19 passos ficou da seguinte maneira:

```
------------------------------------------------------------ D19 ------------------------------------------------------------
0000, 0670, 0619, 0861, 0978, 0211, 0539, 0767, 0504, 0313, 0792, 0456, 0382, 0399, 0643, 0681, 1030, 0794, 1112, 1120
0670, 0000, 0423, 0617, 0365, 0459, 0787, 1015, 0752, 0357, 1040, 0880, 0806, 0823, 1067, 1105, 1454, 1218, 1536, 1544
0619, 0423, 0000, 0554, 0359, 0408, 0736, 0964, 0701, 0306, 0989, 0881, 0807, 0824, 1068, 1106, 1455, 1219, 1537, 1545
0861, 0617, 0554, 0000, 0913, 0758, 1086, 1314, 1051, 0656, 1339, 1231, 1157, 1174, 1418, 1456, 1805, 1569, 1887, 1895
0978, 0365, 0359, 0913, 0000, 0767, 1095, 1323, 1060, 0665, 1348, 1240, 1166, 1183, 1427, 1465, 1814, 1578, 1896, 1904
0211, 0459, 0408, 0758, 0767, 0000, 0328, 0556, 0293, 0102, 0581, 0473, 0399, 0416, 0660, 0698, 1047, 0811, 1129, 1137
0539, 0787, 0736, 1086, 1095, 0328, 0000, 0228, 0035, 0430, 0253, 0663, 0727, 0720, 0731, 1026, 1118, 1139, 1457, 1465
0767, 1015, 0964, 1314, 1323, 0556, 0228, 0000, 0263, 0658, 0025, 0659, 0733, 0716, 0952, 1032, 1339, 1145, 1463, 1471
0504, 0752, 0701, 1051, 1060, 0293, 0035, 0263, 0000, 0395, 0288, 0698, 0692, 0709, 0696, 0991, 1083, 1104, 1422, 1430
0313, 0357, 0306, 0656, 0665, 0102, 0430, 0658, 0395, 0000, 0683, 0575, 0501, 0518, 0762, 0800, 1149, 0913, 1231, 1239
0792, 1040, 0989, 1339, 1348, 0581, 0253, 0025, 0288, 0683, 0000, 0634, 0708, 0691, 0927, 1007, 1314, 1120, 1438, 1446
0456, 0880, 0881, 1231, 1240, 0473, 0663, 0659, 0698, 0575, 0634, 0000, 0074, 0057, 0301, 0373, 0688, 0486, 0804, 0812
0382, 0806, 0807, 1157, 1166, 0399, 0727, 0733, 0692, 0501, 0708, 0074, 0000, 0017, 0261, 0299, 0648, 0412, 0730, 0738
0399, 0823, 0824, 1174, 1183, 0416, 0720, 0716, 0709, 0518, 0691, 0057, 0017, 0000, 0244, 0316, 0631, 0429, 0747, 0755
0643, 1067, 1068, 1418, 1427, 0660, 0731, 0952, 0696, 0762, 0927, 0301, 0261, 0244, 0000, 0560, 0387, 0673, 0877, 0999
0681, 1105, 1106, 1456, 1465, 0698, 1026, 1032, 0991, 0800, 1007, 0373, 0299, 0316, 0560, 0000, 0503, 0113, 0431, 0439
1030, 1454, 1455, 1805, 1814, 1047, 1118, 1339, 1083, 1149, 1314, 0688, 0648, 0631, 0387, 0503, 0000, 0616, 0490, 0942
0794, 1218, 1219, 1569, 1578, 0811, 1139, 1145, 1104, 0913, 1120, 0486, 0412, 0429, 0673, 0113, 0616, 0000, 0544, 0326
1112, 1536, 1537, 1887, 1896, 1129, 1457, 1463, 1422, 1231, 1438, 0804, 0730, 0747, 0877, 0431, 0490, 0544, 0000, 0455
1120, 1544, 1545, 1895, 1904, 1137, 1465, 1471, 1430, 1239, 1446, 0812, 0738, 0755, 0999, 0439, 0942, 0326, 0455, 0000
```

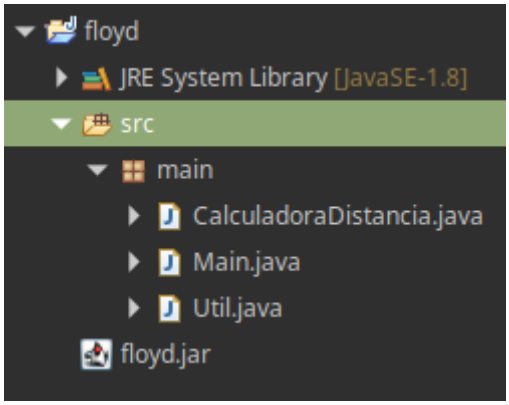
A matriz de roteamento de até 19 passos ficou da seguinte maneira:

```
------------------------------------------------------------ R19 ------------------------------------------------------------
 00,  05,  05,  03,  05,  05,  05,  05,  05,  05,  05,  12,  12,  12,  12,  12,  12,  12,  12,  12
 09,  01,  02,  03,  04,  09,  09,  09,  09,  09,  09,  12,  12,  12,  12,  12,  12,  12,  12,  12
 09,  01,  02,  03,  04,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09
 00,  01,  02,  03,  02,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09,  09
 02,  01,  02,  02,  04,  02,  02,  02,  02,  02,  02,  02,  02,  02,  02,  02,  02,  02,  02,  02
 00,  09,  09,  09,  09,  05,  08,  08,  08,  09,  08,  12,  12,  12,  12,  12,  12,  12,  12,  12
 08,  08,  08,  08,  08,  08,  06,  07,  08,  08,  07,  11,  08,  11,  08,  08,  08,  08,  08,  08
 06,  06,  06,  06,  06,  06,  06,  07,  06,  06,  10,  10,  10,  10,  10,  10,  10,  10,  10,  10
 05,  05,  05,  05,  05,  05,  06,  06,  08,  05,  06,  06,  05,  05,  14,  05,  14,  05,  05,  05
 05,  01,  02,  03,  02,  05,  05,  05,  05,  09,  05,  05,  05,  05,  05,  05,  05,  05,  05,  05
 07,  07,  07,  07,  07,  07,  07,  07,  07,  07,  10,  11,  11,  11,  14,  11,  14,  11,  11,  11
 13,  13,  13,  13,  13,  13,  06,  10,  06,  13,  10,  11,  13,  13,  13,  13,  13,  13,  13,  13
 00,  01,  05,  05,  05,  05,  05,  13,  05,  05,  13,  13,  12,  13,  13,  15,  13,  15,  15,  15
 12,  12,  12,  12,  12,  12,  11,  11,  12,  12,  11,  11,  12,  13,  14,  12,  14,  12,  12,  12
 13,  13,  13,  13,  13,  13,  08,  10,  08,  13,  10,  13,  13,  13,  14,  13,  16,  13,  16,  13
 12,  12,  12,  12,  12,  12,  12,  12,  12,  12,  12,  12,  12,  12,  12,  15,  16,  17,  18,  17
 14,  14,  14,  14,  14,  14,  14,  14,  14,  14,  14,  14,  14,  14,  14,  15,  16,  15,  18,  15
 15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  17,  15,  19
 15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  15,  16,  15,  16,  15,  18,  19
 17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  17,  18,  19
```

Assim, por exemplo, pode-se perceber que que para ir da cidade 1 até a 15 o custo é de 643 e deve-se passar pelas cidades 1 → 13 → 14 → 15. Olhando visualmente no mapa dá a impressão que não é o caminho mais curto, mas se somar os valores na D1 de 1 a 13, 13 a 14 e 14 a 15 pode-se ver que o custo é o menor.

3 Construção do Código

O programa foi codificado em Java e a estrutura do projeto ficou simples:



Utilizou-se apenas 3 classes dentro de um pacote chamado main:

- Main.java: Lê a entrada das duas cidades pelo usuário.
- Util.java: Classe utilitária que contém as matrizes *hardcoded* no código, métodos de impressão, etc.
- CalculdadoraDistancia.java: Classe princípal com o *core* do programa que contém o método principal *calculaDistancia()* que emula o algorítimo que vimos em aula para calcular Floyd.

No código fonte do programa foi todo documentado cada ação do algorítimo explicando como funciona.

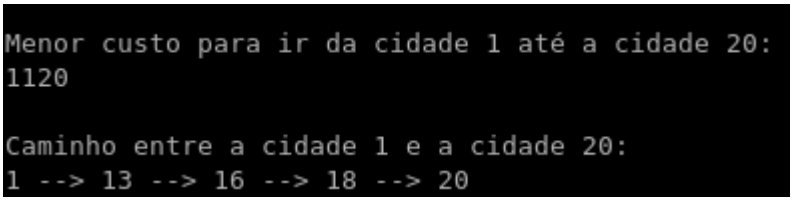
3 Execução do programa

Na pasta raiz do software tem um arquivo chamado **floyd.jar.**
Ele pode ser executado com o comando **java -jar floyd.jar.**

```
geison@geisonpc:~/wsp/floyd$ java -jar floyd.jar
Digite a primeira cidade (Um número inteiro de 1 a 20)
1
Digite a segunda cidade (Um número inteiro de 1 a 20)
20
```

Após pedir para digitar as duas cidades, o programa vai imprimir na tela todas as matrizes, de distância e roteamento, de D1 até D19, de R1 até R19.

No final ele diz o custo e o menor caminho para navegar entre as cidade que o usuário digitou:



Assim para qualquer funcionário saber quais cidades passar para ter o menor custo entre duas cidades, basta rodar o programa e digitar as duas cidade. O programa converte elas em vértices (subtrai 1), aplica o algorítimo de Floyd e imprime na tela o menor custo e o menor caminho.