

Relatório Projeto P2

Teoria dos Grafos

Aluno	TIA
Amanda Laís Xavier Fontes	31949436
Thiago Henrique Quadrado Estacio	42012740
Rafael Junqueira Pezeiro	32035901

GitHub: <https://github.com/Thiago2204/Projeto-Callisto>

Apresentação: <https://www.icloud.com/keynote/057uLVz98XDAUEwB896xOQnlw#Apresenta%C3%A7%C3%A3o>

Replit: <https://replit.com/join/jwbokpuyvpb-amandalais>

Vídeo: <https://www.youtube.com/watch?v=IOfozrUjphk>

Documentação de Documentação de Implementação – Projeto de Grafos (Parte 2)

Descrição do Projeto:

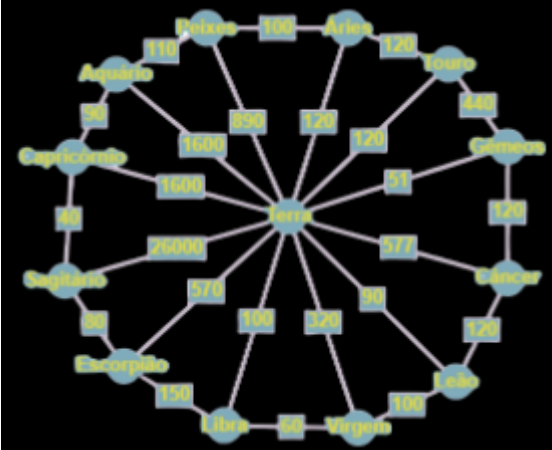
O Objetivo do Projeto consiste em criar uma rota de para que num futuro distante a raça humana possa atravessar a galáxia.

Esse projeto tem o objetivo de satisfazer os quesitos 4 e 9 da ODS:

- **ODS 9: Inovação infraestrutura – Construir infraestrutura resiliente, promover a industrialização inclusiva e sustentável, e fomentar a inovação.**
 - Nosso projeto atende os objetivos de infraestrutura resiliente e promove a industrialização inclusiva e sustentável. O quesito de infraestrutura é contemplado pela construção de meios de transporte que sejam capazes de realizar o percurso definido pelo nosso projeto, graças ao seu potencial de obrigar a indústria espacial a construir máquinas capazes de fazerem tais percursos atualmente impossíveis. Já na parte de infraestrutura, ao serem criadas essas máquinas atualmente inexistentes.
- **ODS 4: Educação de qualidade – Assegurar a educação inclusiva, equitativa e de qualidade, e promover oportunidades de aprendizagem ao longo da vida para todos.**
 - Propomos que as massas tenham uma melhor educação sobre os Cosmos, tendo um maior interesse pelas estrelas para que possamos criar novas gerações mais capacitadas e interessadas no assunto para que possa haver mais pesquisas no futuro, podendo, até, criar mais projetos que impulsionam a humanidade no futuro. Além disso, envia-se a identificação de padrões nas características de constelações, de maneira a aprender sobre os jeitos que as constelações foram definidas pelas culturas.

Testagem do Projeto

Grafos escolhidos para os testes

Grafo	txt
	<pre>1 1 2 12 3 22 4 0 1 120 5 0 2 440 6 0 3 51 7 0 4 577 8 0 5 90 9 0 6 320 10 0 7 100 11 0 8 570 12 0 9 26000 13 0 10 1600 14 0 11 890 15 1 2 120 16 2 3 440 17 3 4 120 18 4 5 100 19 5 6 60 20 6 7 150 21 7 8 80 22 8 9 40 23 9 10 40 24 10 11 110 25 11 1 100</pre>

Conexidade	Caminho mínimo
<pre>Pressione qualquer tecla para continuar... sh: 1: cls: not found ----- Opções ----- 1) Ler dados de um arquivo txt 2) Gravar dados no arquivo txt 3) Inserir vértice 4) Inserir aresta 5) Remover vértice 6) Remover aresta 7) Mostrar conteúdo do arquivo 8) Mostrar grafo 9) Verificar menor caminho 10) Verificar conexidade 11) Visualizar um percurso para a rota 12) Encerrar a aplicação - Escolha uma das opções acima: 10 Digite o número de um vértice: 0 É CONEXO -.- ★ -.-</pre>	<pre>sh: 1: cls: not found ----- Opções ----- 1) Ler dados de um arquivo txt 2) Gravar dados no arquivo txt 3) Inserir vértice 4) Inserir aresta 5) Remover vértice 6) Remover aresta 7) Mostrar conteúdo do arquivo 8) Mostrar grafo 9) Verificar menor caminho 10) Verificar conexidade 11) Visualizar um percurso para a rota 12) Encerrar a aplicação - Escolha uma das opções acima: 9 Digite o número de um vértice: 0 {0: 0, 1: 120, 2: 240, 3: 51, 4: 171, 5: 90, 6: 150, 7: 100, 8: 180, 9: 220, 10: 260, 11: 220} Pressione qualquer tecla para continuar... sh: 1: cls: not found ----- Opções ----- 1) Ler dados de um arquivo txt 2) Gravar dados no arquivo txt 3) Inserir vértice 4) Inserir aresta 5) Remover vértice 6) Remover aresta 7) Mostrar conteúdo do arquivo 8) Mostrar grafo 9) Verificar menor caminho 10) Verificar conexidade 11) Visualizar um percurso para a rota 12) Encerrar a aplicação - Escolha uma das opções acima: 9 Digite o número de um vértice: 4 {0: 171, 1: 291, 2: 411, 3: 120, 4: 0, 5: 100, 6: 160, 7: 271, 8: 351, 9: 391, 10: 431, 11: 391} Pressione qualquer tecla para continuar...</pre>

Grafo	txt
constelacao de touro.png	<pre> 1 11 10 0 1 2 1 2 5 2 3 5 3 4 1 4 5 1 5 6 2 5 7 2 6 8 6 3 9 3 9 10 8 </pre>

Caminho mínimo	Conexidade
<pre> Pressione qualquer tecla para continuar... sh: 1: cls: not found ----- Opções ----- 1) Ler dados de um arquivo txt 2) Gravar dados no arquivo txt 3) Inserir vértice 4) Inserir aresta 5) Remover vértice 6) Remover aresta 7) Mostrar conteúdo do arquivo 8) Mostrar grafo 9) Verificar menor caminho 10) Verificar conexidade 11) Visualizar um percurso para a rota 12) Encerrar a aplicação - Escolha uma das opções acima: 9 Digite o número de um vértice: 2 {0: 7, 1: 5, 2: 0, 3: 5, 4: 6, 5: 7, 6: 9, 7: 9, 8: 15, 9: 8, 10: 16} Pressione qualquer tecla para continuar... sh: 1: cls: not found ----- Opções ----- 1) Ler dados de um arquivo txt 2) Gravar dados no arquivo txt 3) Inserir vértice 4) Inserir aresta 5) Remover vértice 6) Remover aresta 7) Mostrar conteúdo do arquivo 8) Mostrar grafo 9) Verificar menor caminho 10) Verificar conexidade 11) Visualizar um percurso para a rota 12) Encerrar a aplicação - Escolha uma das opções acima: 9 Digite o número de um vértice: 1 {0: 2, 1: 0, 2: 5, 3: 10, 4: 11, 5: 12, 6: 14, 7: 14, 8: 20, 9: 13, 10: 21} </pre>	<pre> Pressione qualquer tecla para continuar... sh: 1: cls: not found ----- Opções ----- 1) Ler dados de um arquivo txt 2) Gravar dados no arquivo txt 3) Inserir vértice 4) Inserir aresta 5) Remover vértice 6) Remover aresta 7) Mostrar conteúdo do arquivo 8) Mostrar grafo 9) Verificar menor caminho 10) Verificar conexidade 11) Visualizar um percurso para a rota 12) Encerrar a aplicação - Escolha uma das opções acima: 10 Digite o número de um vértice: 0 É CONEXO -.- ★ -.- </pre>

Documentação de Implementação – Projeto de Grafos (Parte 1)

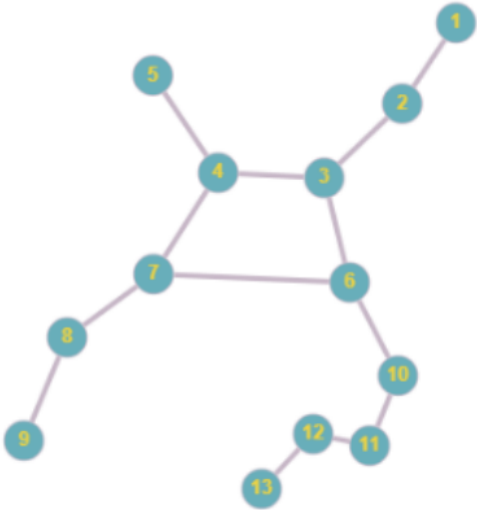
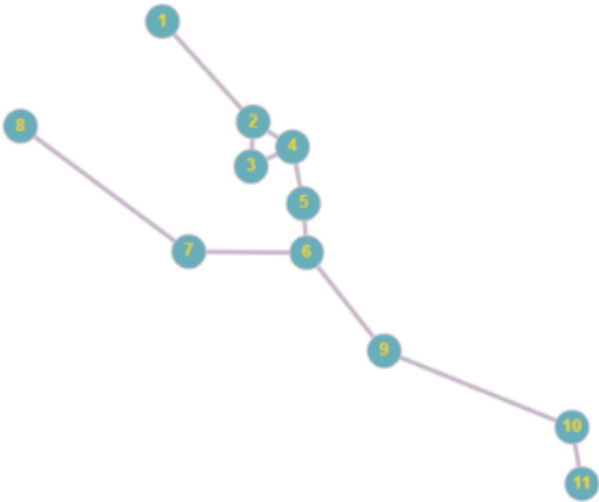
Descrição do Projeto

O Projeto Callisto possui como objetivo analisar a quantidade de desenhos possíveis a partir de um número n fixo de estrelas em uma imagem do espaço.

O software recebe uma imagem do céu noturno e usa as estrelas nela contidas como vértices que serão unidas e, a partir desta ligação, serão criadas formas, ou seja, constelações.

Testagem do Projeto

Constelações escolhidas para os testes

	
Constelação de Virgem ♍	Constelação de Touro ♉

Testes

Testes com a Constelação de Virgem

Opção 1 - Ler dados de um arquivo txt

..*..
Projeto Callisto
...*



Precione qualquer tecla para continuar...
sh: 1: cls: not found

```
----- Opções -----  
1) Ler dados de um arquivo txt  
2) Gravar dados no arquivo txt  
3) Inserir vértice  
4) Inserir aresta  
5) Remover vértice  
6) Remover aresta  
7) Mostrar conteúdo do arquivo  
8) Mostrar grafo  
9) Verificar menor caminho  
10) Verificar conexidade  
11) Visualizar um percurso para a rota  
12) Encerrar a aplicação  
- Escolha uma das opções acima: 1  
Digite o nome do arquivo: teste.txt  
Grafo recuperado de um arquivo
```

Opção 8 - Mostrar grafo

```

8) Mostrar grafo
9) Verificar menor caminho
10) Verificar conexidade
11) Visualizar um percurso para a rota
12) Encerrar a aplicação
- Escolha uma das opções acima: 8

n: 13 m: 13

Adj[ 0, 0] = 0 Adj[ 0, 1] = 3 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0 Adj[ 0,10] = 0 Adj[ 0,11] = 0 Adj[ 0,12] = 0
Adj[ 1, 0] = 3 Adj[ 1, 1] = 0 Adj[ 1, 2] = 1 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 0 Adj[ 1,10] = 2 Adj[ 1,11] = 0 Adj[ 1,12] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 1 Adj[ 2, 2] = 0 Adj[ 2, 3] = 2 Adj[ 2, 4] = 0 Adj[ 2, 5] = 1 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 0 Adj[ 2, 9] = 0 Adj[ 2,10] = 0 Adj[ 2,11] = 0 Adj[ 2,12] = 0
Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 2 Adj[ 3, 3] = 0 Adj[ 3, 4] = 2 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0 Adj[ 3,10] = 0 Adj[ 3,11] = 0 Adj[ 3,12] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 2 Adj[ 4, 4] = 0 Adj[ 4, 5] = 0 Adj[ 4, 6] = 0 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0 Adj[ 4,10] = 0 Adj[ 4,11] = 0 Adj[ 4,12] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 1 Adj[ 5, 3] = 0 Adj[ 5, 4] = 0 Adj[ 5, 5] = 0 Adj[ 5, 6] = 2 Adj[ 5, 7] = 0 Adj[ 5, 8] = 0 Adj[ 5, 9] = 0 Adj[ 5,10] = 2 Adj[ 5,11] = 0 Adj[ 5,12] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 0 Adj[ 6, 5] = 2 Adj[ 6, 6] = 0 Adj[ 6, 7] = 3 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0 Adj[ 6,10] = 0 Adj[ 6,11] = 0 Adj[ 6,12] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 0 Adj[ 7, 6] = 3 Adj[ 7, 7] = 0 Adj[ 7, 8] = 1 Adj[ 7, 9] = 0 Adj[ 7,10] = 0 Adj[ 7,11] = 0 Adj[ 7,12] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 0 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 1 Adj[ 8, 8] = 0 Adj[ 8, 9] = 2 Adj[ 8,10] = 0 Adj[ 8,11] = 0 Adj[ 8,12] = 0
Adj[ 9, 0] = 0 Adj[ 9, 1] = 0 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 0 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 2 Adj[ 9, 9] = 0 Adj[ 9,10] = 0 Adj[ 9,11] = 0 Adj[ 9,12] = 0
Adj[10, 0] = 0 Adj[10, 1] = 2 Adj[10, 2] = 0 Adj[10, 3] = 0 Adj[10, 4] = 0 Adj[10, 5] = 2 Adj[10, 6] = 0 Adj[10, 7] = 0 Adj[10, 8] = 0 Adj[10, 9] = 0 Adj[10,10] = 0 Adj[10,11] = 2 Adj[10,12] = 0
Adj[11, 0] = 0 Adj[11, 1] = 0 Adj[11, 2] = 0 Adj[11, 3] = 0 Adj[11, 4] = 0 Adj[11, 5] = 0 Adj[11, 6] = 0 Adj[11, 7] = 0 Adj[11, 8] = 0 Adj[11, 9] = 0 Adj[11,10] = 2 Adj[11,11] = 0 Adj[11,12] = 2
Adj[12, 0] = 0 Adj[12, 1] = 0 Adj[12, 2] = 0 Adj[12, 3] = 0 Adj[12, 4] = 0 Adj[12, 5] = 0 Adj[12, 6] = 0 Adj[12, 7] = 0 Adj[12, 8] = 0 Adj[12, 9] = 0 Adj[12,10] = 0 Adj[12,11] = 2 Adj[12,12] = 0

fim da impressao do grafo.

Precione qualquer tecla para continuar...[]

```

Opção 5 - Remover vértice (com auxílio da 8 para mostrar como ficou)

```

Precione qualquer tecla para continuar...
sh: 1: cls: not found
----- Opções -----
1) Ler dados de um arquivo txt
2) Gravar dados no arquivo txt
3) Inserir vértice
4) Inserir aresta
5) Remover vértice
6) Remover aresta
7) Mostrar conteúdo do arquivo
8) Mostrar grafo
9) Verificar menor caminho
10) Verificar conexidade
11) Visualizar um percurso para a rota
12) Encerrar a aplicação
- Escolha uma das opções acima: 5
Informe qual vértice será removido: 7

```

Agora só existem 11 vértices:

```

Precione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 8 |

n: 12 m: 11

Adj[ 0, 0] = 0 Adj[ 0, 1] = 3 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0 Adj[ 0,10] = 0 Adj[ 0,11] = 0
Adj[ 1, 0] = 3 Adj[ 1, 1] = 0 Adj[ 1, 2] = 1 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 2 Adj[ 1,10] = 0 Adj[ 1,11] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 1 Adj[ 2, 2] = 0 Adj[ 2, 3] = 2 Adj[ 2, 4] = 0 Adj[ 2, 5] = 1 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 0 Adj[ 2, 9] = 0 Adj[ 2,10] = 0 Adj[ 2,11] = 0
Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 2 Adj[ 3, 3] = 0 Adj[ 3, 4] = 2 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0 Adj[ 3,10] = 0 Adj[ 3,11] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 2 Adj[ 4, 4] = 0 Adj[ 4, 5] = 0 Adj[ 4, 6] = 0 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0 Adj[ 4,10] = 0 Adj[ 4,11] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 1 Adj[ 5, 3] = 0 Adj[ 5, 4] = 0 Adj[ 5, 5] = 0 Adj[ 5, 6] = 2 Adj[ 5, 7] = 0 Adj[ 5, 8] = 0 Adj[ 5, 9] = 2 Adj[ 5,10] = 0 Adj[ 5,11] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 0 Adj[ 6, 5] = 2 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0 Adj[ 6,10] = 0 Adj[ 6,11] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 0 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 2 Adj[ 7, 9] = 0 Adj[ 7,10] = 0 Adj[ 7,11] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 0 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 2 Adj[ 8, 8] = 0 Adj[ 8, 9] = 0 Adj[ 8,10] = 0 Adj[ 8,11] = 0
Adj[ 9, 0] = 0 Adj[ 9, 1] = 2 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 2 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 0 Adj[ 9, 9] = 0 Adj[ 9,10] = 2 Adj[ 9,11] = 0
Adj[10, 0] = 0 Adj[10, 1] = 0 Adj[10, 2] = 0 Adj[10, 3] = 0 Adj[10, 4] = 0 Adj[10, 5] = 0 Adj[10, 6] = 0 Adj[10, 7] = 0 Adj[10, 8] = 0 Adj[10, 9] = 2 Adj[10,10] = 0 Adj[10,11] = 2
Adj[11, 0] = 0 Adj[11, 1] = 0 Adj[11, 2] = 0 Adj[11, 3] = 0 Adj[11, 4] = 0 Adj[11, 5] = 0 Adj[11, 6] = 0 Adj[11, 7] = 0 Adj[11, 8] = 0 Adj[11, 9] = 0 Adj[11,10] = 2 Adj[11,11] = 0

fim da impressao do grafo.

Precione qualquer tecla para continuar...

```

Opção 6 - Remover aresta (com auxílio da 8)

```

Precione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 6 |
Informe o primeiro vértice da ligação será removida: 1
Informe o segundo vértice da ligação será removida: 2

```

Adj [1, 2] = 1 e Adj [2, 1] = 1 passam a ser Adj [1, 2] = 0 e Adj [2, 1] = 0:

```

Precione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 8 |
|
n: 12 m: 10

Adj[ 0, 0] = 0 Adj[ 0, 1] = 3 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0 Adj[ 0,10] = 0 Adj[ 0,11] = 0
Adj[ 1, 0] = 3 Adj[ 1, 1] = 0 Adj[ 1, 2] = 0 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 2 Adj[ 1,10] = 0 Adj[ 1,11] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 0 Adj[ 2, 2] = 0 Adj[ 2, 3] = 2 Adj[ 2, 4] = 0 Adj[ 2, 5] = 1 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 0 Adj[ 2, 9] = 0 Adj[ 2,10] = 0 Adj[ 2,11] = 0
Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 2 Adj[ 3, 3] = 0 Adj[ 3, 4] = 2 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0 Adj[ 3,10] = 0 Adj[ 3,11] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 2 Adj[ 4, 4] = 0 Adj[ 4, 5] = 0 Adj[ 4, 6] = 0 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0 Adj[ 4,10] = 0 Adj[ 4,11] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 1 Adj[ 5, 3] = 0 Adj[ 5, 4] = 0 Adj[ 5, 5] = 0 Adj[ 5, 6] = 2 Adj[ 5, 7] = 0 Adj[ 5, 8] = 0 Adj[ 5, 9] = 2 Adj[ 5,10] = 0 Adj[ 5,11] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 0 Adj[ 6, 5] = 2 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0 Adj[ 6,10] = 0 Adj[ 6,11] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 0 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 2 Adj[ 7, 9] = 0 Adj[ 7,10] = 0 Adj[ 7,11] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 0 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 2 Adj[ 8, 8] = 0 Adj[ 8, 9] = 0 Adj[ 8,10] = 0 Adj[ 8,11] = 0
Adj[ 9, 0] = 0 Adj[ 9, 1] = 2 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 2 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 0 Adj[ 9, 9] = 0 Adj[ 9,10] = 2 Adj[ 9,11] = 0
Adj[10, 0] = 0 Adj[10, 1] = 0 Adj[10, 2] = 0 Adj[10, 3] = 0 Adj[10, 4] = 0 Adj[10, 5] = 0 Adj[10, 6] = 0 Adj[10, 7] = 0 Adj[10, 8] = 0 Adj[10, 9] = 2 Adj[10,10] = 0 Adj[10,11] = 2
Adj[11, 0] = 0 Adj[11, 1] = 0 Adj[11, 2] = 0 Adj[11, 3] = 0 Adj[11, 4] = 0 Adj[11, 5] = 0 Adj[11, 6] = 0 Adj[11, 7] = 0 Adj[11, 8] = 0 Adj[11, 9] = 0 Adj[11,10] = 2 Adj[11,11] = 0

fim da impressão do grafo.
Precione qualquer tecla para continuar...

```

Opção 3 - Inserir vértice (com auxílio da 8)

```

Precione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 3 |
|
SUCESSO NA OPERAÇÃO :D
>

```

Passa a ter 12 vértices novamente:


```
| 8) Mostrar grafo          |
| 9) Verificar menor caminho |
| 10) Verificar conexidade  |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação  |
| - Escolha uma das opções acima: 8 |

n: 13 m: 10

Adj[ 0, 0] = 0 Adj[ 0, 1] = 3 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0 Adj[ 0,10] = 0 Adj[ 0,11] = 0 Adj[ 0,12] = 0

Adj[ 1, 0] = 3 Adj[ 1, 1] = 0 Adj[ 1, 2] = 0 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 2 Adj[ 1,10] = 0 Adj[ 1,11] = 0 Adj[ 1,12] = 0

Adj[ 2, 0] = 0 Adj[ 2, 1] = 0 Adj[ 2, 2] = 0 Adj[ 2, 3] = 2 Adj[ 2, 4] = 0 Adj[ 2, 5] = 1 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 0 Adj[ 2, 9] = 0 Adj[ 2,10] = 0 Adj[ 2,11] = 0 Adj[ 2,12] = 0

Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 2 Adj[ 3, 3] = 0 Adj[ 3, 4] = 2 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0 Adj[ 3,10] = 0 Adj[ 3,11] = 0 Adj[ 3,12] = 0

Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 2 Adj[ 4, 4] = 0 Adj[ 4, 5] = 0 Adj[ 4, 6] = 0 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0 Adj[ 4,10] = 0 Adj[ 4,11] = 0 Adj[ 4,12] = 0

Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 1 Adj[ 5, 3] = 0 Adj[ 5, 4] = 0 Adj[ 5, 5] = 0 Adj[ 5, 6] = 2 Adj[ 5, 7] = 0 Adj[ 5, 8] = 0 Adj[ 5, 9] = 2 Adj[ 5,10] = 0 Adj[ 5,11] = 0 Adj[ 5,12] = 0

Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 0 Adj[ 6, 5] = 2 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0 Adj[ 6,10] = 0 Adj[ 6,11] = 0 Adj[ 6,12] = 0

Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 0 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 2 Adj[ 7, 9] = 0 Adj[ 7,10] = 0 Adj[ 7,11] = 0 Adj[ 7,12] = 0

Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 0 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 2 Adj[ 8, 8] = 0 Adj[ 8, 9] = 0 Adj[ 8,10] = 0 Adj[ 8,11] = 0 Adj[ 8,12] = 0

Adj[ 9, 0] = 0 Adj[ 9, 1] = 2 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 2 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 0 Adj[ 9, 9] = 0 Adj[ 9,10] = 2 Adj[ 9,11] = 0 Adj[ 9,12] = 0

Adj[10, 0] = 0 Adj[10, 1] = 0 Adj[10, 2] = 0 Adj[10, 3] = 0 Adj[10, 4] = 0 Adj[10, 5] = 0 Adj[10, 6] = 0 Adj[10, 7] = 0 Adj[10, 8] = 0 Adj[10, 9] = 2 Adj[10,10] = 0 Adj[10,11] = 2 Adj[10,12] = 0

Adj[11, 0] = 0 Adj[11, 1] = 0 Adj[11, 2] = 0 Adj[11, 3] = 0 Adj[11, 4] = 0 Adj[11, 5] = 0 Adj[11, 6] = 0 Adj[11, 7] = 0 Adj[11, 8] = 0 Adj[11, 9] = 0 Adj[11,10] = 2 Adj[11,11] = 0 Adj[11,12] = 0

Adj[12, 0] = 0 Adj[12, 1] = 0 Adj[12, 2] = 0 Adj[12, 3] = 0 Adj[12, 4] = 0 Adj[12, 5] = 0 Adj[12, 6] = 0 Adj[12, 7] = 0 Adj[12, 8] = 0 Adj[12, 9] = 0 Adj[12,10] = 0 Adj[12,11] = 0 Adj[12,12] = 0

fim da impressão do grafo.

Precione qualquer tecla para continuar...
```

Opção 4 - Inserir aresta (com auxílio da 8)

```
Precione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 4 |
Informe o primeiro dos vértices que serão interligados:
0
Informe o segundo dos vértices que serão interligados:
3
Informe o custo da ligação (pode ser em ponto flutuante): 25
>
```

Adj [0, 3] = 0 e Adj [3, 0] = 0 passam a ser Adj [0, 3] = 25.0 e Adj [3, 0] = 25.0:

```
| 8) Mostrar grafo
| 9) Verificar menor caminho
| 10) Verificar conexidade
| 11) Visualizar um percurso para a rota
| 12) Encerrar a aplicação
| - Escolha uma das opções acima: 8
|
n: 13 m: 11
Adj[ 0, 0] = 0 Adj[ 0, 1] = 3 Adj[ 0, 2] = 0 Adj[ 0, 3] = 25.0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0 Adj[ 0,10] = 0 Adj[ 0,11] = 0 Adj[ 0,12] = 0
Adj[ 1, 0] = 3 Adj[ 1, 1] = 0 Adj[ 1, 2] = 0 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 2 Adj[ 1,10] = 0 Adj[ 1,11] = 0 Adj[ 1,12] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 0 Adj[ 2, 2] = 0 Adj[ 2, 3] = 2 Adj[ 2, 4] = 0 Adj[ 2, 5] = 1 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 0 Adj[ 2, 9] = 0 Adj[ 2,10] = 0 Adj[ 2,11] = 0 Adj[ 2,12] = 0
Adj[ 3, 0] = 25.0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 2 Adj[ 3, 3] = 0 Adj[ 3, 4] = 2 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0 Adj[ 3,10] = 0 Adj[ 3,11] = 0 Adj[ 3,12] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 2 Adj[ 4, 4] = 0 Adj[ 4, 5] = 0 Adj[ 4, 6] = 0 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0 Adj[ 4,10] = 0 Adj[ 4,11] = 0 Adj[ 4,12] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 1 Adj[ 5, 3] = 0 Adj[ 5, 4] = 0 Adj[ 5, 5] = 0 Adj[ 5, 6] = 2 Adj[ 5, 7] = 0 Adj[ 5, 8] = 0 Adj[ 5, 9] = 2 Adj[ 5,10] = 0 Adj[ 5,11] = 0 Adj[ 5,12] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 0 Adj[ 6, 5] = 2 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0 Adj[ 6,10] = 0 Adj[ 6,11] = 0 Adj[ 6,12] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 0 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 2 Adj[ 7, 9] = 0 Adj[ 7,10] = 0 Adj[ 7,11] = 0 Adj[ 7,12] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 0 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 2 Adj[ 8, 8] = 0 Adj[ 8, 9] = 0 Adj[ 8,10] = 0 Adj[ 8,11] = 0 Adj[ 8,12] = 0
Adj[ 9, 0] = 0 Adj[ 9, 1] = 2 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 2 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 0 Adj[ 9, 9] = 0 Adj[ 9,10] = 2 Adj[ 9,11] = 0 Adj[ 9,12] = 0
Adj[10, 0] = 0 Adj[10, 1] = 0 Adj[10, 2] = 0 Adj[10, 3] = 0 Adj[10, 4] = 0 Adj[10, 5] = 0 Adj[10, 6] = 0 Adj[10, 7] = 0 Adj[10, 8] = 0 Adj[10, 9] = 2 Adj[10,10] = 0 Adj[10,11] = 2 Adj[10,12] = 0
Adj[11, 0] = 0 Adj[11, 1] = 0 Adj[11, 2] = 0 Adj[11, 3] = 0 Adj[11, 4] = 0 Adj[11, 5] = 0 Adj[11, 6] = 0 Adj[11, 7] = 0 Adj[11, 8] = 0 Adj[11, 9] = 0 Adj[11,10] = 2 Adj[11,11] = 0 Adj[11,12] = 0
Adj[12, 0] = 0 Adj[12, 1] = 0 Adj[12, 2] = 0 Adj[12, 3] = 0 Adj[12, 4] = 0 Adj[12, 5] = 0 Adj[12, 6] = 0 Adj[12, 7] = 0 Adj[12, 8] = 0 Adj[12, 9] = 0 Adj[12,10] = 0 Adj[12,11] = 0 Adj[12,12] = 0

fim da impressao do grafo.

Precione qualquer tecla para continuar...|
```

Opção 7 - Mostrar conteúdo do arquivo

```
Precione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt
| 2) Gravar dados no arquivo txt
| 3) Inserir vértice
| 4) Inserir aresta
| 5) Remover vértice
| 6) Remover aresta
| 7) Mostrar conteúdo do arquivo
| 8) Mostrar grafo
| 9) Verificar menor caminho
| 10) Verificar conexidade
| 11) Visualizar um percurso para a rota
| 12) Encerrar a aplicação
| - Escolha uma das opções acima: 7
Digite o nome do arquivo: teste.txt
1
13
13
0 1 3
1 2 1
2 3 2
3 4 2
2 5 1
5 6 2
6 7 3
7 8 1
8 9 2
1 10 2
10 11 2
11 12 2
5 10 2

Precione qualquer tecla para continuar...|
```

Opção 2 - Gravar dados no arquivo txt

```
Pressione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 2 |
SUCESSO NA OPERAÇÃO :D
```

Grava em grafo.txt

Então, comparando:

```
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 7 |
Digite o nome do arquivo: teste.txt
1
13
13
0 1 3
1 2 1
2 3 2
3 4 2
2 5 1
5 6 2
6 7 3
7 8 1
8 9 2
1 10 2
10 11 2
11 12 2
5 10 2
```

```

sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 7 |
Digite o nome do arquivo: grafo.txt
1
13
11
0 1 3
0 3 25.0
1 0 3
1 9 2
2 3 2
2 5 1
3 0 25.0
3 2 2
3 4 2
4 3 2
5 2 1
5 6 2
5 9 2
6 5 2
7 8 2
8 7 2
9 1 2
9 5 2
9 10 2
10 9 2
10 11 2
11 10 2

```

Opção 12 - Encerrar a aplicação

```

Precione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
> | 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 12 |
* . * . * - * -

```

Opção 1



Opção 7

```
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
> | 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 7 |
Digite o nome do arquivo: teste.txt
1
11
10
0 1 2
1 2 5
2 3 5
3 4 1
4 5 1
5 6 2
5 7 2
6 8 6
3 9 3
9 10 8

Pressione qualquer tecla para continuar...|
```

Opção 8

```
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 8 |
n: 11 m: 10
> Adj[ 0, 0] = 0 Adj[ 0, 1] = 2 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0 Adj[ 0,10] = 0
Adj[ 1, 0] = 2 Adj[ 1, 1] = 0 Adj[ 1, 2] = 5 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 0 Adj[ 1,10] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 5 Adj[ 2, 2] = 0 Adj[ 2, 3] = 5 Adj[ 2, 4] = 0 Adj[ 2, 5] = 0 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 0 Adj[ 2, 9] = 0 Adj[ 2,10] = 0
Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 5 Adj[ 3, 3] = 0 Adj[ 3, 4] = 1 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 3 Adj[ 3,10] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 1 Adj[ 4, 4] = 0 Adj[ 4, 5] = 1 Adj[ 4, 6] = 0 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0 Adj[ 4,10] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 0 Adj[ 5, 3] = 0 Adj[ 5, 4] = 1 Adj[ 5, 5] = 0 Adj[ 5, 6] = 2 Adj[ 5, 7] = 2 Adj[ 5, 8] = 0 Adj[ 5, 9] = 0 Adj[ 5,10] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 0 Adj[ 6, 5] = 2 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 6 Adj[ 6, 9] = 0 Adj[ 6,10] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 2 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 0 Adj[ 7, 9] = 0 Adj[ 7,10] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 0 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 6 Adj[ 8, 7] = 0 Adj[ 8, 8] = 0 Adj[ 8, 9] = 0 Adj[ 8,10] = 0
Adj[ 9, 0] = 0 Adj[ 9, 1] = 0 Adj[ 9, 2] = 0 Adj[ 9, 3] = 3 Adj[ 9, 4] = 0 Adj[ 9, 5] = 0 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 0 Adj[ 9, 9] = 0 Adj[ 9,10] = 8
Adj[10, 0] = 0 Adj[10, 1] = 0 Adj[10, 2] = 0 Adj[10, 3] = 0 Adj[10, 4] = 0 Adj[10, 5] = 0 Adj[10, 6] = 0 Adj[10, 7] = 0 Adj[10, 8] = 0 Adj[10, 9] = 8 Adj[10,10] = 0

fim da impressao do grafo.

Pressione qualquer tecla para continuar...|
```

Opção 5

Pressione qualquer tecla para continuar...

sh: 1: cls: not found

```
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 5 |
Informe qual vértice será removido: 1
```

n deixa de ser 11 e passa a ser 10.

sh: 1: cls: not found

```
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 8 |
```

n: 10 m: 8

```
Adj[ 0, 0] = 0 Adj[ 0, 1] = 0 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0
Adj[ 1, 0] = 0 Adj[ 1, 1] = 0 Adj[ 1, 2] = 5 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 5 Adj[ 2, 2] = 0 Adj[ 2, 3] = 1 Adj[ 2, 4] = 0 Adj[ 2, 5] = 0 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 3 Adj[ 2, 9] = 0
Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 1 Adj[ 3, 3] = 0 Adj[ 3, 4] = 1 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 1 Adj[ 4, 4] = 0 Adj[ 4, 5] = 2 Adj[ 4, 6] = 2 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 0 Adj[ 5, 3] = 0 Adj[ 5, 4] = 2 Adj[ 5, 5] = 0 Adj[ 5, 6] = 0 Adj[ 5, 7] = 6 Adj[ 5, 8] = 0 Adj[ 5, 9] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 2 Adj[ 6, 5] = 0 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 6 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 0 Adj[ 7, 9] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 3 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 0 Adj[ 8, 8] = 0 Adj[ 8, 9] = 8
Adj[ 9, 0] = 0 Adj[ 9, 1] = 0 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 0 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 8 Adj[ 9, 9] = 0
```

fim da impressao do grafo.

Pressione qualquer tecla para continuar...

I

Opção 6


```

Pressione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 6 |
> Informe o primeiro vértice da ligação será removida: 8
  Informe o segundo vértice da ligação será removida: 9

```

m deixa de ser 8 e passa a ser 7.

```

sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 8 |
>
n: 10 m: 7

Adj[ 0, 0] = 0 Adj[ 0, 1] = 0 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0
Adj[ 1, 0] = 0 Adj[ 1, 1] = 0 Adj[ 1, 2] = 5 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 5 Adj[ 2, 2] = 0 Adj[ 2, 3] = 1 Adj[ 2, 4] = 0 Adj[ 2, 5] = 0 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 3 Adj[ 2, 9] = 0
Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 1 Adj[ 3, 3] = 0 Adj[ 3, 4] = 1 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 1 Adj[ 4, 4] = 0 Adj[ 4, 5] = 2 Adj[ 4, 6] = 2 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 0 Adj[ 5, 3] = 0 Adj[ 5, 4] = 2 Adj[ 5, 5] = 0 Adj[ 5, 6] = 0 Adj[ 5, 7] = 6 Adj[ 5, 8] = 0 Adj[ 5, 9] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 2 Adj[ 6, 5] = 0 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 6 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 0 Adj[ 7, 9] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 3 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 0 Adj[ 8, 8] = 0 Adj[ 8, 9] = 0
Adj[ 9, 0] = 0 Adj[ 9, 1] = 0 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 0 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 0 Adj[ 9, 9] = 0

fim da impressao do grafo.

Pressione qualquer tecla para continuar...

```

Opção 3


```
Pressione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 3 |
SUCESSO NA OPERAÇÃO :D
>
```

n deixa de ser igual a 10 e passa a ser 11.

```
Pressione qualquer tecla para continuar...
sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 8 |
n: 11 m: 7
Adj[ 0, 0] = 0 Adj[ 0, 1] = 0 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0 Adj[ 0,10] = 0
Adj[ 1, 0] = 0 Adj[ 1, 1] = 0 Adj[ 1, 2] = 5 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 0 Adj[ 1,10] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 5 Adj[ 2, 2] = 0 Adj[ 2, 3] = 1 Adj[ 2, 4] = 0 Adj[ 2, 5] = 0 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 3 Adj[ 2, 9] = 0 Adj[ 2,10] = 0
Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 1 Adj[ 3, 3] = 0 Adj[ 3, 4] = 1 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0 Adj[ 3,10] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 1 Adj[ 4, 4] = 0 Adj[ 4, 5] = 2 Adj[ 4, 6] = 2 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0 Adj[ 4,10] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 0 Adj[ 5, 3] = 0 Adj[ 5, 4] = 2 Adj[ 5, 5] = 0 Adj[ 5, 6] = 0 Adj[ 5, 7] = 6 Adj[ 5, 8] = 0 Adj[ 5, 9] = 0 Adj[ 5,10] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 2 Adj[ 6, 5] = 0 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0 Adj[ 6,10] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 6 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 0 Adj[ 7, 9] = 0 Adj[ 7,10] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 3 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 0 Adj[ 8, 8] = 0 Adj[ 8, 9] = 0 Adj[ 8,10] = 0
Adj[ 9, 0] = 0 Adj[ 9, 1] = 0 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 0 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 0 Adj[ 9, 9] = 0 Adj[ 9,10] = 0
Adj[10, 0] = 0 Adj[10, 1] = 0 Adj[10, 2] = 0 Adj[10, 3] = 0 Adj[10, 4] = 0 Adj[10, 5] = 0 Adj[10, 6] = 0 Adj[10, 7] = 0 Adj[10, 8] = 0 Adj[10, 9] = 0 Adj[10,10] = 0

fim da impressao do grafo.

Pressione qualquer tecla para continuar...█
```

Opção 4

```

sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 4 |
Informe o primeiro dos vértices que serão interligados:
5
Informe o segundo dos vértices que serão interligados:
9
Informe o custo da ligação (pode ser em ponto flutuante): 356

```

m deixa de ser 7 e passa a ser igual a 8.

```

sh: 1: cls: not found
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 8 |
n: 11 m: 8
>
Adj[ 0, 0] = 0 Adj[ 0, 1] = 0 Adj[ 0, 2] = 0 Adj[ 0, 3] = 0 Adj[ 0, 4] = 0 Adj[ 0, 5] = 0 Adj[ 0, 6] = 0 Adj[ 0, 7] = 0 Adj[ 0, 8] = 0 Adj[ 0, 9] = 0 Adj[ 0,10] = 0
Adj[ 1, 0] = 0 Adj[ 1, 1] = 0 Adj[ 1, 2] = 5 Adj[ 1, 3] = 0 Adj[ 1, 4] = 0 Adj[ 1, 5] = 0 Adj[ 1, 6] = 0 Adj[ 1, 7] = 0 Adj[ 1, 8] = 0 Adj[ 1, 9] = 0 Adj[ 1,10] = 0
Adj[ 2, 0] = 0 Adj[ 2, 1] = 5 Adj[ 2, 2] = 0 Adj[ 2, 3] = 1 Adj[ 2, 4] = 0 Adj[ 2, 5] = 0 Adj[ 2, 6] = 0 Adj[ 2, 7] = 0 Adj[ 2, 8] = 3 Adj[ 2, 9] = 0 Adj[ 2,10] = 0
Adj[ 3, 0] = 0 Adj[ 3, 1] = 0 Adj[ 3, 2] = 1 Adj[ 3, 3] = 0 Adj[ 3, 4] = 1 Adj[ 3, 5] = 0 Adj[ 3, 6] = 0 Adj[ 3, 7] = 0 Adj[ 3, 8] = 0 Adj[ 3, 9] = 0 Adj[ 3,10] = 0
Adj[ 4, 0] = 0 Adj[ 4, 1] = 0 Adj[ 4, 2] = 0 Adj[ 4, 3] = 1 Adj[ 4, 4] = 0 Adj[ 4, 5] = 2 Adj[ 4, 6] = 2 Adj[ 4, 7] = 0 Adj[ 4, 8] = 0 Adj[ 4, 9] = 0 Adj[ 4,10] = 0
Adj[ 5, 0] = 0 Adj[ 5, 1] = 0 Adj[ 5, 2] = 0 Adj[ 5, 3] = 0 Adj[ 5, 4] = 2 Adj[ 5, 5] = 0 Adj[ 5, 6] = 0 Adj[ 5, 7] = 6 Adj[ 5, 8] = 0 Adj[ 5, 9] = 356.0 Adj[ 5,10] = 0
Adj[ 6, 0] = 0 Adj[ 6, 1] = 0 Adj[ 6, 2] = 0 Adj[ 6, 3] = 0 Adj[ 6, 4] = 2 Adj[ 6, 5] = 0 Adj[ 6, 6] = 0 Adj[ 6, 7] = 0 Adj[ 6, 8] = 0 Adj[ 6, 9] = 0 Adj[ 6,10] = 0
Adj[ 7, 0] = 0 Adj[ 7, 1] = 0 Adj[ 7, 2] = 0 Adj[ 7, 3] = 0 Adj[ 7, 4] = 0 Adj[ 7, 5] = 6 Adj[ 7, 6] = 0 Adj[ 7, 7] = 0 Adj[ 7, 8] = 0 Adj[ 7, 9] = 0 Adj[ 7,10] = 0
Adj[ 8, 0] = 0 Adj[ 8, 1] = 0 Adj[ 8, 2] = 3 Adj[ 8, 3] = 0 Adj[ 8, 4] = 0 Adj[ 8, 5] = 0 Adj[ 8, 6] = 0 Adj[ 8, 7] = 0 Adj[ 8, 8] = 0 Adj[ 8, 9] = 0 Adj[ 8,10] = 0
Adj[ 9, 0] = 0 Adj[ 9, 1] = 0 Adj[ 9, 2] = 0 Adj[ 9, 3] = 0 Adj[ 9, 4] = 0 Adj[ 9, 5] = 356.0 Adj[ 9, 6] = 0 Adj[ 9, 7] = 0 Adj[ 9, 8] = 0 Adj[ 9, 9] = 0 Adj[ 9,10] = 0
Adj[10, 0] = 0 Adj[10, 1] = 0 Adj[10, 2] = 0 Adj[10, 3] = 0 Adj[10, 4] = 0 Adj[10, 5] = 0 Adj[10, 6] = 0 Adj[10, 7] = 0 Adj[10, 8] = 0 Adj[10, 9] = 0 Adj[10,10] = 0

fim da impressao do grafo.

Pressione qualquer tecla para continuar...

```

Opção 2

Pressione qualquer tecla para continuar...

sh: 1: cls: not found

```
| ----- Opções ----- |  
| 1) Ler dados de um arquivo txt |  
| 2) Gravar dados no arquivo txt |  
| 3) Inserir vértice |  
| 4) Inserir aresta |  
| 5) Remover vértice |  
| 6) Remover aresta |  
| 7) Mostrar conteúdo do arquivo |  
| 8) Mostrar grafo |  
| 9) Verificar menor caminho |  
| 10) Verificar conexidade |  
| 11) Visualizar um percurso para a rota |  
| 12) Encerrar a aplicação |  
| - Escolha uma das opções acima: 2 |  
SUCESSO NA OPERAÇÃO :D
```

Grava em grafo.txt, então, comparando:

```
Digite o nome do arquivo: teste.txt
1
11
10
0 1 2
1 2 5
2 3 5
3 4 1
4 5 1
5 6 2
5 7 2
6 8 6
3 9 3
9 10 8
```

Pressione qualquer tecla para continuar...

sh: 1: cls: not found

```
| ----- Opções ----- |
| 1) Ler dados de um arquivo txt |
| 2) Gravar dados no arquivo txt |
| 3) Inserir vértice |
| 4) Inserir aresta |
| 5) Remover vértice |
| 6) Remover aresta |
| 7) Mostrar conteúdo do arquivo |
| 8) Mostrar grafo |
| 9) Verificar menor caminho |
| 10) Verificar conexidade |
| 11) Visualizar um percurso para a rota |
| 12) Encerrar a aplicação |
| - Escolha uma das opções acima: 7 |
Digite o nome do arquivo: grafo.txt
```

```
1
11
8
1 2 5
2 1 5
2 3 1
2 8 3
3 2 1
3 4 1
4 3 1
4 5 2
4 6 2
5 4 2
5 7 6
5 9 356.0
6 4 2
7 5 6
8 2 3
9 5 356.0
```

Código

main.py

Menu do nosso projeto.

```
from grafoMatriz import TGrafoND
import time
import os
```

[illegible]

```

def show_opcoes():
    print("| ----- Opções ----- |")
    print("| 1) Ler dados de um arquivo txt |")
    print("| 2) Gravar dados no arquivo txt |")
    print("| 3) Inserir vértice |")
    print("| 4) Inserir aresta |")
    print("| 5) Remover vértice |")
    print("| 6) Remover aresta |")
    print("| 7) Mostrar conteúdo do arquivo |")
    print("| 8) Mostrar grafo |")
    print("| 9) Verificar menor caminho |")
    print("| 10) Verificar conexidade |")
    print("| 11) Visualizar um percurso para a rota |")
    print("| 12) Encerrar a aplicação |")

def recebe() -> int:
    return int(input("| - Escolha uma das opções acima: "))

def falha():
    print("FALHA NA OPERAÇÃO!! - Grafo inexistente.")

def sucesso():
    print("SUCESSO NA OPERAÇÃO :D")

def op1():
    return str(input("Digite o nome do arquivo: "))

def op2(grafo=None):
    grafo_arq(grafo)
    return True

def op3(grafo=None):
    if not grafo:
        return False
    grafo.insere_v()
    return grafo

def op4(grafo=None):
    if not grafo:
        return False
    v = int(input("Informe o primeiro dos vértices que serão interligados:\n"))
    w = int(input("Informe o segundo dos vértices que serão interligados:\n"))
    if grafo.rotulado:
        p = float(input("Informe o custo da ligação (pode ser em ponto flutuante):
"))
        grafo.insere_a(v, w, p)
        return grafo
    grafo.insere_a(v, w)
    return grafo

def op5(grafo):
    if not grafo:
        return False
    v = int(input("Informe qual vértice será removido: "))
    grafo.remover(v)
    return grafo

```

```
def op6(grafo):
    if not grafo:
        return False
    v = int(input("Informe o primeiro vértice da ligação será removida: "))
    w = int(input("Informe o segundo vértice da ligação será removida: "))
    grafo.remove_a(v, w)
    return grafo

def op7():
    with open(op1()) as file:
        print(file.read())

def op8(grafo):
    if not grafo:
        return False
    grafo.show()

def op9(grafo):
    if not grafo:
        return False
    v = int(input("Digite o número de um vértice: "))
    grafo.dijkstra(grafo, v)

def op10(grafo):
    if not grafo:
        return False
    v = int(input("Digite o número de um vértice: "))
    print(grafo.conexidade(grafo, v))

def op11(grafo):
    if not grafo:
        return False
    v = int(input("Digite o número do vértice inicial: "))
    perc = grafo.percurso_profundidade(v)
    grafo.percurso_feito(perc)

def menu():
    grafo = None
    saudacoes()
    while True:
        input("\n\n Precione qualquer tecla para continuar...")
        os.system('cls')
        show_opcoes()
        escolha = recebe()
        if escolha == 12:
            print(" * . * . * _` ★ `_" )
            return True
        elif escolha == 1:
            grafo = arq_grafo(op1())
            if grafo:
                print("Grafo recuperado de um arquivo")
        elif escolha == 2:
            if not op2(grafo):
                falha()
                continue
            sucesso()
        elif escolha == 3:
            if not grafo:
                falha()
                continue
            grafo = op3(grafo)
            sucesso()
        elif escolha == 4:
```

```

        if not grafo:
            falha()
            continue
        grafo = op4(grafo)
    elif escolha == 5:
        if not grafo:
            falha()
            continue
        grafo = op5(grafo)
    elif escolha == 6:
        if not grafo:
            falha()
            continue
        grafo = op6(grafo)
    elif escolha == 7:
        op7()
    elif escolha == 8:
        if not grafo:
            falha()
            continue
        op8(grafo)
    elif escolha == 9:
        if not grafo:
            falha()
            continue
        op9(grafo)
    elif escolha == 10:
        if not grafo:
            falha()
            continue
        op10(grafo)
    elif escolha == 11:
        if not grafo:
            falha()
            continue
        op11(grafo)

# MAIN -----
if __name__ == "__main__":
    menu()

```

grafoMatriz.py

Classe referente ao grafo em si e funções que o manipulam.

```

import math
from util import Pilha
from queue import PriorityQueue

# CLASSES
# grafo nao direcionado -- rotulado ou não

class TGrafoND:
    TAM_MAX_DEFAULT = 100
    def __init__(self, n=TAM_MAX_DEFAULT, rotulado=False):
        self.n = n #vertices
        self.m = 0 #arestas
        self.rotulado = False
        self.visitados = []
        if rotulado:
            self.rotulado = True
            self.adj = [[math.inf for i in range(n)] for j in range(n)]
        else:
            self.adj = [[0 for i in range(n)] for j in range(n)]

```



```

def insere_v(self):
    self.n += 1
    if self.rotulado:
        for linha in self.adj:
            linha.append(math.inf)
        self.adj.append([math.inf for i in range(self.n)])
    else:
        self.adj.append([0 for i in range(self.n)])

def insere_a(self, v, w, valor: float = 1):
    if self.adj[v][w] == math.inf:
        self.adj[v][w], self.adj[w][v] = valor, valor
        self.m += 1

def remove_a(self, v, w):
    if self.adj[v][w] != math.inf:
        self.adj[v][w], self.adj[w][v] = math.inf, math.inf
        self.m -= 1

def show(self):
    if self.rotulado:
        print(f"\n n: {self.n:2d} ", end="")
        print(f"m: {self.m:2d}\n")
        for i in range(self.n):
            for w in range(self.n):
                if self.adj[i][w] != math.inf:
                    print(f"Adj[{i:2d},{w:2d}] = ", self.adj[i][w], end=" ")
                else:
                    print(f"Adj[{i:2d},{w:2d}] = 0 ", end="")
            print("\n")
        print("\nfim da impressao do grafo.")
    else:
        print(f"\n n: {self.n:2d} ", end="")
        print(f"m: {self.m:2d}\n")
        for i in range(self.n):
            for w in range(self.n):
                if self.adj[i][w] == 1:
                    print(f"Adj[{i:2d},{w:2d}] = 1 ", end="")
                else:
                    print(f"Adj[{i:2d},{w:2d}] = 0 ", end="")
            print("\n")
        print("\nfim da impressao do grafo.")

def show_min(self):
    print(f"\n n: {self.n:2d} ", end="")
    print(f"m: {self.m:2d}\n")
    for i in range(self.n):
        for w in range(self.n):
            if self.rotulado:
                if self.adj[i][w] != math.inf:
                    print(" ", self.adj[i][w], end=" ")
                else:
                    print(" 0 ", end="")
            else:
                if self.adj[i][w] == 1:
                    print(" 1 ", end="")
                else:
                    print(" 0 ", end="")
        print("\n")
    print("\nfim da impressao do grafo.")

def in_degree(self, v: int) -> int:
    return len([linha for linha in self.adj if linha[v] != 0 and linha[v] !=
math.inf])

```

```

def out_degree(self, v: int) -> int:
    return len([sai for sai in self.adj[v] if sai != 0 and sai != math.inf])

def is_fonte(self, v: int) -> int:
    if self.in_degree(v) == 0 and self.out_degree(v) > 0:
        return 1
    return 0

def is_sorvedouro(self, v: int) -> int:
    if self.in_degree(v) > 0 and self.out_degree(v) == 0:
        return 1
    return 0

@staticmethod
def is_simetrico() -> int:
    return 1

def remover(self, v: int) -> int:
    if v < self.n:
        # Remove as arestas
        for _ in range(0, len(self.adj[v])):
            self.remove_a(v, _)
            self.remove_a(_, v)
        # Remove os vértices
        for linha in self.adj:
            del linha[v]
        del self.adj[v]
        self.n -= 1
        return 1
    else:
        return 0

def completo(self) -> int: # dei ctrl c ctrl v
    checa = 1
    for i in range(self.n):
        for w in range(self.n):
            if i != w:
                if self.adj[i][w] != 0 and self.adj[i][w] != math.inf:
                    continue
                else:
                    checa = 0
                    break
    if checa == 1:
        return 1
    else:
        return 0

@staticmethod
def marcar_no(marcados, no):
    marcados.append(no)
    return marcados

def no_adjacente(self, no, marcados):
    ads = self.adj[no]
    for _ in range(self.n):
        if (ads[_] != 0 and ads[_] != math.inf) and _ not in marcados:
            return _
    return -1

def nos_adjacentes(self, no, marcados):
    return [index for index, valor in enumerate(self.adj[no]) if (valor != 0 and valor != math.inf) and index not in marcados]

def is_adjac(self, i, j) -> int:
    if (self.adj[i][j] != 0 or self.adj[j][i] != 0) == True:

```

```

        return True

def is_adjacto(self, i):
    vertices = []
    for x in range(self.n):
        if self.adj[i][x] or self.adj[x][i] > 0:
            vertices.append(x)
    return(vertices)

def percurso_profundidade(self, v_inicio):
    marcados = []
    visita = []
    p = Pilha()
    visita.append(v_inicio)
    marcados = self.marcas_no(marcados, v_inicio)
    p.push(v_inicio)
    while not p.is_empty():
        no_atual = p.pop()
        no_seguinte = self.no_adjacente(no_atual, marcados)
        while no_seguinte != -1:
            visita.append(no_seguinte)
            p.push(no_atual)
            self.marcas_no(marcados, no_seguinte)
            no_atual = no_seguinte
            no_seguinte = self.no_adjacente(no_seguinte, marcados)
    return visita

def dijkstra(self, grafo, no_inicial):
    D = {n:float('inf') for n in range(grafo.n)}
    D[no_inicial] = 0
    pq = PriorityQueue()
    pq.put((0, no_inicial))
    while not pq.empty():
        (dist, no_atual) = pq.get()
        grafo.adj.append(no_atual)
        for vizinho in range(grafo.n):
            if grafo.adj[no_atual][vizinho] != -1:
                dist = grafo.adj[no_atual][vizinho]
                if vizinho not in grafo.visitados:
                    custo_antigo = D[vizinho]
                    novo_custo = D[no_atual] + dist
                    if novo_custo < custo_antigo:
                        pq.put((novo_custo, vizinho))
                        D[vizinho] = novo_custo

    print(D)

def conexidade(self, grafo, no_inicial):
    if len(self.percurso_profundidade(no_inicial)) == self.n:
        return "\nÉ CONEXO -,\` ★ \`,`-"
    else:
        return "\nÉ DESCONEXO -,\` ★ \`,`-"

def constelacao(self, num_const):
    match num_const:
        case 0:
            return "TERRA"
        case 1:
            return "ARIES"
        case 2:
            return "TOURO"
        case 3:
            return "GEMEOS"
        case 4:
            return "CANCER"
        case 5:

```

```

        return "LEAO"
    case 6:
        return "VIRGEM"
    case 7:
        return "LIBRA"
    case 8:
        return "ESCORPIAO"
    case 9:
        return "SAGITARIO"
    case 10:
        return "CAPRICORNIO"
    case 11:
        return "AQUARIO"
    case 12:
        return "PEIXES"

def percurso_feito(self, num_const):
    for i in num_const:
        msg = ""
        if i == 0:
            msg += "Partindo de "
        else:
            print(" ★ ° ☾ ☆ ,. , ★ :. . . , * :. \n")
            msg += ", * . . , . . , :. Agora em "
        msg += self.constelacao(i)
        if i == 0:
            msg += ", * :. . . , "
        print(msg)
    print("\n\nPercurso CONCLUÍDO _,' ★ \n")

```

util.py

Classes utilizadas de apoio.

```

class Pilha:
    def __init__(self):
        self.items = []
    def is_empty(self):
        return self.items == []
    def push(self, item):
        self.items.append(item)
    def pop(self):
        return self.items.pop()
    def peek(self):
        return self.items[len(self.items)-1]
    def size(self):
        return len(self.items)

class Fila:
    def __init__(self):
        self.items = []
    def is_empty(self):
        return self.items == []
    def enqueue(self, item):
        self.items.insert(0,item)
    def dequeue(self):
        return self.items.pop()

```

```
def size(self):  
    return len(self.items)
```