

Trabalho 4 - Grafos

- **Aluno:** Paulo Victor Fernandes de Melo
- **Aluno:** João Luiz Rodrigues da Silva
- **Aluna:** Rebecca Aimée Lima de Lima
- **Aluna:** Beatriz Jacaúna Martins

Questão 1 - Geração de grafos conexos

Para gerar um grafo conexo de tamanho `n`, primeiramente é gerado um vetor auxiliar chamado `conexoes` de tamanho `n` onde cada posição `conexoes[i]` tem valor `i + 1`.

```
// Vetor de conexões a serem feitas no grafo.
size_t *conexoes = malloc(tam * sizeof(size_t));

// Gerar vetor de conexões básico, sem repetições
for (size_t i = 0; i < tam; i++) {
    conexoes[i] = (i + 1) % tam;
}

// Embaralhar vetor
for (size_t i = 0; i < tam; i++) {
    size_t j = rand() % tam;
    size_t a = conexoes[i];
    conexoes[i] = conexoes[j];
    conexoes[j] = a;
}
```

Após isso, o vetor é embaralhado, e cada nó na posição `i` do vetor conecta-se com um nó aleatório anterior.

```
// Conectar todos os nós
for (size_t i = 1; i < tam; i++) {
    // Poderíamos sempre selecionar um nó anterior aleatório,
    // mas o grafo ficaria aleatório demais, e se sempre selecionarmos
    // o nó anterior para conexão, o grafo viraria uma linha.
    if (rand() % 3) {
        grafo_definir_aresta(grafo, conexoes[i], conexoes[rand() % i],
true);
    } else {
        grafo_definir_aresta(grafo, conexoes[i], conexoes[i - 1], true);
    }
}
```

Essa lógica gera um código garantidamente conexo, com ramificações. Uma lógica extra acima garante que não tenham ramificações excessivas.

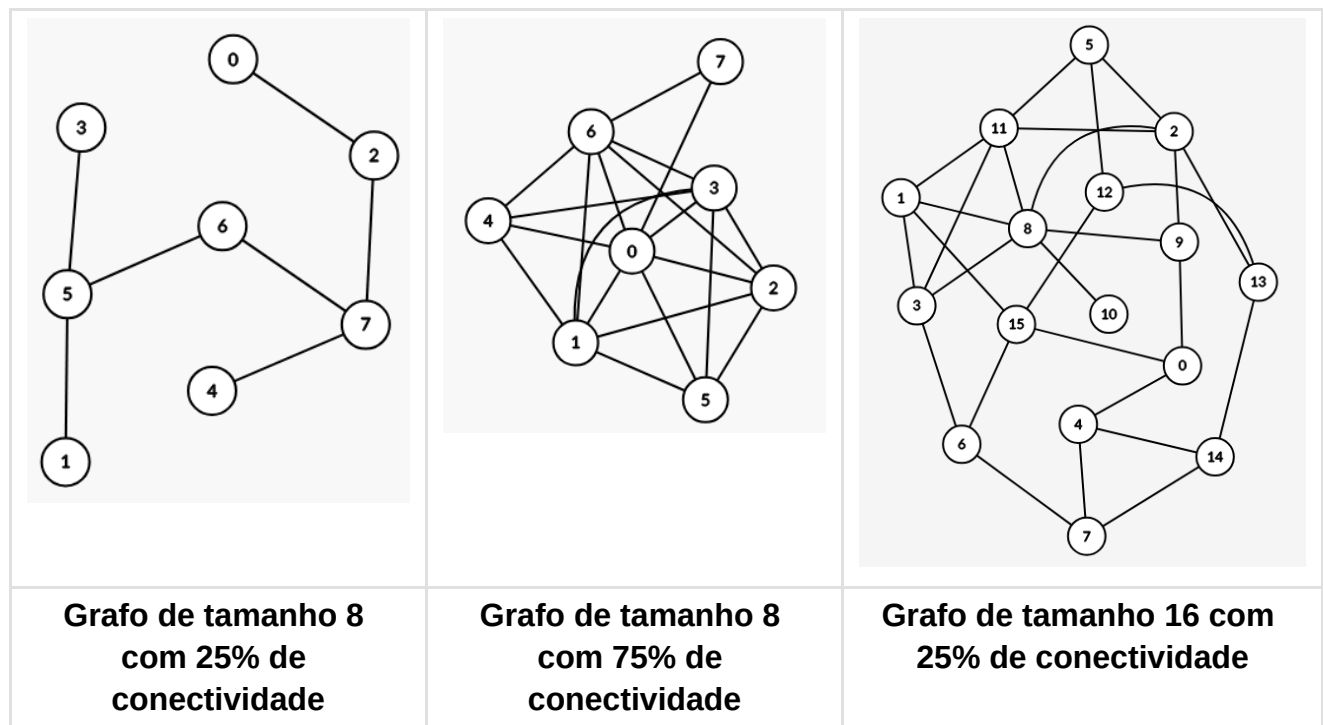
Após isso, para garantir o grau de conectividade, é calculado uma quantidade de arestas alvo a serem geradas, baseado no **numero máximo de arestas do grafo**. O número máximo equivale a quantidade de elementos abaixo da diagonal principal na matriz de adjacência do grafo.

```
// Máximo possível de arestas
size_t max_arestas = (tam - 1) * tam / 2;

// Quantidade de arestas que se deve gerar
size_t alvo = grau * max_arestas;

// Já temos um grafo com a quantidade adequada de arestas
if (alvo <= arestas) return grafo;
```

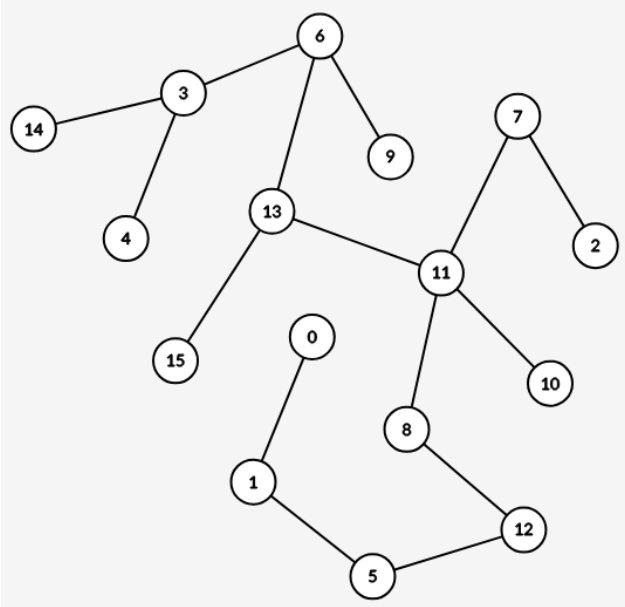
Após o calculo, uma quantidade de arestas indefinidas equivalente a `alvo - arestas` são aleatoriamente selecionados e preenchidos na matriz de adjacência, garantindo o grau de conectividade.



O código gera 12 grafos com diferentes tamanhos e graus de conectividade, porém, por brevidade, apenas 3 foram apresentados.

Questão 2 e 3 - Busca em largura e profundidade (BFS, DFS)

Para DFS e BFS, foram gerados 30 grafos com conectividades entre **10%, 25%, 50%, 75% e 100%**. Para os tamanhos, foram escolhidos **16, 64, 512, 1024, 2048, 4096 vértices**. Nos exemplos abaixo, apenas é mostrado um exemplo de tamanho pequeno com baixa conectividade, pois não foi possível visualizar grafos grandes demais.

	<pre> === QUESTÃO 2 e 3 - BFS e DFS === --- Grafo de tamanho 16 com 10.00% de conectividade --- > Origem escolhida para DFS/BFS: 3 === QUESTÃO 2 - Busca em largura (BFS) === Nível 0: 3 Nível 1: 4 6 14 Nível 2: 9 13 Nível 3: 11 15 Nível 4: 7 8 10 Nível 5: 2 12 Nível 6: 5 Nível 7: 1 Nível 8: 0 === QUESTÃO 3 - Busca em profundidade (DFS) === Ordem de visita a partir do vértice 3: 3, 4, 6, 9, 13, 11, 7, 2, 8, 12, 5, 1, 0, 10, 15, 14 </pre>
<p>Grafo de tamanho 16 com 10% de conectividade</p>	<p>Resultado da execução da BFS e DFS no código</p>
<pre> === QUESTÃO 2 e 3 - BFS e DFS === --- Grafo de tamanho 64 com 10.00% de conectividade --- > Origem escolhida para DFS/BFS: 56 === QUESTÃO 2 - Busca em largura (BFS) === Nível 0: 56 Nível 1: 7 10 19 23 25 27 44 48 Nível 2: 0 1 2 3 4 5 6 8 9 12 16 17 18 22 28 32 33 34 35 36 38 39 40 42 49 51 61 Nível 3: 11 13 14 15 20 21 24 26 29 30 31 37 41 43 45 46 47 50 52 53 54 55 58 59 60 62 63 Nível 4: 57 === QUESTÃO 3 - Busca em profundidade (DFS) === Ordem de visita a partir do vértice 56: 56, 7, 1, 3, 0, 6, 10, 28, 4, 9, 13, 2, 23, 34, 19, 51, 32, 26, 30, 54, 24, 8, 20, 11, 12, 22, 5, 15, 43, 38, 18, 16, 17, 39, 25, 61, 29, 42, 48, 60, 21, 46, 58, 31, 14, 52, 49, 53, 63, 35, 36, 47, 55, 44, 27, 33, 50, 40, 62, 59, 37, 41, 57, 4, 5 </pre>	<pre> === QUESTÃO 2 e 3 - BFS e DFS === --- Grafo de tamanho 64 com 25.00% de conectividade --- > Origem escolhida para DFS/BFS: 61 === QUESTÃO 2 - Busca em largura (BFS) === Nível 0: 61 Nível 1: 0 2 3 5 8 11 28 33 45 60 Nível 2: 1 4 6 7 9 10 12 13 15 16 17 18 19 20 21 22 23 24 25 26 27 29 30 31 32 34 35 36 37 38 39 40 41 42 43 44 46 47 48 49 50 51 52 53 55 56 57 58 59 62 63 Nível 3: 14 54 === QUESTÃO 3 - Busca em profundidade (DFS) === Ordem de visita a partir do vértice 61: 61, 0, 4, 1, 2, 5, 3, 10, 7, 6, 11, 15, 21, 9, 14, 13, 8, 19, 17, 23, 18, 12, 16, 27, 25, 33, 20, 24, 31, 38, 43, 34, 35, 22, 26, 37, 39, 29, 42, 49, 32, 44, 41, 30, 28, 46, 36, 54, 51, 47, 45, 63, 50, 57, 48, 60, 40, 52, 59, 62, 53, 55, 58, 5, 6 Pressione qualquer caractere para continuar a execução... </pre>
<p>Outras execuções (grafos grandes demais)</p>	

Notavelmente, maiores graus de conectividade diminuem os níveis da árvore, pois há mais caminhos para onde a busca pode caminhar.

Abaixo estão os tempos de execução da BFS, em milissegundos;

Conectividade	Tamanho 16	Tamanho 64	Tamanho 512	Tamanho 1024	Tamanho 2048	Tamanho 4096
10.00%	0.0070	0.0820	4.8070	14.6270	57.8810	236.5880
25.00%	0.0070	0.0900	6.6100	18.0210	68.1780	268.6590

Conectividade	Tamanho 16	Tamanho 64	Tamanho 512	Tamanho 1024	Tamanho 2048	Tamanho 4096
50.00%	0.0080	0.1240	5.3770	19.3630	77.0090	300.8650
75.00%	0.0120	0.0720	4.6490	16.4290	66.2370	260.4510
100.00%	0.0080	0.0570	3.5140	13.5700	54.2950	216.8700
Medias	0.0070	0.0708	4.1595	13.6683	53.9333	213.9055

E abaixo, os tempos de execução da DFS, em milissegundos.

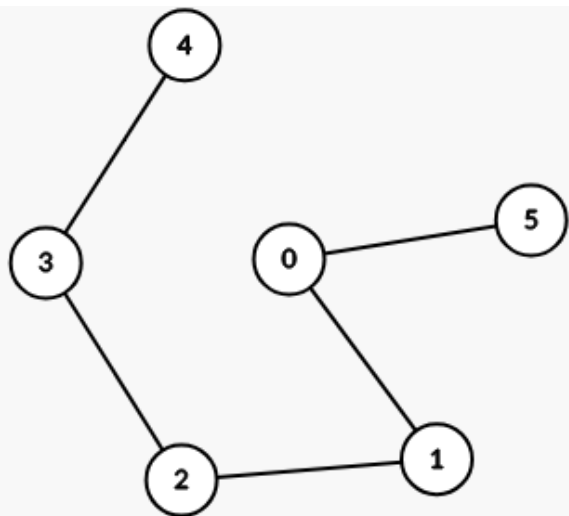
Conectividade	Tamanho 16	Tamanho 64	Tamanho 512	Tamanho 1024	Tamanho 2048	Tamanho 4096
10.00%	0.0050	0.0390	1.3090	3.4590	13.1980	52.8090
25.00%	0.0040	0.0460	1.3890	3.2740	12.7660	51.4170
50.00%	0.0030	0.0300	0.8320	3.2210	12.9370	50.6080
75.00%	0.0050	0.0180	0.9580	3.2210	13.0280	50.1820
100.00%	0.0030	0.0160	0.8360	3.1370	12.3930	50.3480
Medias	0.0033	0.0248	0.8873	2.7187	10.7203	42.5607

Percebe-se a grande diferença no tempo de execução da DFS com a BFS. Isto provavelmente se deve a necessidade de usar uma fila para organizar as execuções na BFS, enquanto na DFS é possível utilizar apenas a pilha de execução.

Questão 4 - Geração de todos os caminhos com DFS

Para gerar todos os caminhos possíveis no grafo, o algoritmo DFS foi modificado. A modificação feita **cria uma "cópia" do contexto atual** da DFS, com o vetor de elementos visitados, para cada vizinho não visitado de cada nó. Os contextos copiados são coletados no final em uma matriz, que então é impressa na tela. A origem da busca é determinada **aleatoriamente**, em cada execução. Todos os grafos possuem tamanho 6, pois, para altas conectividades, tamanhos maiores possuíam caminhos crescendo em ordem fatorial.

Grafo de tamanho 6 gerado



Resultado das execuções

=== QUESTÃO 4 - Geração de caminhos usando DFS ===

>- Mini-grafo de tamanho 6 com 25.00% de conectividade -<
> Origem escolhida para DFS: 3

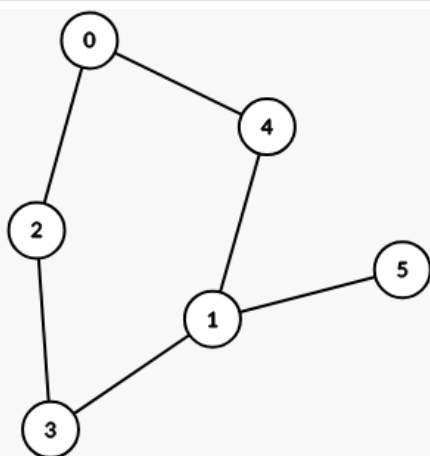
> Pares de arestas do grafo:

0 1
0 5
1 2
2 3
3 4

> Caminhos gerados:

3 -> 4
3 -> 2 -> 1 -> 0 -> 5

Pressione qualquer caractere para continuar a execução...



=== QUESTÃO 4 - Geração de caminhos usando DFS ===

>- Mini-grafo de tamanho 6 com 50.00% de conectividade -<
> Origem escolhida para DFS: 3

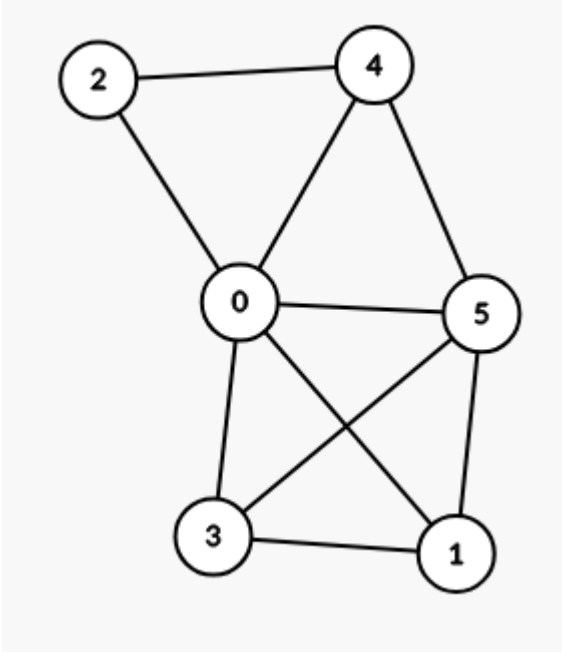
> Pares de arestas do grafo:

0 2
0 4
1 3
1 4
1 5
2 3

> Caminhos gerados:

3 -> 2 -> 0 -> 4 -> 1 -> 5
3 -> 1 -> 5
3 -> 1 -> 4 -> 0 -> 2

Pressione qualquer caractere para continuar a execução...

Grafo de tamanho 6 gerado	Resultado das execuções
	<pre> === QUESTÃO 4 - Geração de caminhos usando DFS === >- Mini-grafo de tamanho 6 com 75.00% de conectividade -< > Origem escolhida para DFS: 1 > Pares de arestas do grafo: 0 1 0 2 0 3 0 4 0 5 1 3 1 5 2 4 3 5 4 5 > Caminhos gerados: 1 -> 5 -> 4 -> 2 -> 0 -> 3 1 -> 5 -> 4 -> 0 -> 3 1 -> 5 -> 4 -> 0 -> 2 1 -> 5 -> 3 -> 0 -> 4 -> 2 1 -> 5 -> 3 -> 0 -> 2 -> 4 1 -> 5 -> 0 -> 4 -> 2 1 -> 5 -> 0 -> 3 1 -> 5 -> 0 -> 2 -> 4 1 -> 3 -> 5 -> 4 -> 2 -> 0 1 -> 3 -> 5 -> 4 -> 0 -> 2 1 -> 3 -> 5 -> 0 -> 4 -> 2 1 -> 3 -> 5 -> 0 -> 2 -> 4 1 -> 3 -> 0 -> 5 -> 4 -> 2 1 -> 3 -> 0 -> 4 -> 5 1 -> 3 -> 0 -> 4 -> 2 1 -> 3 -> 0 -> 2 -> 4 -> 5 1 -> 0 -> 5 -> 4 -> 2 1 -> 0 -> 5 -> 3 1 -> 0 -> 4 -> 5 -> 3 1 -> 0 -> 4 -> 2 1 -> 0 -> 3 -> 5 -> 4 -> 2 1 -> 0 -> 2 -> 4 -> 5 -> 3 Pressione qualquer caractere para continuar a execução... </pre>

Perceba como a adição de poucas arestas no código implica num crescimento enorme na quantidade de caminhos gerados. Por isto, grafos mais complexos não foram calculados, nem apresentados.

Questão 5 - Detecção de ciclos usando DFS

Para detectar ciclos nos grafos, o algoritmo DFS também foi modificado. Para cada vértice no grafo, uma DFS é executada, e se qualquer ciclo maior que 3 for detectado, a função inteira retorna `true`. O algoritmo foi escrito de forma genérica, porém, pelo uso de grafos conexos, é necessária a execução de apenas **uma DFS** para detectar os ciclos.

A lógica se resume a caminhar o grafo usando DFS e, caso um vizinho visitado seja encontrado, **que não seja o nó visitado anteriormente ao atual** (no código, denominado `pai`), o algoritmo conta a presença da aresta com o vizinho como um **ciclo**.

```

for (size_t vizinho = 0; vizinho < grafo->tam; vizinho++) {
    if (grafo_tem_aresta(grafo, atual, vizinho)) {
        if (!visitado[vizinho]) {
            if (dfs_ciclo(grafo, vizinho, atual, visitado)) {

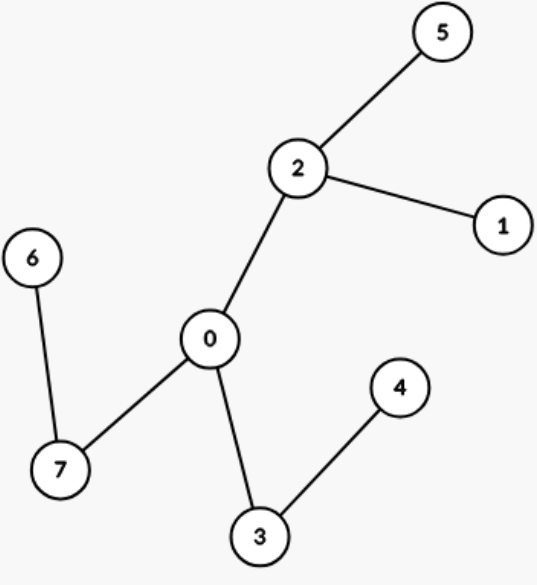
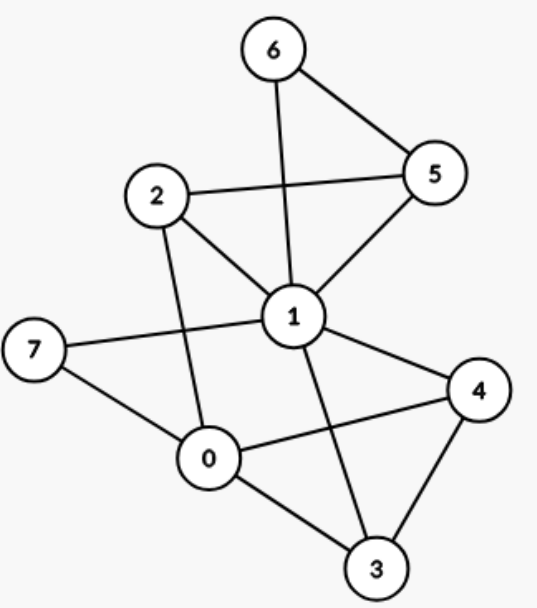
```

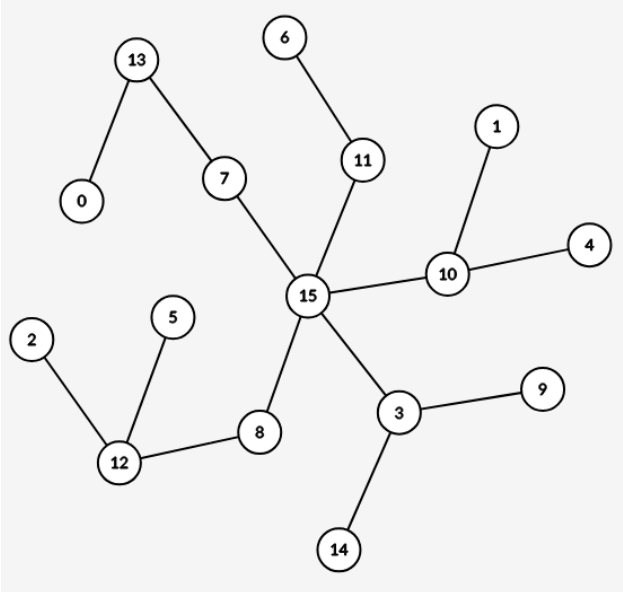
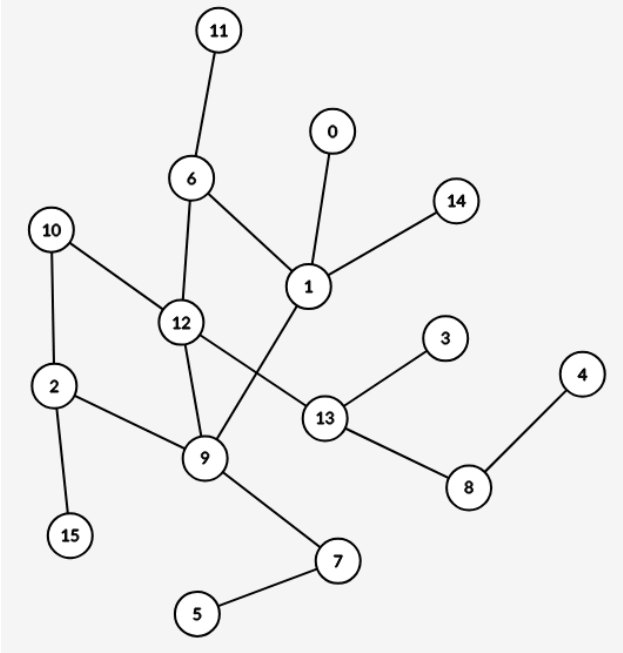
```

        return true;
    }
    } else if (vizinho != pai) {
        return true; // Encontrou um ciclo (tamanho ≥ 3)
    }
}
}

```

Abaixo, estão algumas execuções do algoritmo.

Grafo associado	Resultado da execução	Ciclo?
	<pre> === QUESTÃO 5 - BFS e DFS === --- Grafo de tamanho 8 com 5.00% de conectividade --- > Matriz de adjacência: 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 > Pares de arestas: 0 2 0 3 0 7 1 2 2 5 3 4 6 7 > O grafo NÃO possui ciclo! Pressione qualquer caractere para continuar a execução... </pre>	Não
	<pre> === QUESTÃO 5 - BFS e DFS === --- Grafo de tamanho 8 com 50.00% de conectividade --- > Matriz de adjacência: 0 0 1 1 1 0 0 1 0 0 1 1 1 1 1 1 1 1 0 0 0 1 0 0 1 1 0 0 1 0 0 0 1 1 0 1 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 1 0 0 1 1 0 0 0 0 0 0 > Pares de arestas: 0 2 0 3 0 4 0 7 1 2 1 3 1 4 1 5 1 6 1 7 2 5 3 4 5 6 > O grafo possui ciclo! Pressione qualquer caractere para continuar a execução... </pre>	Sim

Grafo associado	Resultado da execução	Ciclo?
	<pre>--- Grafo de tamanho 16 com 5.00% de conectividade --- > Matriz de adjacência: 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 1 0 1 1 0 0 0 0 > Pares de arestas: 0 13 1 10 2 12 3 9 3 14 3 15 4 10 5 12 6 11 7 13 7 15 8 12 8 15 10 15 11 15 > O grafo NÃO possui ciclo! Pressione qualquer caractere para continuar a execução...</pre>	Não
	<pre>=== QUESTÃO 5 - BFS e DFS === --- Grafo de tamanho 16 com 15.00% de conectividade --- > Matriz de adjacência: 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 > Pares de arestas: 0 1 1 6 1 9 1 14 2 9 2 10 2 15 3 13 4 8 5 7 6 11 6 12 7 9 8 13 9 12 10 12 12 13 > O grafo possui ciclo! Pressione qualquer caractere para continuar a execução... █</pre>	Sim