

Questão 1

Correto

Atingiu 0,9 de 1,0

Instruções:

Utilize a classe base para grafos e insira um método para imprimir na tela o resultado da Busca em Largura em um grafo, a partir de um vértice v , fornecido como parâmetro.

O protótipo da função é BFS(s), onde

- s é o vértice de origem da BFS

#Teste 1

```
g = Graph(4)
g.addEdge(0, 1)
g.addEdge(0, 2)
g.addEdge(1, 2)
g.addEdge(2, 0)
g.addEdge(2, 3)
g.addEdge(3, 3)
g.BFS(2)
```

Saída:

2 0 3 1

#Teste 2

```
g = Graph(13)
g.addEdge(0,1)
g.addEdge(1, 2)
g.addEdge(1, 3)
g.addEdge(1, 4)
g.addEdge(2, 5)
g.addEdge(2, 6)
g.addEdge(4, 7)
g.addEdge(4, 8)
g.addEdge(5, 9)
g.addEdge(5, 10)
g.addEdge(7, 11)
g.addEdge(7, 12)
g.BFS(0)
```

Saída:

0 1 2 3 4 5 6 7 8 9 10 11 12

#Teste 3

```
g = Graph(13)
g.addEdge(0,1)
g.addEdge(1, 2)
g.addEdge(1, 3)
g.addEdge(1, 4)
g.addEdge(2, 5)
g.addEdge(2, 6)
g.addEdge(4, 7)
g.addEdge(4, 8)
g.addEdge(5, 9)
g.addEdge(5, 10)
```

```
g.addEdge(7, 11)
```

```
g.addEdge(7, 12)
```

```
g.BFS(5)
```

Saída:

5 9 10

#Teste 4

```
g = Graph(9)
```

```
g.addEdge(0, 1)
```

```
g.addEdge(0, 3)
```

```
g.addEdge(0, 4)
```

```
g.addEdge(1, 2)
```

```
g.addEdge(1, 4)
```

```
g.addEdge(2, 5)
```

```
g.addEdge(3, 4)
```

```
g.addEdge(3, 6)
```

```
g.addEdge(4, 5)
```

```
g.addEdge(4, 7)
```

```
g.addEdge(6, 4)
```

```
g.addEdge(6, 7)
```

```
g.addEdge(7, 5)
```

```
g.addEdge(7, 8)
```

```
g.BFS(0)
```

Saída:

0 1 3 4 2 6 5 7 8

#Teste 5

```
g = Graph(9)
```

```
g.addEdge(0, 1)
```

```
g.addEdge(0, 3)
```

```
g.addEdge(0, 4)
```

```
g.addEdge(1, 2)
```

```
g.addEdge(1, 4)
```

```
g.addEdge(2, 5)
```

```
g.addEdge(3, 4)
```

```
g.addEdge(3, 6)
```

```
g.addEdge(4, 5)
```

```
g.addEdge(4, 7)
```

```
g.addEdge(6, 4)
```

```
g.addEdge(6, 7)
```

```
g.addEdge(7, 5)
```

```
g.addEdge(7, 8)
```

```
g.BFS(2)
```

Saída:

2 5

Answer: (penalty regime: 10, 20, ... %)

```
1  
2 from collections import defaultdict  
3  
4 # Classe base para grafos  
5 class Graph:  
6     # Construtor
```

```

7  def __init__(self, vertices):
8      self.V = vertices # Número de vértices
9      self.graph = defaultdict(list) # Dicionário de listas de adjacências
10
11 # Função para adicionar uma aresta ao grafo
12 def addEdge(self, u, v):
13     self.graph[u].append(v)
14
15 # Função para imprimir o grafo
16 def printGraph(self):
17     for i in range(self.V):
18         print(i, ":", self.graph[i])
19
20 # Função para imprimir o resultado da BFS
21 def BFS(self, s):
22     # Inicializa a fila
23     queue = []
24     # Marca todos os vértices como não visitados
25     visited = [False] * (self.V)
26     # Marca o vértice de origem como visitado e o insere na fila
27     queue.append(s)
28     visited[s] = True
29     # Enquanto a fila não estiver vazia
30     while queue:
31         # Remove um vértice da fila
32         s = queue.pop(0)
33         # Imprime o vértice
34         print(s, end=" ")
35         # Para cada vértice adjacente a s
36         for i in self.graph[s]:
37             # Se o vértice não foi visitado
38             if visited[i] == False:
39                 # Marca o vértice como visitado e o insere na fila
40                 queue.append(i)
41                 visited[i] = True
42
43

```

| | Test | Expected | Got | |
|---|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------|------------------------------|---|
| ✓ | g = Graph(4) g.addEdge(0, 1) g.addEdge(0, 2) g.addEdge(1, 2) g.addEdge(2, 0) g.addEdge(2, 3) g.addEdge(3, 3) g.BFS(2) | 2 0 3 1 | 2 0 3 1 | ✓ |
| ✓ | g = Graph(13) g.addEdge(0,1) g.addEdge(1, 2) g.addEdge(1, 3) g.addEdge(1, 4) g.addEdge(2, 5) g.addEdge(2, 6) g.addEdge(4, 7) g.addEdge(4, 8) g.addEdge(5, 9) g.addEdge(5, 10) g.addEdge(7, 11) g.addEdge(7, 12) g.BFS(0) | 0 1 2 3 4 5 6 7 8 9 10 11 12 | 0 1 2 3 4 5 6 7 8 9 10 11 12 | ✓ |

| | Test | Expected | Got | |
|---|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|-------------------|---|
| ✓ | <pre> g = Graph(13) g.addEdge(0, 1) g.addEdge(1, 2) g.addEdge(1, 3) g.addEdge(1, 4) g.addEdge(2, 5) g.addEdge(2, 6) g.addEdge(4, 7) g.addEdge(4, 8) g.addEdge(5, 9) g.addEdge(5, 10) g.addEdge(7, 11) g.addEdge(7, 12) g.BFS(5) </pre> | 5 9 10 | 5 9 10 | ✓ |
| ✓ | <pre> g = Graph(9) g.addEdge(0, 1) g.addEdge(0, 3) g.addEdge(0, 4) g.addEdge(1, 2) g.addEdge(1, 4) g.addEdge(2, 5) g.addEdge(3, 4) g.addEdge(3, 6) g.addEdge(4, 5) g.addEdge(4, 7) g.addEdge(6, 4) g.addEdge(6, 7) g.addEdge(7, 5) g.addEdge(7, 8) g.BFS(0) </pre> | 0 1 3 4 2 6 5 7 8 | 0 1 3 4 2 6 5 7 8 | ✓ |
| ✓ | <pre> g = Graph(9) g.addEdge(0, 1) g.addEdge(0, 3) g.addEdge(0, 4) g.addEdge(1, 2) g.addEdge(1, 4) g.addEdge(2, 5) g.addEdge(3, 4) g.addEdge(3, 6) g.addEdge(4, 5) g.addEdge(4, 7) g.addEdge(6, 4) g.addEdge(6, 7) g.addEdge(7, 5) g.addEdge(7, 8) g.BFS(2) </pre> | 2 5 | 2 5 | ✓ |

Passou em todos os teste! ✓

Correto

Notas para o envio: 1,0/1,0. De acordo com as tentativas anteriores **0,9/1,0**.

Questão **2**

Correto

Atingiu 0,7 de 1,0

Instruções:

Utilizando a classe criada anteriormente, escreva uma função para imprimir na tela o resultado da [Busca em Profundidade](#) em um grafo, a partir de um vértice inicial v , fornecido como parâmetro

O protótipo da função é $\text{DFS}(s)$, onde

s é o vértice de origem da DFS

OBS:

(1) é possível e indicado que vc crie funções auxiliares para sua resposta.

(2) selecione os vértices em ordem crescente de valor

#Teste 1

 $g = \text{Graph}(4)$ $g.\text{addEdge}(0, 1)$ $g.\text{addEdge}(0, 2)$ $g.\text{addEdge}(1, 2)$ $g.\text{addEdge}(2, 0)$ $g.\text{addEdge}(2, 3)$ $g.\text{addEdge}(3, 3)$ $g.\text{DFS}(2)$

Saída:

2 0 1 3

#Teste 2

 $g = \text{Graph}(13)$ $g.\text{addEdge}(0,1)$ $g.\text{addEdge}(1, 2)$ $g.\text{addEdge}(1, 3)$ $g.\text{addEdge}(1, 4)$ $g.\text{addEdge}(2, 5)$ $g.\text{addEdge}(2, 6)$ $g.\text{addEdge}(4, 7)$ $g.\text{addEdge}(4, 8)$ $g.\text{addEdge}(5, 9)$ $g.\text{addEdge}(5, 10)$ $g.\text{addEdge}(7, 11)$ $g.\text{addEdge}(7, 12)$ $g.\text{DFS}(0)$

Saída:

0 1 2 5 9 10 6 3 4 7 11 12 8

#Teste 3

 $g = \text{Graph}(13)$ $g.\text{addEdge}(0,1)$ $g.\text{addEdge}(1, 2)$ $g.\text{addEdge}(1, 3)$ $g.\text{addEdge}(1, 4)$ $g.\text{addEdge}(2, 5)$ $g.\text{addEdge}(2, 6)$

```
g.addEdge(4, 7)
g.addEdge(4, 8)
g.addEdge(5, 9)
g.addEdge(5, 10)
g.addEdge(7, 11)
g.addEdge(7, 12)
g.DFS(2)
```

Saída:

2 5 9 10 6

#Teste 4

```
g = Graph(9)
g.addEdge(0, 1)
g.addEdge(0, 3)
g.addEdge(0, 4)
g.addEdge(1, 2)
g.addEdge(1, 4)
g.addEdge(2, 5)
g.addEdge(3, 4)
g.addEdge(3, 6)
g.addEdge(4, 5)
g.addEdge(4, 7)
g.addEdge(6, 4)
g.addEdge(6, 7)
g.addEdge(7, 5)
g.addEdge(7, 8)
g.DFS(0)
```

Saída:

0 1 2 5 4 7 8 3 6

#Teste 5

```
g = Graph(9)
g.addEdge(0, 1)
g.addEdge(0, 3)
g.addEdge(0, 4)
g.addEdge(1, 2)
g.addEdge(1, 4)
g.addEdge(2, 5)
g.addEdge(3, 4)
g.addEdge(3, 6)
g.addEdge(4, 5)
g.addEdge(4, 7)
g.addEdge(6, 4)
g.addEdge(6, 7)
g.addEdge(7, 5)
g.addEdge(7, 8)
g.DFS( 2)
```

Saída:

2 5

Answer: (penalty regime: 10, 20, ... %)

```

1 from collections import defaultdict
2
3 class Vertex:
4     def __init__(self):
5         self.cor = "BRANCO"
6         self.tempo_descoberta = float("inf")
7         self.tempo_finalizacao = float("inf")
8         self.to = []
9
10
11 class Graph:
12     def __init__(self, n):
13         self.n = n
14         self.adj = [Vertex() for i in range(n)]
15         self.time = 0
16
17     def addEdge(self, u, v):
18         self.adj[u].to.append(v)
19
20     def printGraph(self):
21         for i in range(self.n):
22             print(i, end=" ")
23             for j in self.adj[i].to:
24                 print("->", j, end=" ")
25             print()
26
27     def DFS(self, s):
28         self.time = 0
29         for i in range(self.n):
30             self.adj[i].cor = "BRANCO"
31             self.adj[i].tempo_descoberta = float("inf")
32             self.adj[i].tempo_finalizacao = float("inf")
33         self.DFS_VISIT(s)
34
35     def DFS_VISIT(self, u):
36         self.time += 1
37         self.adj[u].tempo_descoberta = self.time
38         self.adj[u].cor = "CINZA"
39         print(u, end=" ")
40         for v in self.adj[u].to:
41             if self.adj[v].cor == "BRANCO":
42                 self.DFS_VISIT(v)
43         self.adj[u].cor = "PRETO"
44         self.time += 1
45         self.adj[u].tempo_finalizacao = self.time

```

| | Test | Expected | Got | |
|---|--------------------------------------------------------------------------------------------------------------------------------------------|----------|---------|---|
| ✓ | g = Graph(4) g.addEdge(0, 1) g.addEdge(0, 2) g.addEdge(1, 2) g.addEdge(2, 0) g.addEdge(2, 3) g.addEdge(3, 3) g.DFS(2) | 2 0 1 3 | 2 0 1 3 | ✓ |

| | Test | Expected | Got | |
|---|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------|------------------------------|---|
| ✓ | <pre> g = Graph(13) g.addEdge(0,1) g.addEdge(1, 2) g.addEdge(1, 3) g.addEdge(1, 4) g.addEdge(2, 5) g.addEdge(2, 6) g.addEdge(4, 7) g.addEdge(4, 8) g.addEdge(5, 9) g.addEdge(5, 10) g.addEdge(7, 11) g.addEdge(7, 12) g.DFS(0) </pre> | 0 1 2 5 9 10 6 3 4 7 11 12 8 | 0 1 2 5 9 10 6 3 4 7 11 12 8 | ✓ |
| ✓ | <pre> g = Graph(13) g.addEdge(0,1) g.addEdge(1, 2) g.addEdge(1, 3) g.addEdge(1, 4) g.addEdge(2, 5) g.addEdge(2, 6) g.addEdge(4, 7) g.addEdge(4, 8) g.addEdge(5, 9) g.addEdge(5, 10) g.addEdge(7, 11) g.addEdge(7, 12) g.DFS(2) </pre> | 2 5 9 10 6 | 2 5 9 10 6 | ✓ |
| ✓ | <pre> g = Graph(9) g.addEdge(0, 1) g.addEdge(0, 3) g.addEdge(0, 4) g.addEdge(1, 2) g.addEdge(1, 4) g.addEdge(2, 5) g.addEdge(3, 4) g.addEdge(3, 6) g.addEdge(4, 5) g.addEdge(4, 7) g.addEdge(6, 4) g.addEdge(6, 7) g.addEdge(7, 5) g.addEdge(7, 8) g.DFS(0) </pre> | 0 1 2 5 4 7 8 3 6 | 0 1 2 5 4 7 8 3 6 | ✓ |
| ✓ | <pre> g = Graph(9) g.addEdge(0, 1) g.addEdge(0, 3) g.addEdge(0, 4) g.addEdge(1, 2) g.addEdge(1, 4) g.addEdge(2, 5) g.addEdge(3, 4) g.addEdge(3, 6) g.addEdge(4, 5) g.addEdge(4, 7) g.addEdge(6, 4) g.addEdge(6, 7) g.addEdge(7, 5) g.addEdge(7, 8) g.DFS(2) </pre> | 2 5 | 2 5 | ✓ |

Passou em todos os teste! ✓

Correto

Notas para o envio: 1,0/1,0. De acordo com as tentativas anteriores **0,7/1,0**.

[◀ Video - Busca em Profundidade](#)

Seu nome