

PIBIC – UFAPE

Análise da corretude de algoritmos gerados por Inteligência Artificial

Comparação deles com algoritmos de repositórios existentes

Índice.

3. OBJETIVOS

4. ALGORITMO DE DIJKSTRA

5. ANÁLISE DA CORRETEDE

5.1. CRIAÇÃO DO
REPOSITÓRIO

5.2. GRUPO DE CONTROLE

5.3 TECNOLOGIAS USADAS

5.4 TESTE

6. RESULTADO

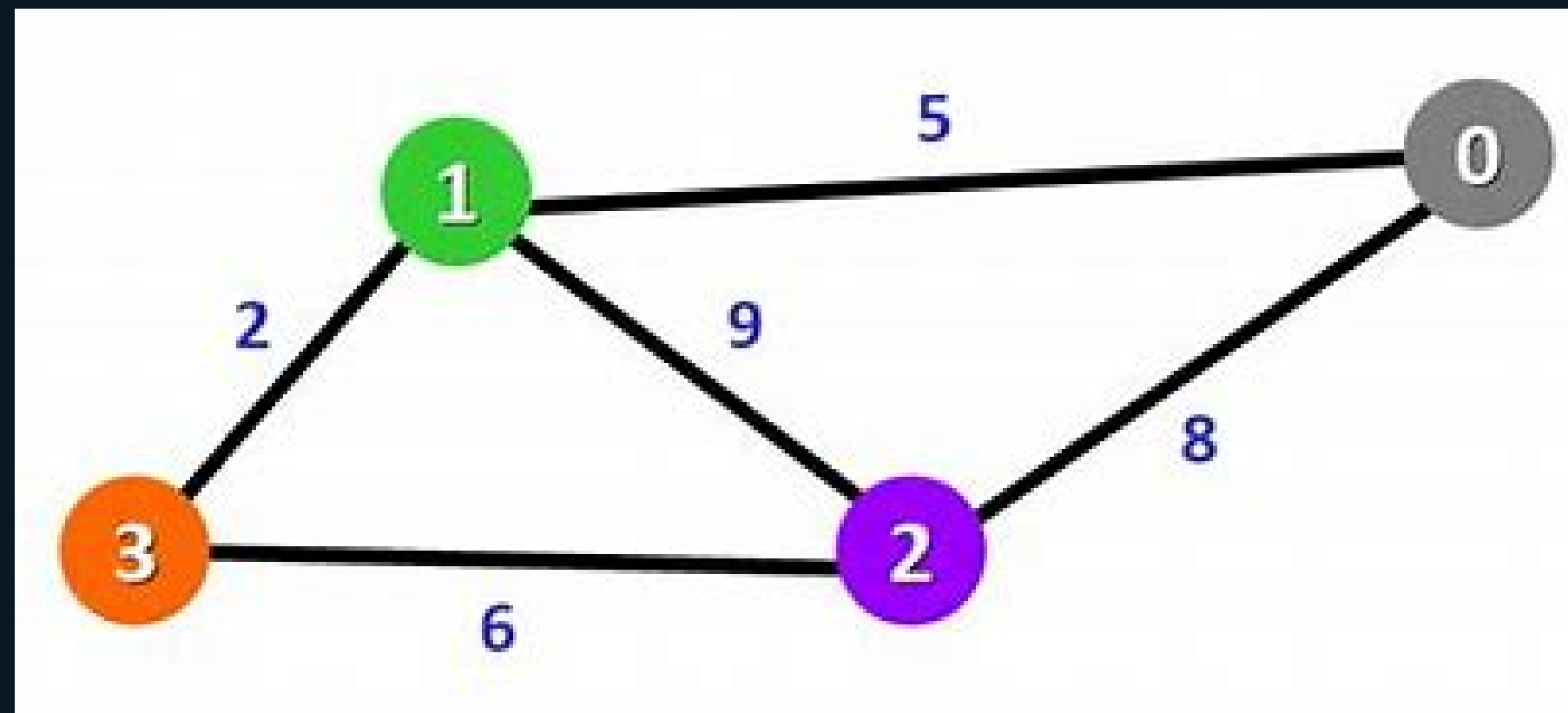
Objetivos.

Gerar o algoritmo de Dijkstra usando Inteligências Artificiais

Comparar a execução deles com dois algoritmos de Dijkstra já existentes em repositórios

Algoritmo de Dijkstra.

O algoritmo de Dijkstra é uma técnica utilizada para encontrar o caminho mais curto entre dois pontos em um grafo, podendo ser tanto em grafos com pesos (onde as arestas têm valores numéricos associados) quanto em grafos sem pesos. Ele explora os caminhos possíveis, sempre escolhendo o caminho de menor custo até alcançar o destino.



Análise da corretude.

- Criação de um repositório para guardar todos os códigos utilizados;
- Seleção de repositórios para o grupo de controle;
- Escolha das Inteligências Artificiais e das tecnologias necessárias para o desenvolvimento da análise;
- Testar os algoritmos gerados e os do repositório.

Análise da corretude.

- Consideramos a seguinte classificação quanto à corretude:

Correct: códigos sintática e semanticamente equivalentes ao resultado dos repositórios;

Plausible: Códigos sintaticamente corretos, mas com poucas falhas semânticas;

Invalid: Códigos sintaticamente corretos, mas semanticamente incorretos;

Incorrect: Códigos que não compilaram - sintaticamente incorretos.

Criação do repositório.

PIBIC / [com](#) / [dijkstra](#) /

Add file ▾

Jeffersonalanmm updates

16ba1db · now History

| Name | Last commit message | Last commit date |
|--------------------|---|------------------|
| .. | | |
| Main.java | updates | now |
| amazonQ.java | copilot code implemented and prints removed | last week |
| blackBox.java | updates | now |
| chatGPT.java | updates | now |
| codeium.java | updates | now |
| copilot.java | copilot code implemented and prints removed | last week |
| googleBard.java | updates | now |
| rosetta.java | updates | now |
| theAlgorithms.java | updates | now |

Grupo de Controle.

[LinkedIn](#)
[The Algorithms](#)
[chat](#)
357 online

 receives 0.40 USD/week

[Stars](#) 365k

[Follow us](#)

We are a group of programmers helping each other build new things, whether it be writing complex encryption programs, or simple ciphers. Our goal is to work together to document and model beautiful, helpful and interesting algorithms using code. We are an open-source community - anyone can contribute. We check each other's work, communicate and collaborate to solve problems. We strive to be welcoming, respectful, yet make sure that our code follows the latest programming guidelines.

[View all](#)

● Python ● C++ ● TypeScript ● HTML
● Zig

[algorithms](#)
[hacktoberfest](#)
[data-structures](#)
[algorithm](#)
[search](#)

Developer Program Member

[Report abuse](#)

Rosetta Code

[Main Page](#) [Discussion](#)

Read View source View history Tools ▾

Rosetta Code is a [programming chrestomathy](#) site. The idea is to present solutions to the same task in as many different languages as possible, to demonstrate how languages are similar and different, and to aid a person with a grounding in one approach to a problem in learning another. Rosetta Code currently has 1,264 [tasks](#), 404 [draft tasks](#), and is aware of 931 [languages](#), though we do not (and cannot) have solutions to every task in every language.

Places to start

Recently-Updated Tasks

| |
|---|
| |
| Base64 decode data |
| Zsigmondy numbers |
| Append a record to the end of a text file |
| Quine |
| Anadromes |
| Barnsley fern |
| Factorial of a number |

Tecnologias usadas.



Tecnologias usadas.

Please write your best implementation of the Dijkstra's algorithm in the Java programming language.

Teste.

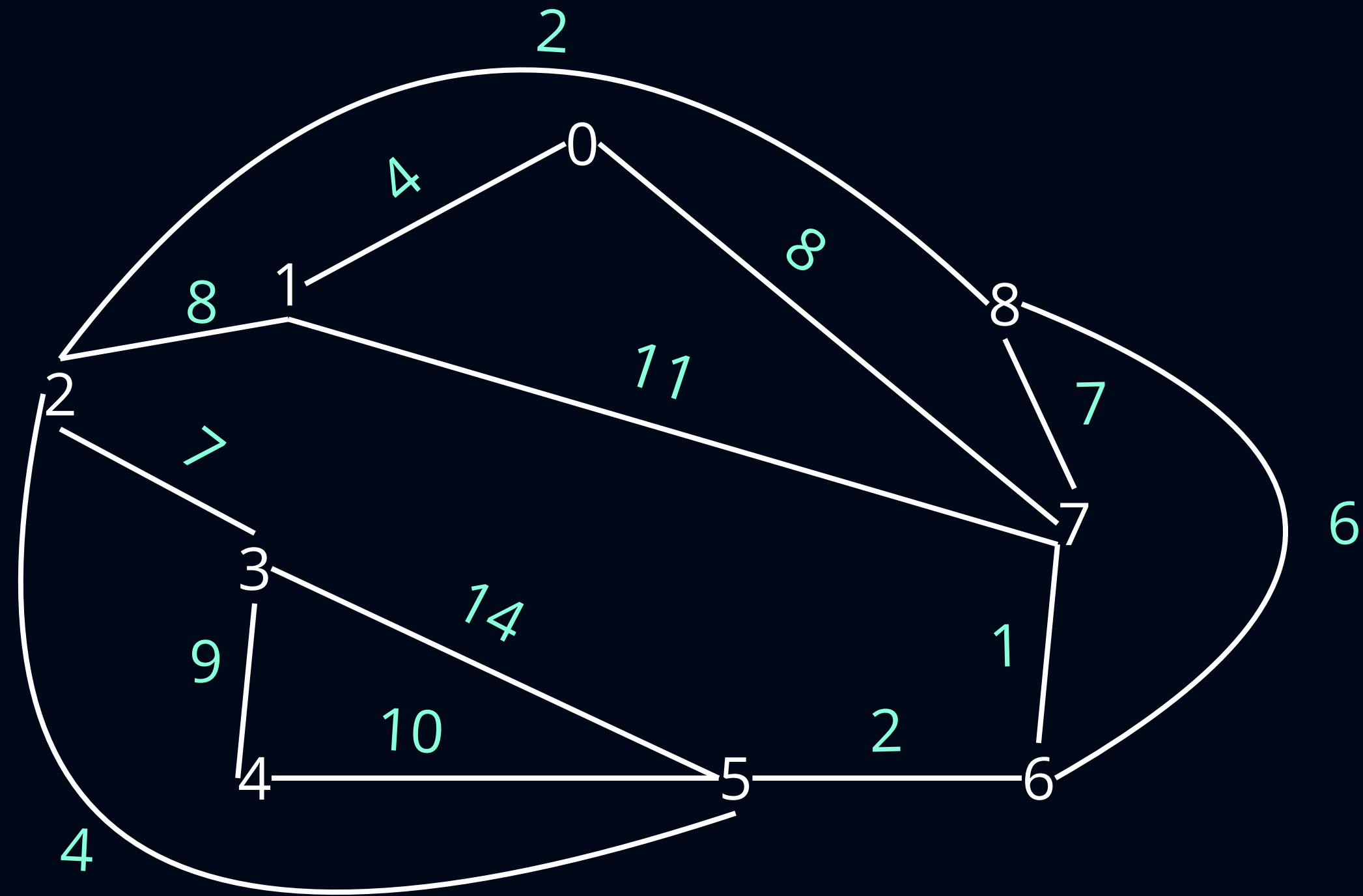
J Main.java 1, U X

com > dijkstra > J Main.java > ...

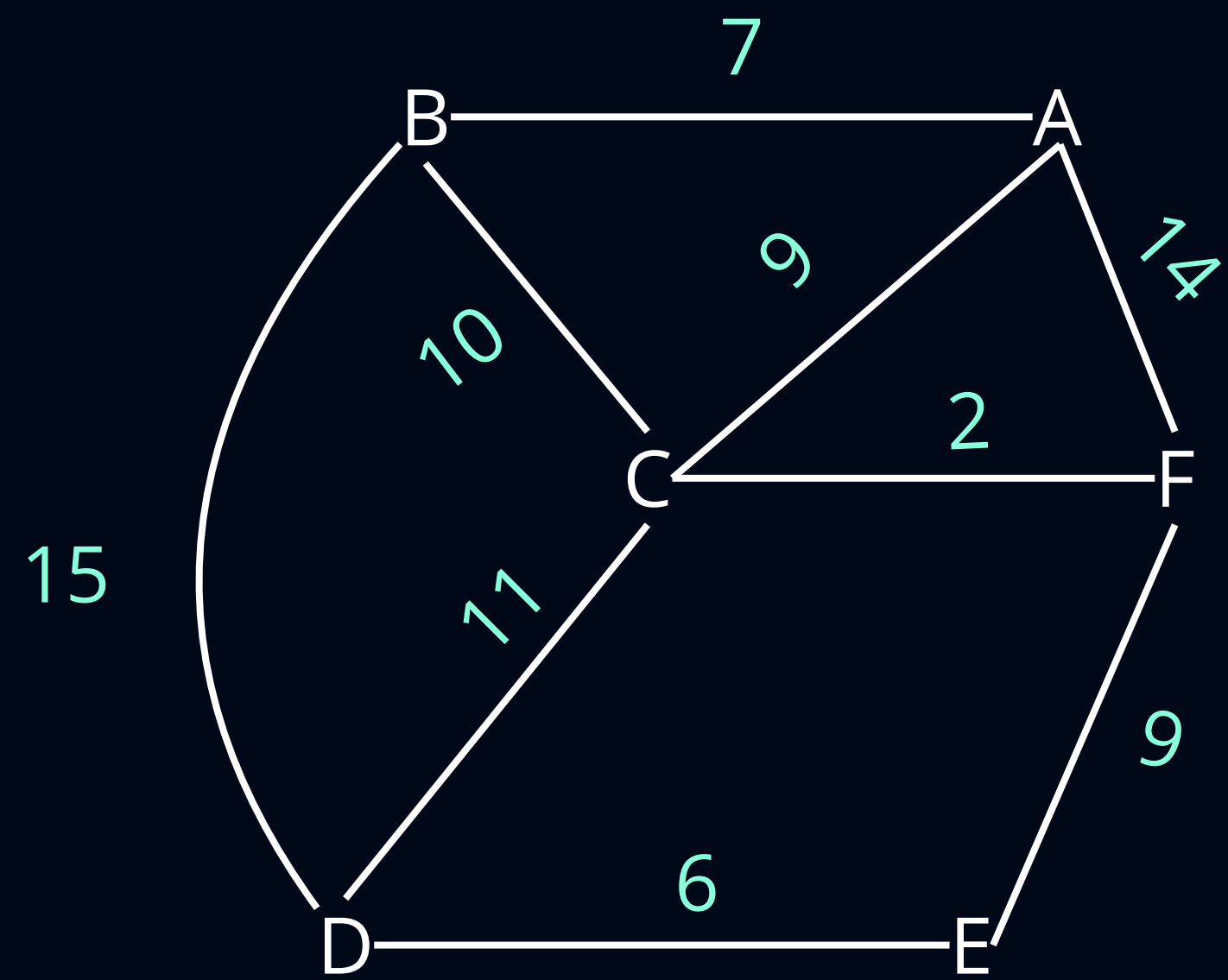
```
1 package com.dijkstra;
2
3 Codeium: Explain
4 public class Main {
5     private static final Graph.Edge[] GRAPH = {
6         new Graph.Edge(v1:"a", v2:"b", dist:7),
7         new Graph.Edge(v1:"a", v2:"c", dist:9),
8         new Graph.Edge(v1:"a", v2:"f", dist:14),
9         new Graph.Edge(v1:"b", v2:"c", dist:10),
10        new Graph.Edge(v1:"b", v2:"d", dist:15),
11        new Graph.Edge(v1:"c", v2:"d", dist:11),
12        new Graph.Edge(v1:"c", v2:"f", dist:2),
13        new Graph.Edge(v1:"d", v2:"e", dist:6),
14        new Graph.Edge(v1:"e", v2:"f", dist:9),
15    };
16    private static final String START = "a";
17    private static final String END = "e";
18
19    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | X
20    public static void main(String[] args) {
21        //amazonQ dijkstraAmazonq = new amazonQ();
22        theAlgorithms dijkstraTheAlgorithms = new theAlgorithms();
```

```
18 public static void main(String[] args) {
19     //amazonQ dijkstraAmazonq = new amazonQ();
20     theAlgorithms dijkstraTheAlgorithms = new theAlgorithms();
21
22     int[][] graph = {
23         {0, 4, 0, 0, 0, 0, 8, 0},
24         {4, 0, 8, 0, 0, 0, 11, 0},
25         {0, 8, 0, 7, 0, 4, 0, 2},
26         {0, 0, 7, 0, 9, 14, 0, 0},
27         {0, 0, 0, 9, 0, 10, 0, 0},
28         {0, 0, 4, 14, 10, 0, 2, 0},
29         {0, 0, 0, 0, 0, 2, 0, 1, 6},
30         {8, 11, 0, 0, 0, 0, 1, 0, 7},
31         {0, 0, 2, 0, 0, 0, 6, 7, 0}
32     };
33
34     Graph g = new Graph(GRAPH);
35     g.dijkstra(START);
36
37     dijkstraTheAlgorithms.dijkstra(graph, src:0);
38     blackBox.dijkstra(graph, startNode:0);
39     chatGPT.dijkstra(graph, src:0);
40     codeium.dijkstra(graph, source:0);
41     googleBard.dijkstra(graph, src:0);
42
43
44 }
45 }
```

Grafo por Matriz de Adjacência.



Array de Arestas.



Resultado.

| | AmazonQ | BlackBox | ChatGPT | Codeium | Copilot | Gemini |
|------------------|---------|----------|---------|---------|---------|--------|
| <i>Correct</i> | | X | X | X | X | X |
| <i>Plausible</i> | | | | | | |
| <i>Incorrect</i> | | | | | | |
| <i>Invalid</i> | X | | | | | |
| | | | | | | |

Referências.

- CORSO, Vincenzo et al. **Generating Java Methods: An Empirical Assessment of Four AI-Based Code Assistants.** arXiv preprint arXiv:2402.08431, 2024. Acesso em fevereiro de 2024.
- LIBÓRIO, Felipe Tenório de Holanda Rocha. **Análise da Ferramenta ChatGPT para a Geração de Código-fonte de Algoritmos de Ordenação em Diferentes Linguagens de Programação.** Brasil. Acesso em fevereiro de 2024.