PIBIC - UFAPE

Análise da corretude de algoritmos gerados por Inteligência Artificial

Comparação deles com algoritmos de repositórios existentes

Índice.

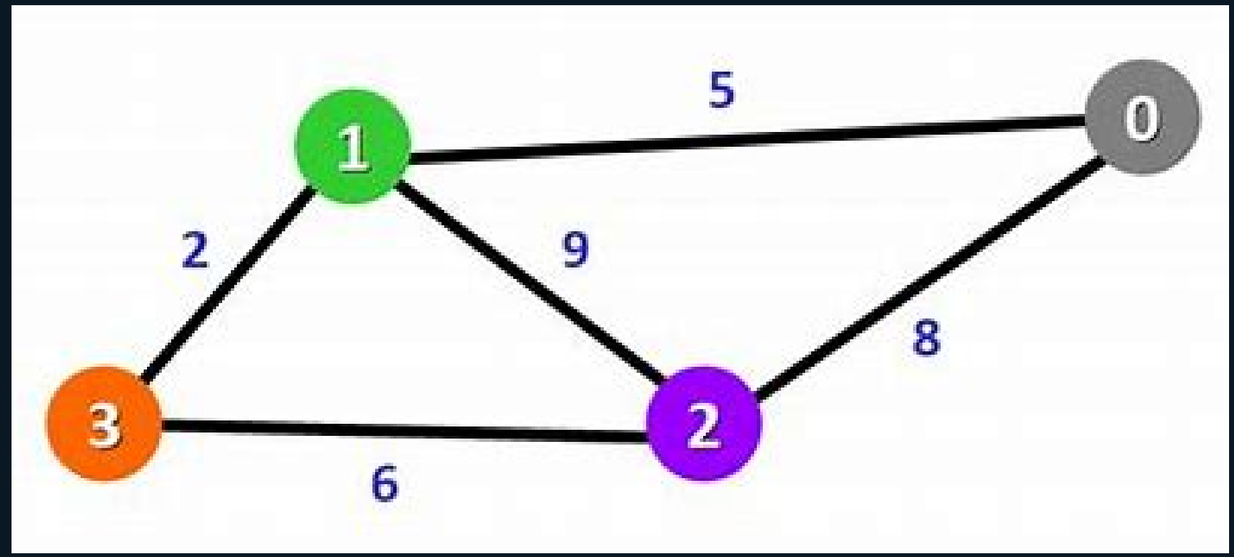
Objetivos.

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.

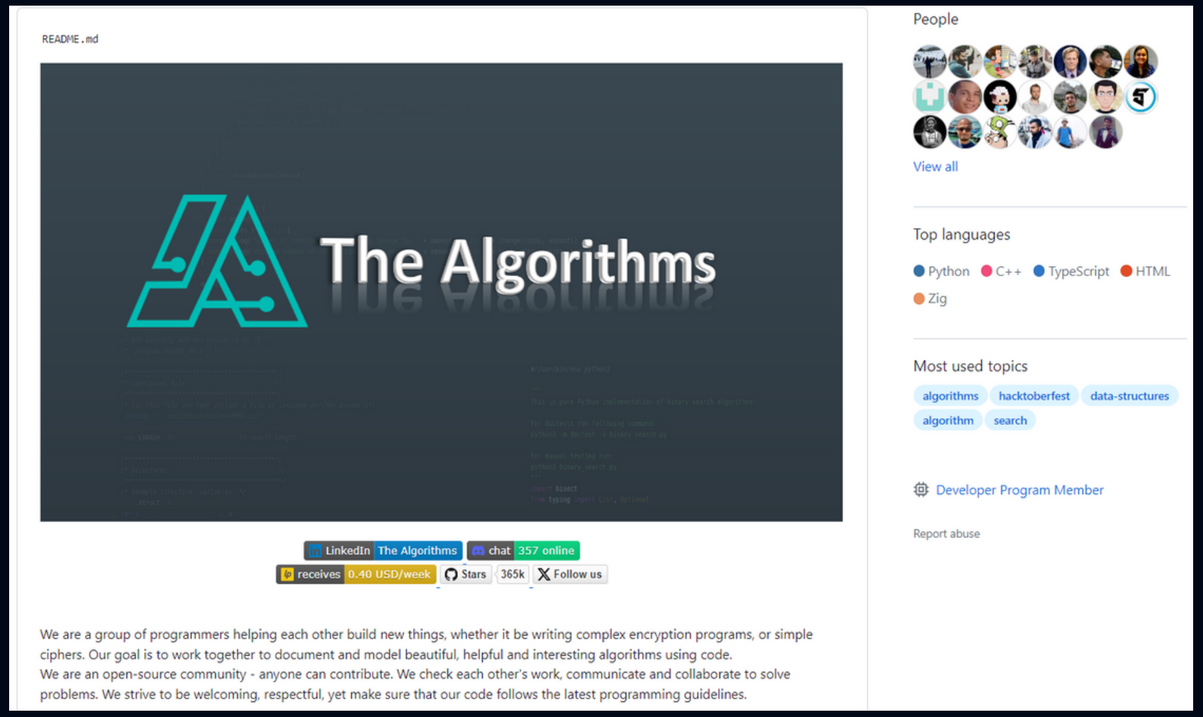
PIBIC / com / dijkstra /

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.



Tecnologias usadas.

Tecnologias usadas.

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

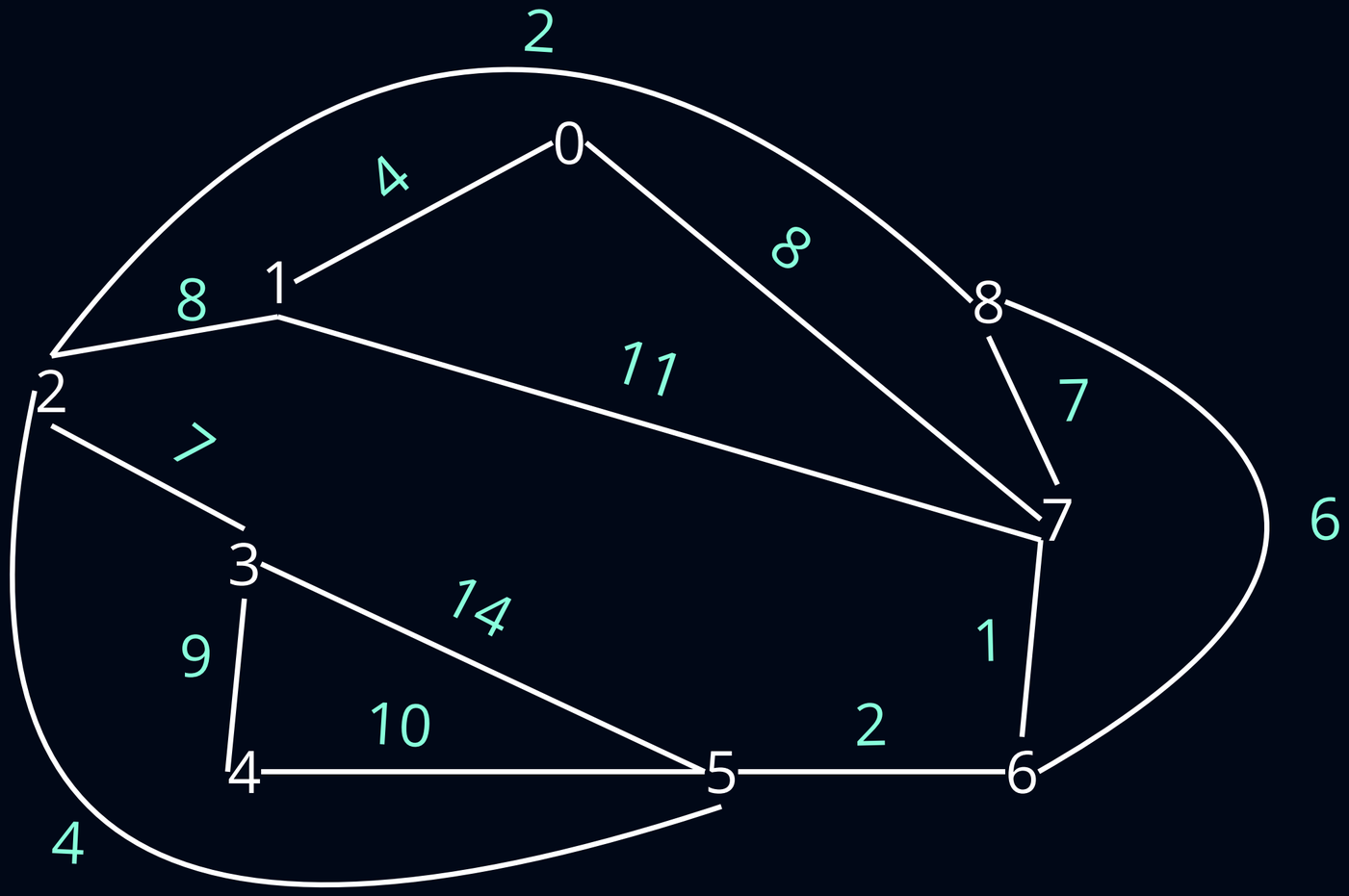
Teste.

Main.java 1, U

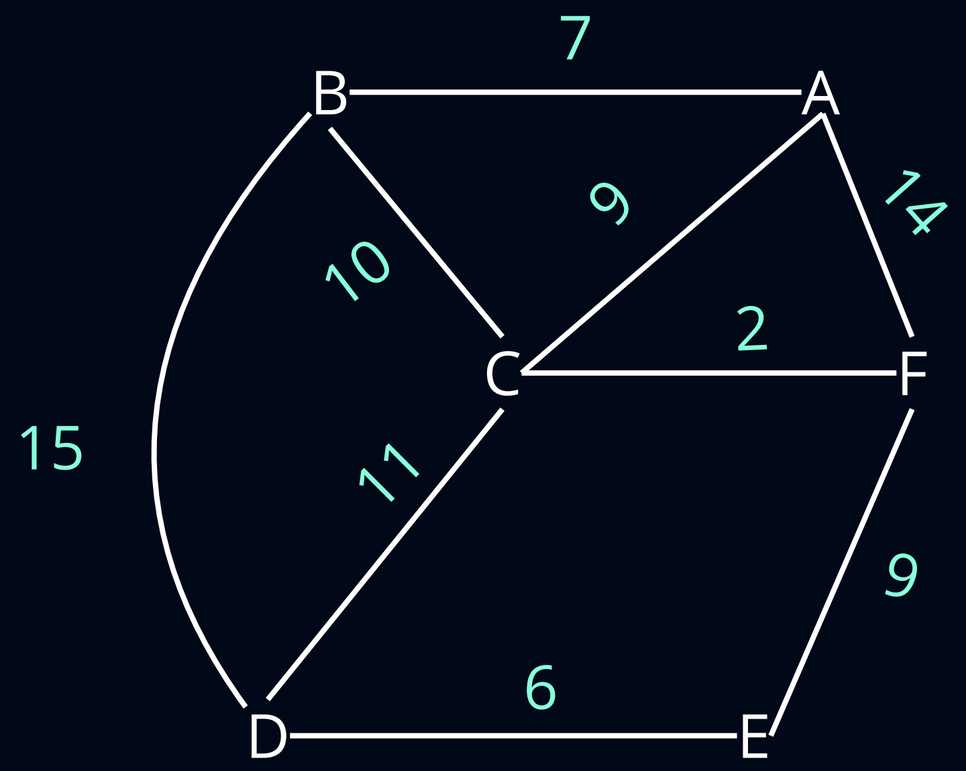
com > dijkstra > Main.java > ...

```java
package com.dijkstra;

Codeium: Explain
public class Main {
    private static final Graph.Edge[] GRAPH = {
        new Graph.Edge(v1:"a", v2:"b", dist:7),
        new Graph.Edge(v1:"a", v2:"c", dist:9),
        new Graph.Edge(v1:"a", v2:"f", dist:14),
        new Graph.Edge(v1:"b", v2:"c", dist:10),
        new Graph.Edge(v1:"b", v2:"d", dist:15),
        new Graph.Edge(v1:"c", v2:"d", dist:11),
        new Graph.Edge(v1:"c", v2:"f", dist:2),
        new Graph.Edge(v1:"d", v2:"e", dist:6),
        new Graph.Edge(v1:"e", v2:"f", dist:9),
    };
    private static final String START = "a";
    private static final String END = "e";

    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc |
    public static void main(String[] args) {
        //amazonQ dijkstraAmazonq = new amazonQ();
        theAlgorithms dijkstraTheAlgorithms = new theAlgorithms();

        int[][] graph = {
            {0, 4, 0, 0, 0, 0, 0, 8, 0},
            {4, 0, 8, 0, 0, 0, 0, 11, 0},
            {0, 8, 0, 7, 0, 4, 0, 0, 2},
            {0, 0, 7, 0, 9, 14, 0, 0, 0},
            {0, 0, 0, 9, 0, 10, 0, 0, 0},
            {0, 0, 4, 14, 10, 0, 2, 0, 0},
            {0, 0, 0, 0, 0, 2, 0, 1, 6},
            {8, 11, 0, 0, 0, 0, 1, 0, 7},
            {0, 0, 2, 0, 0, 0, 6, 7, 0}
        };

        Graph g = new Graph(GRAPH);
        g.dijkstra(START);

        dijkstraTheAlgorithms.dijkstra(graph, src:0);
        blackBox.dijkstra(graph, startNode:0);
        chatGPT.dijkstra(graph, src:0);
        codeium.dijkstra(graph, source:0);
        googleBard.dijkstra(graph, src:0);


    }
}
```

Grafo por Matriz de Adjacência.



Array de Arestas.



Resultado.

	AmazonQ	BlackBox	ChatGPT	Codeium	Copilot	Gemini
Correct		X	X	X	X	X
Plausible						
Incorrect						
Invalid	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.