TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI

**KHOA CÔNG NGHỆ THÔNG TIN**

----- 🙡 🕮 🙣 -----

Icon

Description automatically generated

**BÁO CÁO ĐỒ ÁN**

**Lập trình hướng đối tượng**

***Đề tài:* Bài toán tìm đường đi ngắn nhất**

Giảng viên: **Lê Đức Quang**

Nhóm sinh viên thực hiện: 01

|  |  |  |
| --- | --- | --- |
| STT | Họ và tên | MSSV |
| 1 | Mai Ngọc Đoàn | 1509765 |
| 2 | Nguyễn Mai Nhật Linh |  |

**Hà Nội, năm 2022**

# I, Mô tả bài toán

1, Đề bài

Bài toán mà nhóm chọn:

*Đường đi ngắn nhất: Xây dựng chương trình giải quyết bài toán tìm đường đi ngắn nhất trong một đồ thị: Cho trước một đồ thị gồm V đỉnh, E cạnh và một hàm trọng số giá trị thực f: E → R. Nếu cho trước một đỉnh v E, hãy tìm đường P để đi từ v đến một đỉnh v’ nào đó sao cho: ∑ p∈P f (P) là nhỏ nhất trong số tất cả các con đường nối v với v’. Có thể sử dụng một trong các thuật toán sau để giải quyết bài toán này (thuật toán Dijkstra, thuật toán Bellman-Ford, hoặc thuật toán heuristics)*

Như vậy, để giải quyết bài toán đặt ra, chúng ta có nhiều cách chọn thuật toán để giải quyết, ở đây nhóm chọn ***Dijkstra*** để làm thuật toán chính trong đề tài này.

2, Mô tả chi tiết

2.1 Mô tả về thuật toán

Để xây dựng bài toán này, chúng ta cần hiểu về thuật toán Dijkstra (Dijkstra Algorithm). Đây là thuật toán giúp duyệt cây đồ thị và tìm đường đi có giá cực tiểu từ một điểm đến một điểm bất khì khác. Thuật toán này có nhiều dạng biến thể khác nhau. Thuật toán gốc ban đầu sẽ cho biết đường đi ngắn nhất giữa 2 điểm cho trước còn biến thể phổ biến của nó là tìm đường đi ngắn nhất từ một điểm đến tất cả các điểm còn lại, tạo ra một cây lời giải chứa các đường đi đó (shortest – path tree).

Ở đồ án của nhóm, chúng em thiên về việc tìm đường đi giữa chỉ 2 điểm cho trước mà không tạo ra toàn bộ cây lời giải đến tất cả các điểm khác, hay nói cách khác, đây chính là biến thể gốc ban đầu của Dijkstra.

2.2 Triển khai bài toán.

2.2.1 Lựa chọn công nghệ.

- Java: Java là ngôn ngữ lập trình hướng đối tượng (OOP) phổ biến, dễ sử dụng và triển khai. Thêm vào đó, nó còn hỗ trợ SWING – một bulit – in library dùng cho việc thiết kế giao diện ứng dụng (GUI).

- SQL Server: Trong các bài toán thiên về thuật toán như thế này, nhóm nhận định không cần dùng quá nhiều đến Database, mục đích của Database chỉ cần để lưu trữ những thông tin cơ bản của đồ thị.

- Các hình mẫu thiết kế (Design Pattern):

+ MVC: Xây dựng các cách triển khai code theo kiểu MVC, trong đó phần Model sẽ đảm nhiệm chức năng liên hệ với Database, View sẽ đảm nhiệm việc tương tác với người dùng, phần thực thi thuật toán Dijkstra sẽ giao cho Controller.

+ Singleton: Đây là hình mẫu kinh điển cho những bài toán sử dụng Database. Khi đó, nó sẽ đảm bảo chỉ có một biến *Connection* kết nối đến Database được tạo ra trong suốt quá trình chạy của chương trình.

+ Observer. Sẽ được nói cụ thể hơn vào phần sau.

2.2.2 Phân tích cấu trúc dữ liệu.

Để giải quyết bài toán đồ thị vào trong lập trình, điều đầu tiên đó là biểu diễn được đồ thị đó thông qua các cấu trúc dữ liệu. Để biểu diễn một điểm trong đồ thị, ta sử dụng cấu trúc sau:

*public class* Node *implements* Comparable<Node> {  
 *private final* String name;  
 *private int* estimate;

*private final* HashMap<Node, Integer> adjacentNodes;

// methods below

}

Với:

* name: lưu trữ tên của đỉnh đó
* estimate: dùng cho thuật toán Dijkstra, biểu diễn cho chi phí từ điểm Source đến chính nó bằng bao nhiêu.
* adjacentNodes: lưu trữ các Node là con (direct – children) của nó.

Với cách xây dựng như thế này, ta sẽ có một đồ thị một chiều (directed – graph), tức là nếu B là con của A, sẽ có đường nối từ A đến B, điều ngược lại chưa chắc đúng.

2.3 Các sơ đồ thiết kế:

(Trang bên)

2.3.1 Sơ đồ Usecase:

Diagram

Description automatically generated

2.3.2 Sơ đồ Package:

Diagram

Description automatically generated

2.3.3: Sơ đồ Class:

Diagram, engineering drawing, schematic

Description automatically generated

Trong đó:

* *Dijkstra:* là class chứa các logic thuật toán Dijkstra
* *Parser* : Dùng để biến đổi dạng file \*.dot (một file dùng để biểu diễn đồ thị) sang thành một dãy các *Node* có mối quan hệ tương đương.
* *AppDB:* thực hiện kết nối đến cơ sở dữ liệu. Class này được implement theo mô hình thiết kế Singleton.
* *GraphvizFileFormatException:* một exception, xảy ra nếu file \*.dot đầu vào có dữ liệu không hợp lệ.

Dữ liệu đầu vào là hợp lệ khi nó có dạng:

Table

Description automatically generated

Với *graph* là tên loại kiểu đồ thị, nó có thể là graph (hai chiều) hay digraph (một chiều). Tìm hiểu thêm về file \*.dot tại: [DOT (graph description language) - Wikipedia](https://en.wikipedia.org/wiki/DOT_(graph_description_language))

* *NoDataException:* Gần giống chức năng như *GraphvizFileFormatException* nhưng nó sẽ báo lỗi khi file đầu vào không có dữ liệu hoặc graph rỗng.
* *PathNotFoundException:* Nếu trong quá trình tìm đường đi từ S tới G, khi duyệt hết tất cả phần tử mà không tìm thấy đường nối giữa S và G thì sẽ sinh ra Exception này.
* *Controller:* Một abstract class dùng để them các sự kiện người dùng cho dứng dụng chính, từ mẫu Controller này sẽ viết các class khác kế thừa từ nó để “control” cho một sự kiện cụ thể.
* *Subject và Observer:* Hai interface chính cho hình mẫu Observer (Observer Pattern). Việc thiết kế hình mẫu này là để kết nối dữ liệu giữa 2 JFrame: *MainApp* và *FileSelectorView.* MainApp sẽ implement Subject và FileSelectorView sẽ kế thừa Observer. Khi người dùng chọn được file mình mong muốn để làm dữ liệu đầu vào, FileSelectorView (Observer) sẽ kích hoạt hàm update của MainApp (Subject) và truyền file vừa chọn cho MainApp thực hiện việc đọc dữ liệu.

2.3.4 Component Diagram

Diagram

Description automatically generated

2.3.5 Sơ đồ thiết kế bảng dữ liệu trong SQL

Dữ liệu chỉ chứa trong một bảng duy nhất

* Sơ đồ ER

Diagram

Description automatically generated

* Sơ đồ ERD:

Table

Description automatically generated

# II, Mô tả ứng dụng.

1, Cài đặt ứng dụng:

* Ứng dụng được viết bằng ngôn ngữ Java, Jre18
* Yêu cầu để chạy:
* Compile và Run trên nền tảng IntelliJ IDEA.
* JRE18
* Database dựa theo sơ đồ thiết kế ở trên (mục I – 2.3.5).

2, Source Files:

Có các cách để download Project online sau:

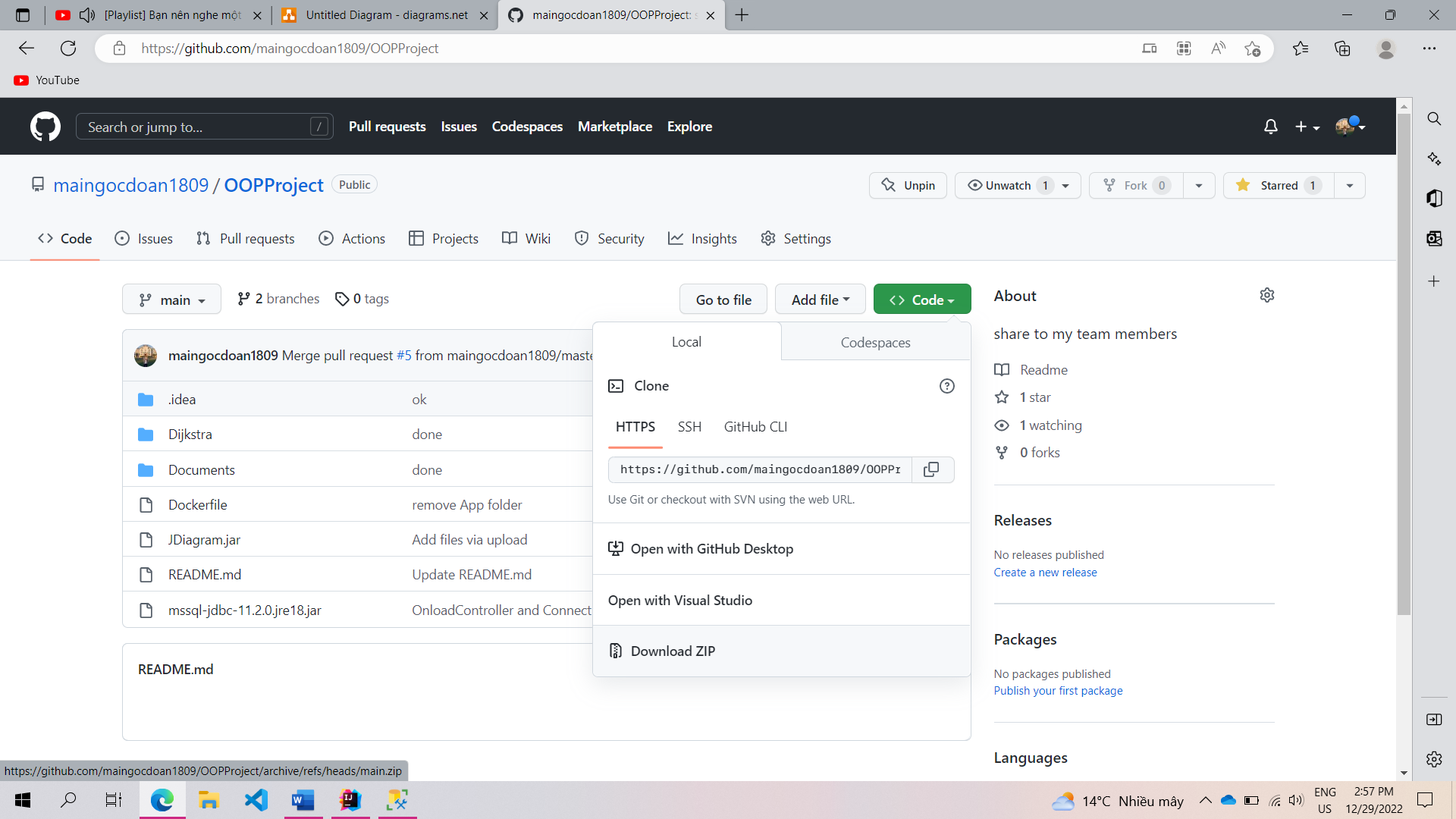
Cách 1: Dùng git và github:

* Tải file zip

B1, Truy cập: <https://github.com/maingocdoan1809/OOPProject>

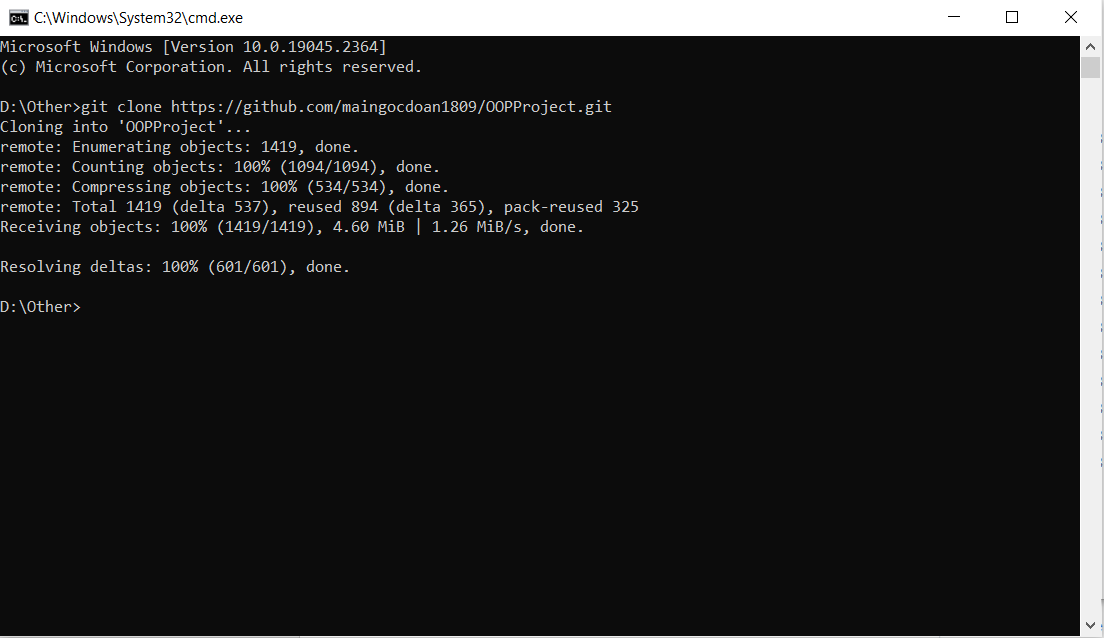
B2: Có 2 cách download:

+ Nhấn vào nút <> Code màu xanh, chọn Download ZIP và tải về máy:



* Clone bằng git local trên máy tính, vào command prompt trên máy tính, target đến thư mục cần lưu project và nhập:

git clone <https://github.com/maingocdoan1809/OOPProject.git>



Nếu màn hình xuất hiện như thế này thì là đã download thành công.

Cách 2: Download qua google Drive:

Truy cập vào <https://m5.gs/bE94M2> để tải.

3, Chạy chương trình.

3.1 Giao diện:

* Giao diện chính

Graphical user interface, application

Description automatically generated

* Giao diện chọn dữ liệu đầu vào:

Trong bảng Available testcases, chọn 1 graph đầu vào mà bạn đã insert sẵn trong SQL server.

Graphical user interface, text, application

Description automatically generated

Click vào *Choose* để thực hiện import.

- Sau khi chọn xong, giao diện kết quả là:Graphical user interface

Description automatically generated with medium confidence

Ta có thể thực hiện thay đổi cách bố trí các node, màu sắc của chúng, cũng như độ thu phóng to nhỏ bằng thanh công cụ ở đây (Tùy ý):

Graphical user interface, text, application

Description automatically generated

* Sau khi điều chỉnh các thông số xong, tiến hành chọn điểm bắt đầu và điểm kết thúc ở panel bên trái, nếu thay đổi Source thì vào nút *From/Root,* nếu thay đổi điểm đích Goal/Dest thì vào *To*

Graphical user interface

Description automatically generated with medium confidence

* Nếu bạn muốn đồ thị không duyệt qua một nút bất kì thì chọn vào ô *Block*

Ví dụ bạn muốn bỏ qua node C, nhấn chọn nút C ở ô *Block* và nhấn nút Block ở dưới cùng, Node đó sẽ được thêm vào bảng. Bạn có thể chỉnh màu cho nó cho dễ phân biệt ở thanh công cụ.

A picture containing graphical user interface

Description automatically generated

* Nhấn start để bắt đầu tìm đường đi

A picture containing graphical user interface

Description automatically generated

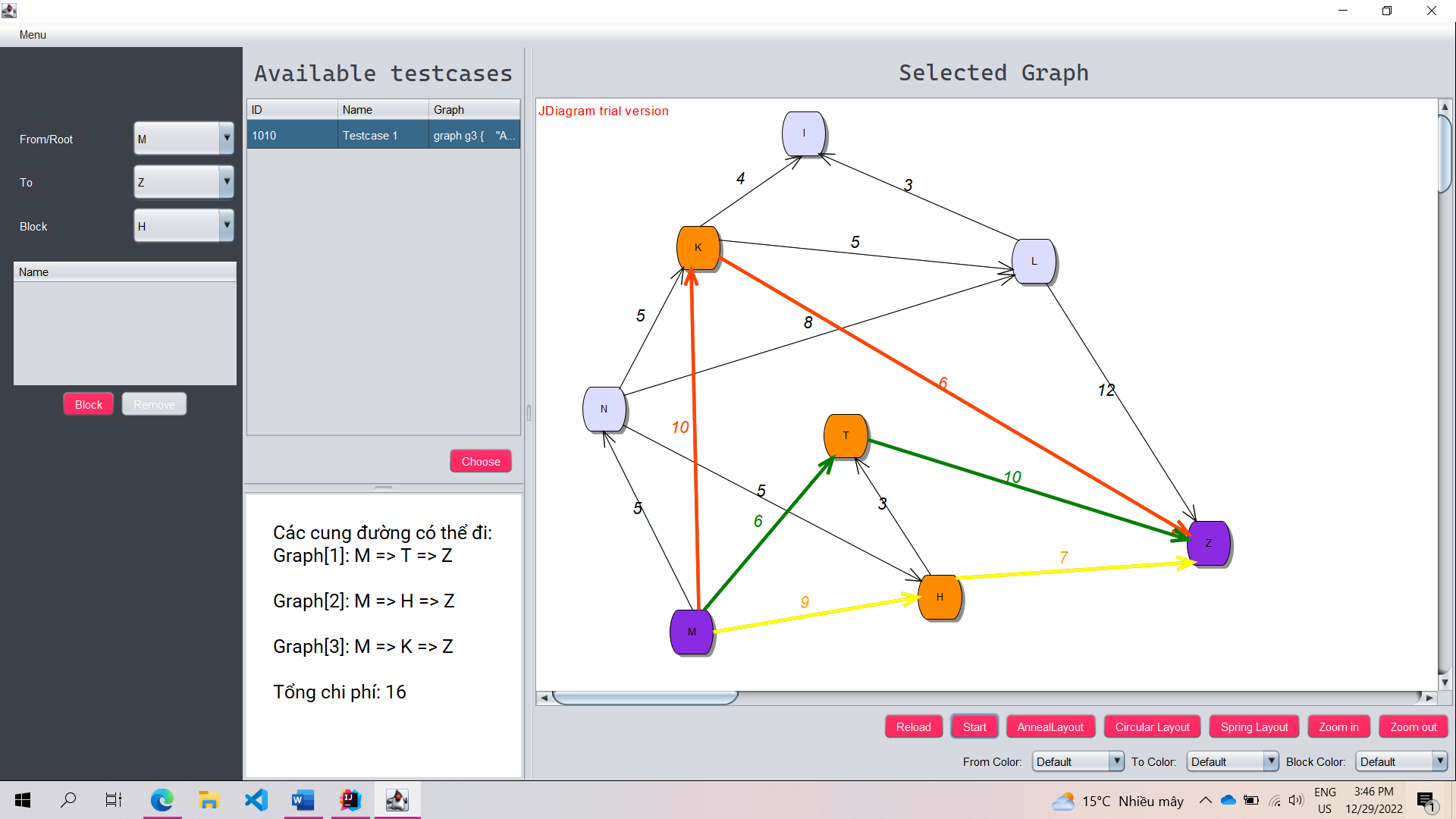
Đường đi sẽ hiện lên trên màn hình bằng đường nối có màu. Có thể có 3 màu, trong đó:

- Màu xanh biểu hiện cho đoạn đường tốt nhất, nhưng đi qua ít đỉnh nhất.

- Màu vàng biểu hiện cho đoạn đường tốt nhất, nhưng đi qua nhiều điểm hơn hoặc bằng màu xanh

- Màu vàng cam biểu hiện cho đoạn đường tốt nhất, nhưng đi qua nhiều hơn hoặc bằng màu vàng.

- Ví dụ trường hợp đó:



* Kết quả bài toán sẽ hiện ở góc dưới bên trái, cạnh panel bên trái

Graphical user interface, application

Description automatically generated

* Nếu bạn muốn thay đổi đích đến, chọn một đỉnh trong nút *To,* và không muốn nó đi qua đỉnh C:

A picture containing application

Description automatically generated

* Block C:

A picture containing application

Description automatically generated

* Start:

A picture containing graphical user interface

Description automatically generated

* Xóa C ra khỏi danh sách chặn:

Graphical user interface, application

Description automatically generated

* Thử nhấn start lại lần nữa:

A picture containing application

Description automatically generated

Có thể thấy, nếu bỏ C ra khỏi danh sách chặn của A, đường đi bây giờ đã thay đổi.

* Nếu bạn muốn xem lại thì có thể nhấn nút Restart rồi Start, hoặc đơn giản là Start.

Nếu muốn thực hiện truy nhập bằng file, hãy chọn Menu ở phía trên góc trái màn hình:

A picture containing application

Description automatically generated

* Cửa sổ chọn file sẽ hiện ra, chọn đến file bạn muốn chọn và nhấn ok. Lưu ý, cửa sổ chọn file chỉ xuất hiện các file dạng .txt và .dot.

Graphical user interface, application, Word

Description automatically generated

* Nhấn open, ta thu được đồ thị của file đó, các bước thực hiện tiếp theo tương tự như chọn ở Database.

Application

Description automatically generated with low confidence