BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP THỰC PHẨM TP. HCM

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN

Môn học: Trí tuệ nhân tạo

Đề tài:

TÌM LỘ TRÌNH XE THU GOM RÁC TỐI ƯU THÀNH PHỐ HCM

Giảng viên hướng dẫn: TS. Văn Thế Thành

Nhóm thực hiện: **Condors**

Các thành viên: **Đặng Hoàng Cẩm My – 2001180476**

Nguyễn Ngọc Hải – 2001181090

Nguyễn Hồng Phúc – 2001181265

TP. HỒ CHÍ MINH, tháng 08 năm 2020

LỜI CẢM ƠN

Trong thời gian nghiên cứu và học tập môn học Trí tuệ nhân tạo cùng với nội dung bài tập lớn về chương trình tìm lộ trình xe thu gom rác tối ưu ở thành phố Hồ Chí Minh, chúng em gặp không ít khó khăn về cách làm và kiến thức. Tuy nhiên chúng em đã nhận được rất nhiều sự giúp đỡ quý báu của các thầy cô giáo và các bạn, nhóm chúng em đã hoàn thành chương trình và báo cáo bài tập lớn môn Trí tuệ nhân tạo với đề tài chương trình tìm lộ trình xe thu gom rác tối ưu ở thành phố Hồ Chí Minh. Nếu không có sự giúp đỡ của thầy cô và các bạn, nhóm em sẽ rất khó để có thể hoàn thiện được.

Nhóm em xin gửi lời cám ơn chân thành đến Ban giám hiệu trường Đại học Công nghiệp thực phẩm Thành phố Hồ Chí Minh đã tạo cho chúng em điều kiện học tập tốt nhất để phát triển và hoàn thiện bản thân.

Để có được thành quả này, cho phép nhóm em được bày tỏ lời cám ơn đến các thầy cô giáo trong khoa Công nghệ thông tin trường Đại học Công nghiệp thực phẩm Thành phố Hồ Chí Minh đã truyền đạt cho chúng em những kiến thức về lý thuyết và thực hành để chúng em có nền tảng

Đồng thời, nhóm em xin gửi lời cám ơn đặc biệt về sự hướng dẫn và chỉ bảo tận tình của giáo viên hướng dẫn thầy Văn Thế Thành đã tận tình hướng dẫn và giúp đỡ nhóm em trong suốt quá trình hoàn thành chương trình cũng như bài báo cáo này. Em không chỉ nhận được những kiến thức bổ ích mà còn là hành trang theo em trong suốt thời gian học tập và làm việc sau này.

Do thời gian có hạn, cũng như kinh nghiệm còn thiếu nên trong chương trình báo cáo này sẽ không tránh khỏi sai sót, hạn chế nhất định. Những ý kiến nhận xét góp ý của thầy cô và các bạn là cơ sở để chúng em học hỏi và hoàn thiện kiến thức của mình. Em rất mong nhận được sự góp ý của thầy cô và các bạn để báo cáo đạt được kết quả tốt hơn

Lời cuối cùng, em xin kính chúc Thầy nhiều sức khỏe, thành công và hạnh phúc.

Em xin chân thành cám ơn !

Thành phố Hồ Chí Minh, tháng 08 năm 2020

LỜI CAM ĐOAN

Nhóm em xin cam đoan

1. Những nội dung trong bài báo cáo này là do nhóm em thực hiện dưới sự hướng dẫn trực tiếp của thầy Văn Thế Thành

2. Mọi tham khảo dùng trong báo cáo đều có trích dẫn rõ ràng tên tác giả, tên công trình, thời gian, địa điểm công bố

3. Mọi sao chép không hợp lệ, vi phạm quy chế đào tạo, hay gian trá, nhóm em xin chịu hoàn toàn trách nhiệm

Sinh viên

Nguyễn Ngọc Hải

Đặng Hoàng Cẩm My

Nguyễn Hồng Phúc

Xin cam đoan

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

Tính tương tác của nhóm trong quá trình làm đồ án/báo cáo: .....

.....

.....

.....

.....

.....

Đánh giá hình thức và nội dung thuyết minh: .....

.....

.....

.....

.....

Đánh giá sản phẩm: .....

.....

.....

.....

.....

.....

Kết luận: .....

.....

.....

.....

Ngày .....tháng…… năm ……

Giảng viên hướng dẫn

(Ký và ghi rõ họ tên)

LỊCH LÀM VIỆC CỦA NHÓM

Thứ 2	19h – 22h
Thứ 3	19h – 22h
Thứ 4	19h – 22h
Thứ 5	19h – 22h
Thứ 6	19h – 22h
Thứ 7	19h – 22h
Chủ nhật	09h – 17h

BẢNG PHÂN CÔNG

Họ tên	MSSV	Công việc	Điểm
Nguyễn Ngọc Hải	2001181090	○ Lên ý tưởng ○ Tìm hiểu thuật toán GTS ○ Coding thuật toán GTS2 ○ Chỉnh sửa, debug ○ Kết nối các lớp	
Đặng Hoàng Cẩm My	2001180476	○ Lên ý tưởng ○ Tìm hiểu thuật toán GTS ○ Thiết kế giao diện ○ Coding điểm 2D ○ Vẽ đường đi ○ Viết báo cáo word	
Nguyễn Hồng Phúc	2001181265	○ Lên ý tưởng ○ Tìm hiểu thuật toán GTS ○ Tạo database ○ Liên kết database ○ Coding thuật toán GTS1 ○ Viết báo cáo word	

DANH MỤC

GIỚI THIỆU

1. Phạm vi nghiên cứu

- ✓ Tập trung nghiên cứu về thuật giải GTS để tìm đường đi ngắn nhất.
- ✓ Áp dụng thuật giải vào chương trình tìm lộ trình gom rác tối ưu ở Thành phố Hồ Chí Minh với đường đi ngắn nhất và chi phí thấp nhất.

2. Sự cần thiết và lí do chọn đề tài:

Ngày nay, cùng với sự phát triển mọi mặt của xã hội, ngành Công nghệ thông tin đã trở thành một nhu cầu không thể thiếu trong đời sống con người.

Nền khoa học máy tính ngày nay đang giữ một vị trí trung tâm trong hầu hết các lĩnh vực của xã hội. Với những lợi ích do công nghệ thông tin mang lại, nhóm em đã đưa những ứng dụng tin học vào chương trình tìm lộ trình xe thu gom rác tối ưu của Thành phố Hồ Chí Minh.

Là một đề tài mang tính thực tiễn cao cùng với môi trường ngày càng ô nhiễm, nhiều chỗ xử lý rác chưa được tối ưu. Vì vậy nhóm em chọn đề tài "Tìm lộ trình xe thu gom rác tối ưu của Thành phố Hồ Chí Minh" . Đề tài này sẽ đưa ra được những nhận xét, những đánh giá tổng thể và từ đó đưa ra được tuyến đường tốt nhất dựa trên sự hổ trợ của máy tính. Chương trình được xây dựng trên nền tảng Java.

3. Mục tiêu

Nhằm giải quyết và đáp ứng một cách hiệu quả các nhu cầu về việc tìm đường đi ngắn nhất. Tin học hóa trong công tác tìm đường đi ngắn nhất nhằm rút gọn bớt sức lao động của con người, tiết kiệm được thời gian, độ chính xác cao, nhanh gọn và tiện lợi hơn nhiều so với việc tìm đườmg đi ngẫu nhiên như trước đây. Tin học hóa giúp thu hẹp không gian lưu trữ, tránh thất lạc dữ liệu, hệ thống tự động hóa các thông tin theo nhu cầu của con người.

4. Cấu trúc đồ án

Phần 1: Cơ sở lý thuyết.

Phần 2: Tìm hiểu giải thuật GTS tìm đường đi ngắn nhất.

Phần 3: Demo ứng dụng chương trình tìm lộ trình gom rác tối ưu ở Thành phố Hồ Chí Minh bằng ngôn ngữ Java.

PHẦN I. CƠ SỞ LÝ THUYẾT

1. Giới thiệu về ngôn ngữ Java

Java là một ngôn ngữ lập lập trình, được phát triển bởi **Sun Microsystem** vào năm 1995, là ngôn ngữ kế thừa trực tiếp từ C/C++ và là một ngôn ngữ lập trình hướng đối tượng.

Ngày nay Java được sử dụng với các mục đích sau:

- Phát triển ứng dụng cho các thiết bị điện tử thông minh, các ứng dụng cho doanh nghiệp với quy mô lớn.
- Tạo các trang web có nội dung động (web applet), nâng cao chức năng của server.
- Phát triển nhiều loại ứng dụng khác nhau: Cơ sở dữ liệu, mạng, Internet, viễn thông, giải trí,...

Tiêu chí hàng đầu của Ngôn ngữ Lập trình Java là **"Write Once, Run Anywhere"** (*Viết một lần, chạy mọi nơi),* nghĩa là Java cho phép chúng ta viết code một lần và thực thi được trên các hệ điều hành khác nhau. Ví dụ, chúng ta viết code trên Hệ điều hành Windows và nó có thể thực thi được trên các Hệ điều hành Linux và Mac OS...

Với đặc điểm nổi bật đó, Java có những đặc điểm cơ bản như sau:

- Đơn giản và quen thuộc: Vì Java kế thừa trực tiếp từ C/C++ nên nó có những đặc điểm của ngôn ngữ này, Java đơn giản vì mặc dù dựa trên cơ sở C++ nhưng Sun đã cẩn thận lược bỏ các tính năng khó nhất của của C++ để làm cho ngôn ngữ này dễ sử dụng hơn.
- Hướng đối tượng và quen thuộc.
- Mạnh mẽ (thể hiện ở cơ chế tự động thu gom rác - Garbage Collection) và an toàn.
- Kiến trúc trung lập, độc lập nền tảng và có tính khả chuyển (*Portability*).
- Hiệu suất cao.
- Máy ảo (biên dịch và thông dịch).
- Phân tán.

- Đa nhiệm: Ngôn ngữ Java cho phép xâ dựng trình ứng dụng, trong đó nhiều quá trình có thể xảy ra đồng thời. Tính đa nhiệm cho phép các nhà lập trình có thể biên soạn phần mềm đáp ứng tốt hơn, tương tác tốt hơn và thực hiện theo thời gian thực.
- …

2. Lịch sử phát triển của Trí tuệ nhân tạo

Lịch sử của trí tuệ nhân tạo cho thấy ngành khoa học này có nhiều kết quả đáng ghi nhận. Theo các mốc phát triển, người ta thấy trí tuệ nhân tạo được sinh ra từ những năm 50 với các sự kiện sau:

- Turing được coi là người khai sinh ngành trí tuệ nhân tạo bởi phát hiện của ông về máy tính có thể lưu trữ chương trình và dữ liệu

-Tháng 8/1956 J. Mc Carthy, M. Minsky, A. Newell, Shannon.Simon,… đưa ra khái niệm "trí tuệ nhân tạo".

- Vào khoảng năm 1960 tại Đại học MIT (Massachussets Institure of Technology) ngôn ngữ LISP ra đời, phù hợp với các nhu cầu xử lý đặc trưng của trí tuệ nhân tạo - đó là ngôn ngữ lập trình đầu tiên dùng cho trí tuệ nhân tạo.

- Thuật ngữ trí tuệ nhân tạo được dùng đầu tiên vào năm 1961 cũng tại MIT.

- Những năm 60 là giai đoạn lạc quan cao độ về khả năng làm cho máy tính biết suy nghĩ. Trong giai đoạn này người ta đã được chứng kiến máy chơi cờ đầu tiên và các chương trình chứng minh định lý tự động. Cụ thể:

+ 1961: Chương trình tính tích phân bất định.

+ 1963: Các chương trình Heuristics: Chương trình chứng minh các định lý hình học không gian có tên là "tương tự", chương trình chơi cờ của Samuel, tìm kiếm đường đi có giá thành cực tiểu.

+ 1964: Chương trình giải phương trình đại số sơ cấp, chương trình trợ giúp ELIZA (có khả năng làm việc giống như một chuyên gia phân tích tâm lý).

+ 1966: Chương trình phân tích và tổng hợp tiếng nói.

+ 1968: Chương trình điều khiển người máy (Robot) theo đồ án

"Mát-tay", chương trình học nói.

- Vào những năm 60, do giới hạn khả năng của các thiết bị, bộ nhớ và đặc biệt là yếu tố thời gian thực hiện nên có sự khó khăn trong việc tổng quát hoá các kết quả cụ thể vào trong một chương trình mềm dẻo thông minh.

- Vào những năm 70, máy tính với bộ nhớ lớn và tốc độ tính toán nhanh nhưng các phương pháp tiếp cận trí tuệ nhân tạo cũ vẫn thất bại (do sự bùng nổ tổ hợp trong quá trình tìm kiếm lời giải các bài toán đặt ra).

- Vào cuối những năm 70 một vài kết quả như xử lý ngôn ngữ tự nhiên, biểu diễn tri thức và giải quyết vấn đề. Những kết quả đó đã tạo điều kiện cho sản phẩm thương mại đầu tiên của trí tuệ nhân tạo ra đời đó là hệ chuyên gia, được đem áp dụng trong các lĩnh vực khác nhau (hệ chuyên gia là một phần mềm máy tính chứa các thông tin và tri thức về một lĩnh vực cụ thể nào đó, có khả năng giải quyết những yêu cầu của người sử dụng trong một mức độ nào đó, ở một trình độ như một chuyên gia con người có kinh nghiệm khá lâu năm).

- Một sự kiện quan trọng vào những năm 70 là sự ra đời ngôn ngữ Prolog, tương tự LISP nhưng nó có cơ sở dữ liệu đi kèm.

- Vào những năm 80, thị trường các sản phẩm dân dụng đã có khá nhiều sản phẩm ở trình đô cao như: máy giặt, máy ảnh,... sử dụng trí tuệ nhân tạo. Các hệ thống nhận dạng và xử lý ảnh, tiếng nói.

- Những năm 90, các nghiên cứu nhằm vào cài đặt thành phần thông minh trong các hệ thống thông tin, gọi chung là cài đặt trí tuệ nhân tạo, làm rõ hơn các ngành của khoa học trí tuệ nhân tạo và tiến hành các nghiên cứu mới, đặc biệt là nghiên cứu về cơ chế suy lý, về trí tuệ nhân tạo phân tạo, về các mô hình tương tác.

PHẦN II. DEMO ỨNG DỤNG CHƯƠNG TRÌNH TÌM LỘ TRÌNH GOM RÁC TỐI ƯU Ở THÀNH PHỐ HỒ CHÍ MINH BẰNG NGÔN NGỮ JAVA

1. PHÂN TÍCH ĐỀ TÀI

1.1. Phân tích yêu cầu

- Hiển thị toàn view Thành phố Hồ Chí Minh có tất cả 24 quận huyện, bao gồm 19 quận và 5 huyện.
- Mỗi quận, huyện sẽ có một điểm thu gom rác cố định. Vị trí xuất phát sẽ ở trung tâm lớn bất kỳ.
- Tiến hành tìm đường đi ngắn nhất dựa trên chi phí cho trước sau đó nối các điểm đã tìm được hướng đi tối ưu nhất.
- Lưu lại những điểm đã đi qua và tiếp tục tìm hướng đi tiếp theo dựa trên những điểm đã lưu trước đó.
- Cập nhật và lưu lại toàn bộ quá trình di chuyển giữa các điểm gom rác trên toàn thành phố với chi phí thấp nhất.
- Hiển thị bảng thống kê thứ tự các điểm gom rác và chi phí đạt được trong suốt quá trình gom rác ở Thành phố Hồ Chí Minh.

1.2. Yêu cầu chức năng

› Xuất hiện bản đồ view Thành phố Hồ Chí Minh.

› Hiển thị được địa điểm gom rác ở mỗi quận.

› Tìm được lộ trình tối ưu theo tiêu chí: Ngắn nhất, chi phí thấp nhất mà lộ trình đem lại.

› Cho phép thay đổi thông tin lộ trình phù hợp nhu cầu cá nhân.

› Lưu lại những lộ trình sau khi tìm được để tiện cho những lần sau.

› Lưu lại kết quả sau khi thu gom rác cung cấp cho việc thống kê kết quả.

› Thống kê kết quả: hiển thị được bảng thông tin các vị trí đã kết thúc thu gom rác và chi phí đạt được.

2. THIẾT KẾ

2.1. Đề xuất sử dụng thuật giải

Sử dụng thuật giải GTS:

Đây là thuật giải tìm chu trình có trọng số nhỏ nhất trong một đơn đồ thị có hướng có trọng số. Thuật toán tham lam cho bài toán là chọn điểm có chi phí nhỏ nhất tính từ điểm hiện thời đến các điểm chưa qua.

2.2. Cách thức giải quyết bài toán

Một xe rác muốn gom rác n quận T1,.., Tn . Xuất phát từ một quận nào đó, xe gom rác muốn đi qua tất cả các quận còn lại, mỗi quận đi qua đúng 1 lần rồi quay trở lại quận xuất phát.

- Thuật giải GTS1

❖ Đặt vấn đề

Xây dựng một lịch trình gom rác có chi phí Cost tối thiểu cho bài toán trong trường hợp phải qua n quận với ma trận chi phí C và bắt đầu tại một đỉnh U nào đó.

❖ Thuật giải

Bước 1: {Khởi đầu}

- Đặt Tour:={};

- Cost:=0;

- V:=U; {V là đỉnh hiện tại đang làm việc}

Bước 2: {Thăm tất cả các quận}

- For k:=1 To n Do qua bước 3;

Bước 3: {Chọn cung kế tiếp}

- Đặt (V,W) là cung có chi phí nhỏ nhất tính từ V đến các đỉnh W chưa dùng:

- Tour := Tour + {(V,W)};

- Cost := Cost + Cost{(V,W)};

- Nhãn W được sử dụng

- Đặt V := W; {Gán để xét bước kế tiếp}

Bước 4: {Chuyển đi hoàn thành}

- Đặt Tour := Tour + {(V,U)};

- Cost := Cost + {(V,U)};

- Dừng.

- Thuật giải GTS2

❖ Đặt vấn đề

Tạo ra lịch trình từ p quận xuất phát riêng biệt.

Tìm chu trình của xe gom rác qua n quận (1<p<n) và p chu trình được tạo ra và chỉ chu trình tốt nhất trong p chu trình được giữ lại mà thôi (Thuật giải này đòi hỏi phải nhập n, p và C).

❖ Thuật giải

Bước 1: {Khởi đầu}

- k :=0; {Đếm số quận đi qua}

- Best :={}; {Ghi nhớ chu trình tốt nhất tìm thấy có chi phí là Cost}

- Cost := ∞;

Bước 2: {Bắt đầu chu trình mới}

- Chuyển qua bước 3 khi k< p, ngược lại dừng.

Bước 3: {Tạo chu trình mới}

- k: = k + 1;

- Call (GTS1(Vk): Trả về một chu trình T(k) ứng với chi phí C(k).

Bước 4: {Cập nhật chu trình tốt nhất}

- Nếu C(k) < Cost thì

- Best := T(k);

- Cost := C(k);

DANH MỤC CÁC KÝ HIỆU TRONG CODE

V	Điểm xuất phát
T	Biến đếm
N	Số cột
M	Số dòng
U	Biến gán điểm bắt đầu
W	Biến gán điểm đến
D	Khoảng cách giữa 2 điểm
Tour	Các quận đã đi qua
X	Tọa độ x
Y	Tọa độ y
Flag	Cờ

3. THỰC HIỆN

a. Tạo tiêu đề tìm đường đi ngắn nhất cho thanh cửa sổ

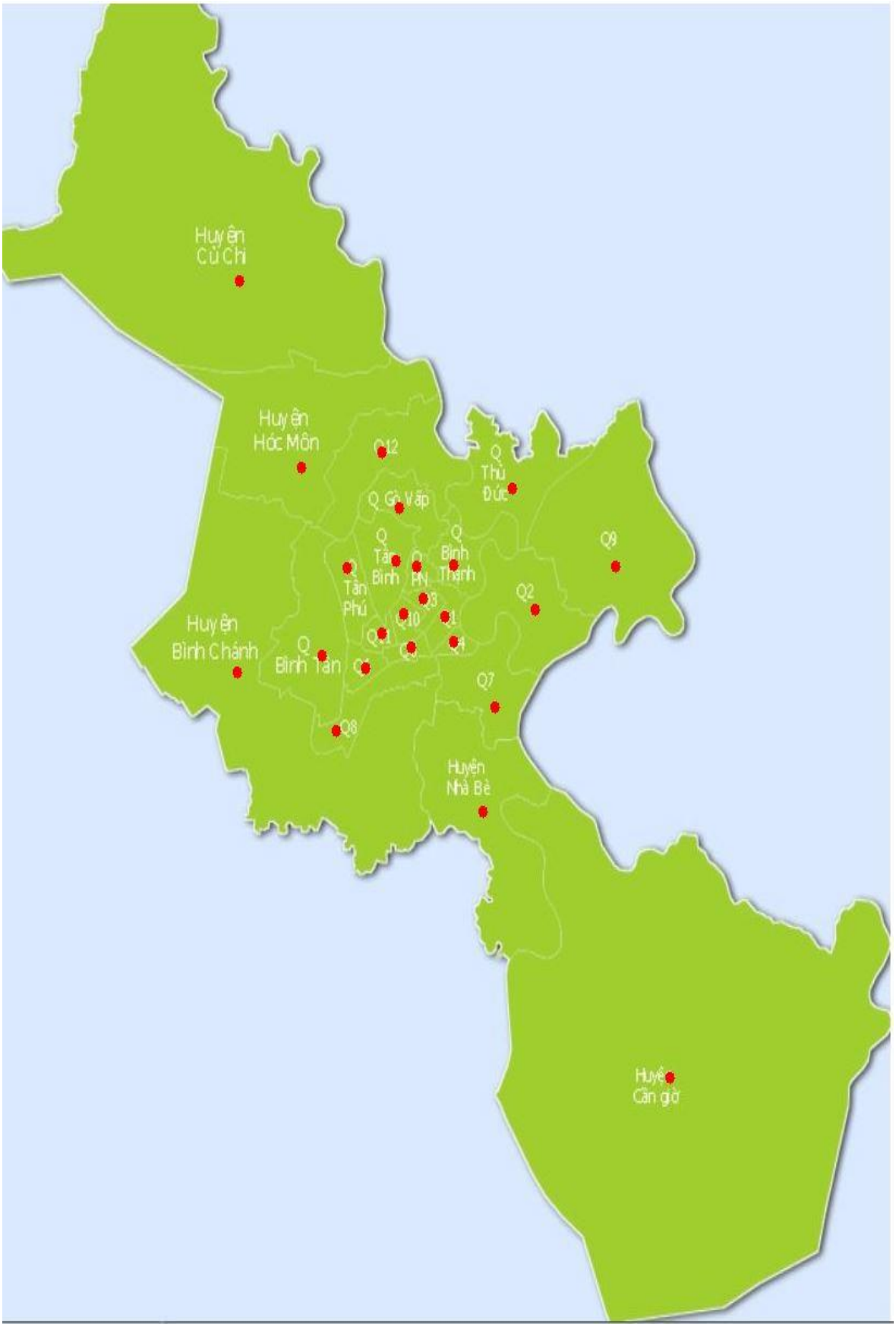
```
33  public class TimDuongDiNganNhat extends javax.swing.JFrame {
34
35      private Image mshi = new ImageIcon("2.jpg").getImage();
36      Dimension d;
37      Connection connection = null;
38      Statement sta = null;
39      ResultSet res = null;
40      DefaultTableModel dsQuanHuyen;
41
42      public TimDuongDiNganNhat() {
43          initComponents();
44          this.setTitle("Tìm đường đi ngắn nhất để dọn rác (GTS)");
45          d = new Dimension();
46          d.width = mshi.getWidth(null) + 250;
47          d.height = mshi.getHeight(null);
48          this.setSize(d);
49          this.setLocationRelativeTo(null);
50          ketNoiCSDL();
51      }
```

b. Tạo kết nối cơ sở dữ liệu

```
53  private void ketNoiCSDL() {
54      String strSever = "DESKTOP-GIU8V36\\SQLEXPRESS";
55      String strDatabase = "QuanHuyen";
56      String strUserName = "sa";
57      String strPassword = "123";
58      try {
59          Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
60      } catch (ClassNotFoundException ex) {
61          Logger.getLogger(TimDuongDiNganNhat.class.getName()).log(Level.SEVERE, null, ex);
62      }
63      String connectionURL = "jdbc:sqlserver://" + strSever
64              + ":1433; databaseName =" + strDatabase
65              + ";user=" + strUserName + ";password=" + strPassword;
66      try {
67          connection = DriverManager.getConnection(connectionURL);
68          sta = connection.createStatement();
69      } catch (SQLException ex) {
70          Logger.getLogger(TimDuongDiNganNhat.class.getName()).log(Level.SEVERE, null, ex);
71      }
72      if (connection != null) {
73          System.out.println("Ket noi thanh cong den csdl");
74      } else {
75          System.out.println("Loi ket noi, vui long kiem tra lai");
76      }
77  }
78
```

c. Vẽ bản đồ và xác định vị trí của các quận và huyện

```
80      public void paint(Graphics g) {
81          setForeground(Color.RED);
82          g.drawImage(mshi, 0, 0, pnBanDo);
83          DSCacQuan quan;
84          ArrayList<DSCacQuan> ds = new ArrayList<>();
85          dsQuanHuyen = new DefaultTableModel();
86          tbQuanHuyen.setModel(dsQuanHuyen);
87          String[] a = {"Mã quận", "Tên quận/huyện", "Tọa độ x", "Tọa độ y"};
88          dsQuanHuyen.setColumnIdentifiers(a);
89          String sqlSelect = "SELECT MaQuan,TenQuan,ToaDoX,ToaDoY FROM QuanHuyen";
90          try {
91              sta = connection.createStatement();
92              res = sta.executeQuery(sqlSelect);
93              while (res.next()) {
94                  String ma = res.getString("MaQuan");
95                  String tenQuan = res.getString("TenQuan");
96                  int x = res.getInt("ToaDoX");
97                  int y = res.getInt("ToaDoY");
98                  Vector<Object> vec = new Vector<Object>();
99                  vec.add(ma);
100                 vec.add(tenQuan);
101                 vec.add(x);
102                 vec.add(y);
103                 dsQuanHuyen.addRow(vec);
104                 quan = new DSCacQuan(tenQuan, new Diem2D(x, y));
105                 ds.add(quan);
106             }
107             Graphics2D g2d = (Graphics2D) g;
108             BasicStroke bs2 = new BasicStroke(9, BasicStroke.CAP_ROUND,
109                     BasicStroke.JOIN_BEVEL);
110             g2d.setStroke(bs2);
111             g2d.setColor(Color.red);
112             ds.forEach((dsq) -> {
113                 int x = dsq.getToaDo().getX();
114                 int y = dsq.getToaDo().getY();
115                 g2d.drawLine(x, y, x, y);
116             });
117         } catch (SQLException ex) {
118             Logger.getLogger(TimDuongDiNganNhat.class.getName()).log(Level.SEVERE, null, ex);
119         }
    }
```


Huyện
Củ Chi
Huyện
Hóc Môn
Q12
Q
Thủ
Đức
Q. Gò Vấp
Q
Tân
Bình
Q
PN
Q
Bình
Thạnh
Q9
Tân
Phú
Q2
Q10
Huyện
Bình Chánh
Q
Bình Tân
Q7
Q8
Huyện
Nhà Bè
Huyện
Cần giờ

d. Tìm đường đi ngắn nhất của từ điểm cho sẵn (v)

```
280    public double GTS1(int v) {
281        double kq = 0;
282        double maTran[][];
283        int t = 0;
           int n = 0, m = 0, u = 0, w = 0;
           double d = 0, min = 99999;
286        n = tbQuanHuyen.getRowCount();
287        m = n;
288        int flag[] = new int[n];
289        maTran = new double[n][m];
290
291        for (int i = 0; i < n; i++) {
292            for (int j = 0; j < m; j++) {
293                if (i == j) {
294                    maTran[i][j] = 999999;
295                } else {
296                    int x1 = (int) tbQuanHuyen.getValueAt(i, 2);
297                    int y1 = (int) tbQuanHuyen.getValueAt(i, 3);
298                    int x2 = (int) tbQuanHuyen.getValueAt(j, 2);
299                    int y2 = (int) tbQuanHuyen.getValueAt(j, 3);
300                    d = Math.sqrt(Math.pow((x2 - x1), 2) + Math.pow((y2 - y1), 2));
301                    double roundOff = Math.floor(d * 100) / 100;
302                    maTran[i][j] = roundOff;
303                }
304            }
305        }
```

```
307        for (int i = 0; i < n; i++) {
308            flag[i] = -1;
309        }
310        u = v;
311        flag[u] = 1;
312        do {
313            for (int j = 0; j < m; j++) {
314                if (maTran[u][w] > maTran[u][j] && flag[j] != 1) {
315                    min = maTran[u][j];
316                    w = j;
317                }
318            }
319            kq += min;
320            t += 1;
321            u = w;
322            flag[w] = 1;
323        } while (t < n);
324        kq += maTran[v][u];
325        return kq;
326    }
```

- Cờ kiểm tra điểm đã đi qua

```
288        int flag[] = new int[n];
```

- Khoảng cách giữ 2 điểm đưa vào ma trận đường đi và lấy 2 số thập phân sau dấu phẩy

```
301                    double roundOff = Math.floor(d * 100) / 100;
302                    maTran[i][j] = roundOff;
```

- Gán cờ cho các điểm -1 (chưa đi qua)

```
308                 flag[i] = -1;
```

- Gán u = vị trí bắt đầu (v) và vị trí đã đi qua gán cờ 1 (đã đi qua)

```
310             u = v;
311             flag[u] = 1;
```

- Kiểm tra điểm đã đi qua chưa và có phải là ngắn nhất không
- Gán đường đi ngắn nhất
- Gán w là đỉnh có thể đi qua

```
312         do {
313             for (int j = 0; j < m; j++) {
314                 if (maTran[u][w] > maTran[u][j] && flag[j] != 1) {
315                     min = maTran[u][j];
316                     w = j;
```

- Cộng đường đi vào kết quả
- Xét lại điểm xuất phát (u=w)
- Vị trí đã đi qua gán cờ 1(flag[w]=1)
- Trả về đường đi tốt nhất

```
319             kq += min;
320             t += 1;
321             u = w;
322             flag[w] = 1;
323         } while (t < n);
324         kq += maTran[v][u];
325         return kq;
326     }
```

e. Xử lí button GTS1

- Chọn điểm bắt đầu

```
416     private void btGTS1ActionPerformed(java.awt.event.ActionEvent evt) {
417         // TODO add your handling code here:
418         if (tfBatDau.getText().equals("")) {
419             JOptionPane.showMessageDialog(null, "Hãy nhập điểm bắt đầu!");
420         } else {
421             Graphics g = getGraphics();
422             String tour = "";
423             double kq = 0;
424             double maTran[][];
425             int t = 0;
                int n = 0, m = 0, u = 0, w = 0;
                double d = 0, min = 99999;
428             n = tbQuanHuyen.getRowCount();
429             m = n;
430             int flag[] = new int[n];
431             maTran = new double[n][m];
432             for (int i = 0; i < n; i++) {
433                 for (int j = 0; j < m; j++) {
434                     if (i == j) {
435                         maTran[i][j] = 999999;
436                     } else {
437                         int x1 = (int) tbQuanHuyen.getValueAt(i, 2);
438                         int y1 = (int) tbQuanHuyen.getValueAt(i, 3);
439                         int x2 = (int) tbQuanHuyen.getValueAt(j, 2);
440                         int y2 = (int) tbQuanHuyen.getValueAt(j, 3);
441                         d = Math.sqrt(Math.pow((x2 - x1), 2) + Math.pow((y2 - y1), 2));
442                         double roundOff = Math.floor(d * 100) / 100;
443                         maTran[i][j] = roundOff;
444                     }
```

```
447                 for (int i = 0; i < n; i++) {
448                     flag[i] = -1;
449                 }
450                 for (int c = 0; c < n; c++) {
451                     if (tbQuanHuyen.getValueAt(c, 1).equals(tfBatDau.getText())) {
452                         u = c;
453                     }
454                 }
455                 flag[u] = 1;
456                 do {
457                     for (int j = 0; j < m; j++) {
458                         if (maTran[u][w] > maTran[u][j] && flag[j] != 1) {
459                             min = maTran[u][j];
460                             w = j;
461                         }
462                     }
463                     try {
464                         TimeUnit.SECONDS.sleep(1);
465                         int x1 = (int) tbQuanHuyen.getValueAt(u, 2);
466                         int y1 = (int) tbQuanHuyen.getValueAt(u, 3);
467                         int x2 = (int) tbQuanHuyen.getValueAt(w, 2);
468                         int y2 = (int) tbQuanHuyen.getValueAt(w, 3);
469                         Graphics2D g2d = (Graphics2D) g;
470                         BasicStroke bs2 = new BasicStroke(3, BasicStroke.CAP_ROUND,
471                                 BasicStroke.JOIN_BEVEL);
472                         g2d.setStroke(bs2);
473                         g2d.setColor(Color.red);
474                         g2d.drawLine(x1, y1, x2, y2);
475                     } catch (InterruptedException ex) {
476                         Logger.getLogger(TimDuongDiNganNhat.class.getName()).log(Level.SEVERE, null, ex);
477                     }
```

```
474                         g2d.drawLine(x1, y1, x2, y2);
475                     } catch (InterruptedException ex) {
476                         Logger.getLogger(TimDuongDiNganNhat.class.getName()).log(Level.SEVERE, null, ex);
477                     }
478                     tour = tour + tbQuanHuyen.getValueAt(u, 1).toString() + "\n";
479                     kq += min;
480                     t += 1;
481                     u = w;
482                     flag[w] = 1;
483                 } while (t < n);
484                 for (int c = 0; c < n; c++) {
485                     if (tbQuanHuyen.getValueAt(c, 1).equals(tfBatDau.getText())) {
486                         u = c;
487                     }
488                 }
489                 int x1 = (int) tbQuanHuyen.getValueAt(u, 2);
490                 int y1 = (int) tbQuanHuyen.getValueAt(u, 3);
491                 int x2 = (int) tbQuanHuyen.getValueAt(w, 2);
492                 int y2 = (int) tbQuanHuyen.getValueAt(w, 3);
493                 Graphics2D g2d = (Graphics2D) g;
494                 BasicStroke bs2 = new BasicStroke(3, BasicStroke.CAP_ROUND,
495                         BasicStroke.JOIN_BEVEL);
496                 g2d.setStroke(bs2);
497                 g2d.setColor(Color.red);
498                 g2d.drawLine(x1, y1, x2, y2);
499                 tour = tour + tbQuanHuyen.getValueAt(u, 1).toString();
500                 double roundOff = Math.floor(kq * 100) / 100;
501                 taKetQua.setText(tour);
502                 lbKetQua.setText(String.valueOf(roundOff) + " km");
503             }
504         }
```

♣ Lấy tọa độ x,y của 2 điểm có đường đi ngắn nhất

```
464                     TimeUnit.SECONDS.sleep(1);
465                     int x1 = (int) tbQuanHuyen.getValueAt(u, 2);
466                     int y1 = (int) tbQuanHuyen.getValueAt(u, 3);
467                     int x2 = (int) tbQuanHuyen.getValueAt(w, 2);
468                     int y2 = (int) tbQuanHuyen.getValueAt(w, 3);
469                     Graphics2D g2d = (Graphics2D) g;
```

♪ Vẽ đường đi của các điểm

```
474                 g2d.drawLine(x1, y1, x2, y2);/
```

☻ Danh sách đường đi

```
478                 tour = tour + tbQuanHuyen.getValueAt(u, 1).toString() + "\n";/
479                 kq += min;
480                 t += 1;
481                 u = w;
482                 flag[w] = 1;
483             } while (t < n);
484             for (int c = 0; c < n; c++) {
485                 if (tbQuanHuyen.getValueAt(c, 1).equals(tfBatDau.getText())) {
```

○ Lấy tọa độ điểm cuối đến điểm bắt đầu

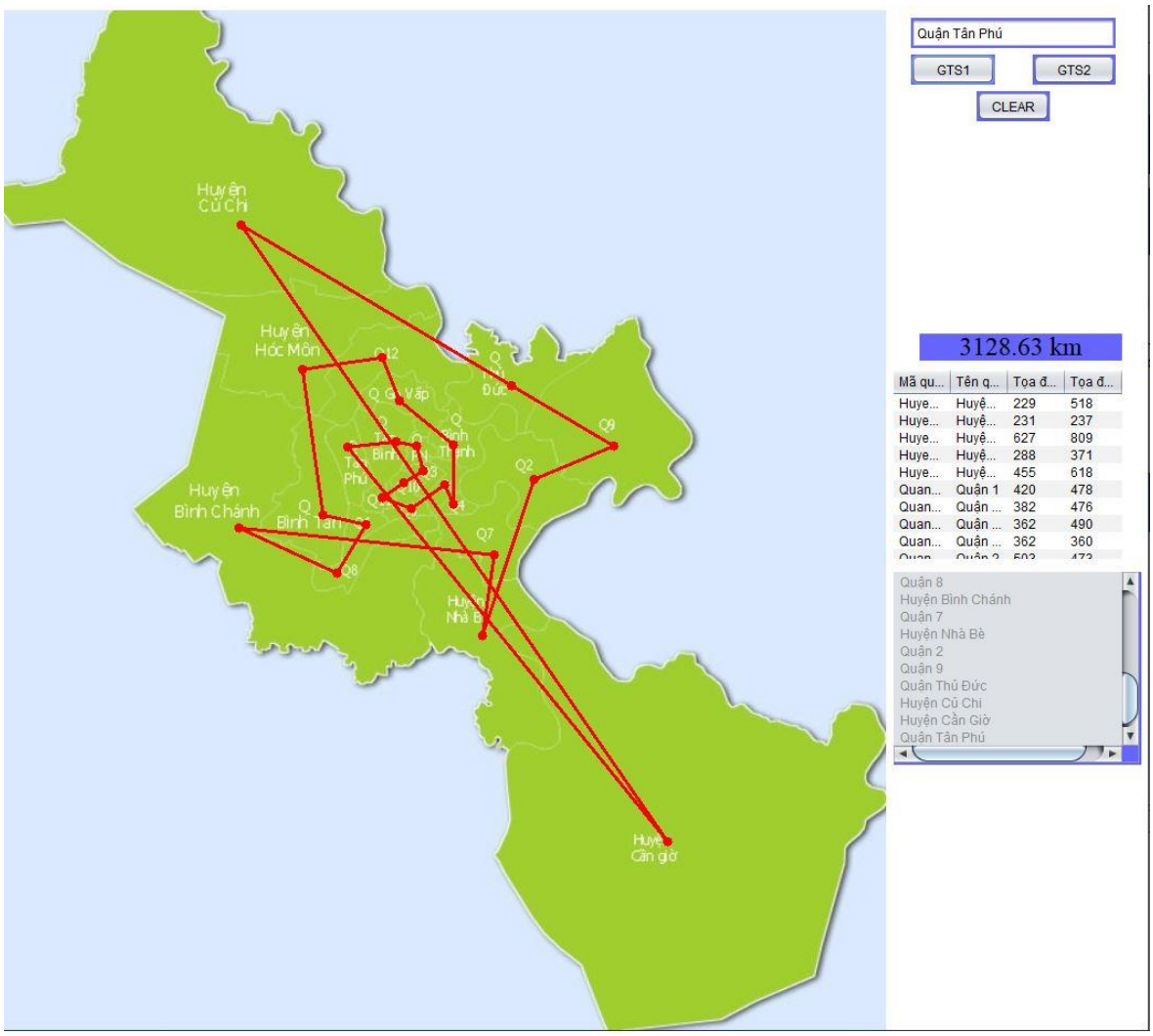
```
489                 int x1 = (int) tbQuanHuyen.getValueAt(u, 2);
490                 int y1 = (int) tbQuanHuyen.getValueAt(u, 3);
491                 int x2 = (int) tbQuanHuyen.getValueAt(w, 2);
492                 int y2 = (int) tbQuanHuyen.getValueAt(w, 3);
493                 Graphics2D g2d = (Graphics2D) g;
```

☼ Vẽ đường đi

☼ Xuất kết quả sau khi tối ưu

☼ Xuất tổng chi phí

```
498                 g2d.drawLine(x1, y1, x2, y2);
499                 tour = tour + tbQuanHuyen.getValueAt(u, 1).toString();
500                 double roundOff = Math.floor(kq * 100) / 100;
501                 taKetQua.setText(tour);
502                 lbKetQua.setText(String.valueOf(roundOff) + " km");
503             }
504         }
```


Quận Tân Phú
GTS1
GTS2
CLEAR
3128.63 km
Mã qu... Tên q... Tọa đ... Tọa đ...
Huye... Huyệ... 229 518
Huye... Huyệ... 231 237
Huye... Huyệ... 627 809
Huye... Huyệ... 288 371
Huye... Huyệ... 455 618
Quan... Quận 1 420 478
Quan... Quận ... 382 476
Quan... Quận ... 362 490
Quan... Quận ... 362 360
Quận 8
Huyện Bình Chánh
Quận 7
Huyện Nhà Bè
Quận 2
Quận 9
Quận Thủ Đức
Huyện Củ Chi
Huyện Cần Giờ
Quận Tân Phú
Huyện Củ Chi
Huyện Hóc Môn
Huyện Bình Chánh
Huyện Cần giờ
Q9
Q2
Q7
Q8

f. Xử lý button GTS2 (Tìm đường đi tốt nhất)

```
328  private void btGTS2ActionPerformed(java.awt.event.ActionEvent evt) {
329
330      Graphics g = getGraphics();
331      String tour = "";
332      double kq = 0;
333      double maTran[][];
334      int t = 0;
         int n = 0, m = 0, u = 0, w = 0;
         double d = 0, min = 99999, bestCost = 99999;
337      n = tbQuanHuyen.getRowCount();
338      m = n;
339      int flag[] = new int[n];
340      maTran = new double[n][m];
341      int v = 0;
342      for (int i = 0; i < n; i++) {
343          if (bestCost > GTS1(i)) {
344              bestCost = GTS1(i);
345              v = i;
346          }
347      }
348      for (int i = 0; i < n; i++) {
349          for (int j = 0; j < m; j++) {
350              if (i == j) {
351                  maTran[i][j] = 999999;
352              } else {
353                  int x1 = (int) tbQuanHuyen.getValueAt(i, 2);
354                  int y1 = (int) tbQuanHuyen.getValueAt(i, 3);
355                  int x2 = (int) tbQuanHuyen.getValueAt(j, 2);
356                  int y2 = (int) tbQuanHuyen.getValueAt(j, 3);
357                  d = Math.sqrt(Math.pow((x2 - x1), 2) + Math.pow((y2 - y1), 2));
358                  double roundOff = Math.floor(d * 100) / 100;
359                  maTran[i][j] = roundOff;
360              }
361          }
362      }
363      for (int i = 0; i < n; i++) {
364          flag[i] = -1;
365      }
366      u = v;
367      flag[u] = 1;
368      do {
369          for (int j = 0; j < m; j++) {
370              if (maTran[u][w] > maTran[u][j] && flag[j] != 1) {
371                  min = maTran[u][j];
372                  w = j;
373              }
374          }
375          try {
376              TimeUnit.SECONDS.sleep(1);
377              int x1 = (int) tbQuanHuyen.getValueAt(u, 2);
378              int y1 = (int) tbQuanHuyen.getValueAt(u, 3);
379              int x2 = (int) tbQuanHuyen.getValueAt(w, 2);
380              int y2 = (int) tbQuanHuyen.getValueAt(w, 3);
381              Graphics2D g2d = (Graphics2D) g;
382              BasicStroke bs2 = new BasicStroke(3, BasicStroke.CAP_ROUND,
383                      BasicStroke.JOIN_BEVEL);
384              g2d.setStroke(bs2);
385              g2d.setColor(Color.red);
386              g2d.drawLine(x1, y1, x2, y2);
```

```
387             } catch (InterruptedException ex) {
388                 Logger.getLogger(TimDuongDiNganNhat.class.getName()).log(Level.SEVERE, null, ex);
389             }
390
391             tour = tour + tbQuanHuyen.getValueAt(u, 1).toString() + "\n";
392             kq += min;
393             t += 1;
394             u = w;
395             flag[w] = 1;
396         } while (t < n);
397
398         u = v;
399         int x1 = (int) tbQuanHuyen.getValueAt(u, 2);
400         int y1 = (int) tbQuanHuyen.getValueAt(u, 3);
401         int x2 = (int) tbQuanHuyen.getValueAt(w, 2);
402         int y2 = (int) tbQuanHuyen.getValueAt(w, 3);
403         Graphics2D g2d = (Graphics2D) g;
404         BasicStroke bs2 = new BasicStroke(3, BasicStroke.CAP_ROUND,
405                 BasicStroke.JOIN_BEVEL);
406         g2d.setStroke(bs2);
407         g2d.setColor(Color.red);
408         g2d.drawLine(x1, y1, x2, y2);
409         tour = tour + tbQuanHuyen.getValueAt(u, 1).toString();
410         kq += maTran[w][u];
411         double roundOff = Math.floor(kq * 100) / 100;
412         taKetQua.setText(tour);
413         lbKetQua.setText(String.valueOf(roundOff) + " km");
414     }
```

- ✧ Thời gian vẽ đường đi
- ✧ Lấy tọa độ x,y của 2 điểm có đường đi ngắn nhất
- ✧ Thư viện vẽ

```
376                 TimeUnit.SECONDS.sleep(1);
377                 int x1 = (int) tbQuanHuyen.getValueAt(u, 2);
378                 int y1 = (int) tbQuanHuyen.getValueAt(u, 3);
379                 int x2 = (int) tbQuanHuyen.getValueAt(w, 2);
380                 int y2 = (int) tbQuanHuyen.getValueAt(w, 3);
381                 Graphics2D g2d = (Graphics2D) g;
382                 BasicStroke bs2 = new BasicStroke(3, BasicStroke.CAP_ROUND,
383                         BasicStroke.JOIN_BEVEL);
```

- ➢ Chỉnh đậm nhạt cho đường đi
- ➢ Chỉnh màu
- ➢ Vẽ đường đi của các điểm

```
384                 g2d.setStroke(bs2);
385                 g2d.setColor(Color.red);
386                 g2d.drawLine(x1, y1, x2, y2);
```

- ✓ Danh sách đường đi

```
391             tour = tour + tbQuanHuyen.getValueAt(u, 1).toString() + "\n";
392             kq += min;
393             t += 1;
394             u = w;
395             flag[w] = 1;
```

❖ Vẽ lại đường đi từ điểm kết thúc về lại điểm bắt đầu

```
398             u = v;
399             int x1 = (int) tbQuanHuyen.getValueAt(u, 2);
400             int y1 = (int) tbQuanHuyen.getValueAt(u, 3);
401             int x2 = (int) tbQuanHuyen.getValueAt(w, 2);
402             int y2 = (int) tbQuanHuyen.getValueAt(w, 3);
403             Graphics2D g2d = (Graphics2D) g;
```

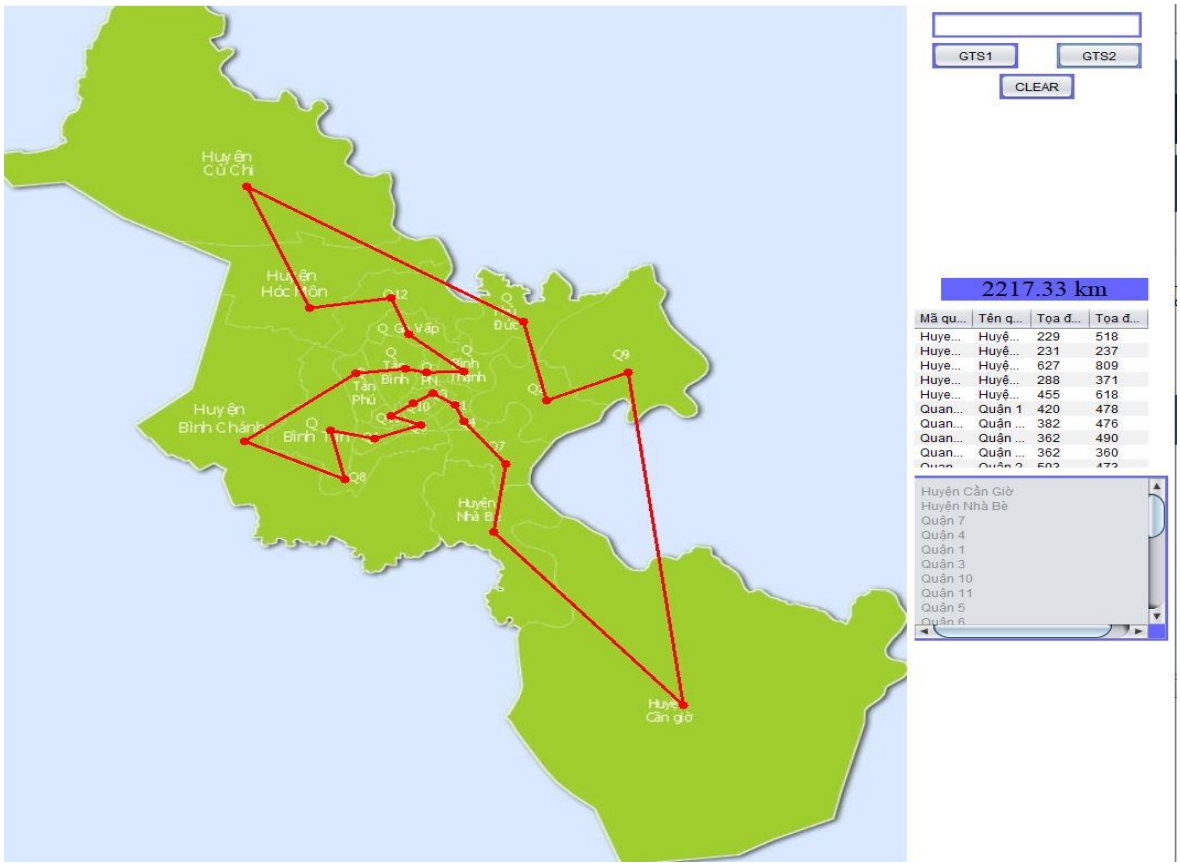
♥ Vẽ đường đi

```
408             g2d.drawLine(x1, y1, x2, y2);
409             tour = tour + tbQuanHuyen.getValueAt(u, 1).toString();
410             kq += maTran[w][u];
411             double roundOff = Math.floor(kq * 100) / 100;
```

♦ Xuất kết quả sau khi tối ưu

♦ Xuất tổng chi phí

```
412             taKetQua.setText(tour);
413             lbKetQua.setText(String.valueOf(roundOff) + " km");
```



g. Xử lý button Xóa

```
private void btXoaActionPerformed(java.awt.event.ActionEvent evt) {
    new TimDuongDiNganNhat().setVisible(true);
}
```

4. KẾT LUẬN VÀ ĐỊNH HƯỚNG PHÁT TRIỂN

4.1. Kết luận

Qua đề tài "Tìm lộ trình xe thu gom rác tối ưu ở Thành phố Hồ Chí Minh", đồ án đã đạt được những kết quả sau:

- Tìm hiểu những kiến thức cơ bản thuật giải GTS.
- Tìm hiểu về bài toán tìm đường đi ngắn nhất.
- Xây dựng được chương trình minh họa tìm lộ trình gom rác tối ưu ở TPHCM qua thuật toán GTS tìm đường đi ngắn nhất.

4.2. Hướng phát triển

Đồ án đã xây dựng được chương trình minh họa thuật toán tìm đường đi ngắn nhất và xây dựng chương trình minh họa cho lộ trình gom rác tối ưu ở TPHCM, cụ thể đó là thuật toán GTS, được ứng dụng vào nhiều lĩnh vực trong thực tế như trò chơi trí tuệ, tìm lộ trình gom rác,…. Tuy nhiên không thể áp dụng tất cả các thuật toán tìm đường đi ngắn nhất để xây dựng lộ trình do mỗi thuật toán còn phức tạp. Hướng phát triển của đề tài trong thời gian tới là thực hiện thêm một số thuật toán tìm đường đi ngắn nhất để xây dựng hướng đi tối ưu trong quá trình gom rác, cải tiến thuật toán nhằm giảm thời gian xử lý, thiết kế thêm một số chức năng cho chương trình được hoàn thiện hơn, thiết kế chương trình để phát triển trên môi trường mạng. Tìm hiểu thêm các ngôn ngữ khác để viết chương trình và có thể chạy trên nhiều nền tảng khác nhau. Thiết kế giao diện thân thiện với người sử dụng hơn, hoàn thành các phần còn thiếu sót và hạn chế. Chúng em sẽ tiếp tục phát triển đề tài này bằng cách nghiên cứu thêm ngôn ngữ khác để kết hợp với các công cụ của ngôn ngữ lập trình Java nhằm làm cho giao diện đẹp hơn, thuận tiện hơn với người sử dụng và sản phẩm sẽ được đóng gói trước khi tới tay của người sử dụng. Xin chân thành cảm ơn Thầy đã đọc bài báo cáo của nhóm em, chúng em rất mong nhận được sự góp ý của Thầy về đồ án này, nhận xét góp ý của Thầy sẽ giúp cho nhóm em có được những kinh nghiệm quí báu để có thể thực hiện những đồ án sau được hoàn thiện hơn.

TÀI LIỆU THAM KHẢO

- Applegate, D. L.; Bixby, R. M.; Chvátal, V.; Cook, W. J. (2006), The Traveling Salesman Problem
- Allender, Eric; Bürgisser, Peter; Kjeldgaard-Pedersen, Johan; Mitersen, Peter Bro (2007), "On the Complexity of Numerical Analysis"
- Arora, Sanjeev (1998), "Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems",
- Beardwood, J.; Halton, J.H.; Hammersley, J.M. (1959), "The Shortest Path Through Many Points", Proceedings of the Cambridge Philosophical Society,
- Bellman, R. (1960), "Combinatorial Processes and Dynamic Programming", in Bellman, R.; Hall, M. Jr. (eds.), Combinatorial Analysis, Proceedings of Symposia in Applied Mathematics 10, American Mathematical Society, pp. 217–249.

Hết