实验报告

——project4

组长：王迎旭（16340226）
组员：王友坤 （16340228）汤万鹏（16340205）
班级：16 级软件工程教务三班
日期：2017.12.16

目录

第一部分

问题描述

【题目要求】

（1）从中山大学东校区的平面图中选取有代表性的景点（10-15 个），抽象成一个无向带权图。以图中定点表示景点，边上的权值表示两地之间距离。

（2）本程序的目的是为用户提供路径咨询。根据用户指定的始点和终点输出相应路径，或者根据用户指定的景点输出景点的信息。

【需求分析】

（1）渲染中山大学东校区抽象而成的无向带权图，显示景点和边权

（2）选中景点，提供相关信息的查询：代号、简介

（3）查询模式：可让用户自由、方便地选择两个景点，清晰地标注起点和终点，输出两地的具体长度，以及沿途中所要经过的景点

（4）以区别汽车线路和步行线路的方式，查询最短的简单路径，提供文字、总路径长度，以及步行或者汽车所要行使的具体长度等全方位的信息，使用户一目了然

第二部分

编译细节的介绍

2.1 运行环境

①Windows10 （部分特殊字符无法显示）
②Fedora 虚拟机或者 Ubuntu 虚拟机**（推荐）**

2.2编程语言

c++

2.3开发工具

（1）Windows ，linux 环境下的 g++编译器与 Sublime Text3 编辑器

（2）Dev c++（注:由于本地的编写实在 Sublime 中完成，所以在 c++中编译时候会产生注释乱码情况）

2.4编码的规范

Sub 中编辑时候，已经严格按照首行缩进的方式进行编译。但美中不足，在代码进行复制粘贴时候有可能导致代码错位的问题。

第三部分

分析与设计

3.1 数据结构

认真研读需求，决定用邻接链表来存储图定点的信息，使用邻接矩阵存储顶点与顶点之间的长度。

对图的每个顶点建立一个单链表，存储该顶点所有邻接顶点及其相关信息。每一个单链表设一个表头结点。链表中的结点称为表结点，每个结点由三个域组成。其中邻接点域指示与顶点 Vi 邻接的顶点在图中的位置(顶点编号)，链域指向下一个与顶点 Vi 邻接的表结点，以及这个结点的具体信息。同时每个链表设一个表头结点，由两个域组成。链域指向链表中的第一个结点，数据域存储顶点名或其他信息。在图的邻接链表表示中，所有顶点结点用一个向量以顺序结构形式存储，可以随机访问任意顶点的链表，该向量称为表头向量，向量的下标指示顶点的序号。用邻接链表存储图时，对无向图，其邻接链表是唯一的。

邻接矩阵则用来存储和边或弧相关的信息，如本次的路径长度（权值），以及是否可以通行汽车。

3.2 设计思路

前期考虑：

如果使用迪杰斯特拉算法和弗洛伊德算法实现最短路径的话，比较简单，同时也比较容易实现，但是会浪费很多无用的时间与空间（~~虽然测试样例很少，数据也比较好算~~）。恰好组内某位成员最近在准备 ACM 新手赛时候在竞赛书上看到一个可以求最短路径的新算法：**SPFA 算法**，所以组内制作这次 project 就尝试用了一次。

SPFA 算法个人在这里的使用与分析：

SPFA（Shortest Path Faster Algorithm）（队列优化）算法是求单源最短路径的一种算法，它在 Bellman-ford 算法的基础上加上一个队列优化，减少了冗余的松弛操作，是一种高效的最短路算法。具体思想是动态逼近：设立一个先进先出的队列用来保存待优化的结点，优化时每次取出队首结点 u，并且用 u 点当前的最短路径估计值对离开 u 点所指向的结点 v 进行松弛操作，如果 v 点的最短路径估计值有所调整，且 v 点不在当前的队列中，就将 v 点放入队尾。这样不断从队列中取出结点来进行松弛操作，直至队列空为止。

只要最短路径存在，上述 SPFA 算法必定能求出最小值。证明：每次将点放入队尾，都是经过松弛操作达到的。换言之，每次的优化将会有某个点 v 的最短路径估计值 d[v]变小。所以算法的执行会使 d 越来越小。由于我们假定图中不存在负权回路，所以每个结点都有最短路径值。因此，算法不会无限执行下去，随着 d 值的逐渐变小，直到到达最短路径值时，算法结束，这时的最短路径估计值就是对应结点的最短路径值。

整体的思路介绍：

1. **插入结点：**使用邻接表创建各个标志点，完成各类信息的输入，存储各个点名字以及介绍等信息

2. **插入结点之间的边：**先输入起点终点（在图中遍历起点终点，如果任何一个不存在就发出报错信息并返回），然后输入边的长度【使用邻接矩阵存储信息，由于是无向图，所以可以同时将图上（i,j）(j,i)两个点同时置数】，最后输入该边是否可以通行汽车（同样是用邻接矩阵存信息）

3. **删除结点：**删除该结点，并将该结点在邻接矩阵中的信息全部置为 0，并将该结点在邻接表中存储的信息全部删除

4. **删除边：**删除两个结点之间的边的信息，将邻接表里该边的长度置为 0，同时在临接表中将该边的信息删除（可能会导致图不连通）

5. **求最短路径：**使用 SPFA 算法求出最短路径（算法解释如上方）

6. **显示所有的顶点以及所有边的信息：**使用深度优先遍历，遍历每一条边以及每一个结点的具体信息就可以实现。

第四部分

代码展示

Vertex 类：

```
class Vertex{

    private:
        string Data;     //该结点内存储的信息
        int Number; //该结点的编号
        string Description;
        vector<pair<string, int> > Adjacency_Vertex;     //记录该节点相邻结点的信息
        static int Total_Number;     //记录一共有多少个结点

    public:
        friend class Undirected_Graph;

    public:
        Vertex(string = "NULL", string = "NULL");    //构造函数
        ~Vertex();  //析构函数
        void Set_Data(string);  //重新设置该结点的信息
        string Get_Data(void) const;     //返回该结点信息
        void Set_Description(string);    //重置结点具体信息
        string Get_Description(void) const; //返回具体信息
        int Get_Number(void) const; //返回该结点编号
        void Display_Information() const;    //输出该结点的详细信息
        bool Is_Adjacent(string) const; //判断参数内信息对应的结点是否与该结点相邻

};
```

Edge 类：

```
class Edge{

    private:
        int Weight; //边的权重，也是路径的长度
        string First_Vertex_Data, Second_Vertex_Data;    //记录该路径连接的两个结点的名字
        string Edge_Info;    //记录路径内存储的具体信息
        bool Car_Allowed;    //判断该路径是否允许车辆通过
        int Number; //该路径的编号
        static int Total_Number;     //创造的路径的总数

    public:
        friend class Undirected_Graph;

    public:
        Edge(int = 0, string = "NULL", string = "NULL", bool = true);    //构造函数
        ~Edge();     //析构函数
        void Set_Weight(int);    //重置路径的长度
        int Get_Weight(void) const; //返回路径的长度
        string Get_Edge_Info() const;    //返回路径的详细信息
        string Get_First_Vertex_Data(void) const;    //返回该路径连接的两个结点中第一个结点的名字
        string Get_Second_Vertex_Data(void) const;  //返回该路径连接的两个结点中第二个结点的名字
        void Display_Information() const;    //输出该路径的详细信息

};
```

邻接矩阵类：

```
class Adjacency_Matrix{

    private:
        int **Total_Matrix;
        int Size;

    public:
        friend class Undirected_Graph;

    public:
        Adjacency_Matrix(); //构造函数
        ~Adjacency_Matrix();     //析构函数
        void Display_Matrix() const;     //输出邻接矩阵
        void Destroy_Matrix();  //释放邻接矩阵所申请的内存空间

};
```

无向图类：

```
class Undirected_Graph{
    private:
        vector<Vertex*> Total_Vertex;    //记录该无向图中所有的结点
        vector<Edge*> Total_Edge;    //记录该无向图中所有的路径
        Adjacency_Matrix Matrix;     //邻接矩阵
    public:
        Undirected_Graph(); //无向图构造函数
        ~Undirected_Graph();    //无向图析构函数
        Vertex* Get_Vertex_From_Data(string) const; //根据名字返回结点的指针，参数为目标结点的名字
        Vertex* Get_Vertex_From_Number(int) const;  //根据编号返回结点的指针，参数为目标结点的编号
        Edge* Get_Edge_From_Info(string) const; //根据信息返回路径的指针，参数为目标路径的信息
        Edge* Get_Edge_From_Number(int) const;  //根据编号返回路径的指针，参数为目标路径的编号
        Edge* Get_Edge_From_Vertices(Vertex*, Vertex*) const;    //根据两个结点返回连接该两个结点的路径指针
        Vertex* Get_Nearest_Adjacency_Vertex(Vertex*) const;     //返回距离参数中结点最近的结点指针
        void Update_Adjacency_Matrix_Info();     //根据无向图内部结点与路径的变化更新邻接矩阵的信息
        bool Insert_Vertex(Vertex*);     //将参数结点指针插入图中，不允许同名结点存在
        bool Delete_Vertex_By_Data(string); //根据结点的名称删除图内的结点，参数为目标结点的名称
        bool Delete_Vertex_By_Number(int);  //根据结点的编号删除图内的结点，参数为目标结点的编号
        bool Insert_Edge(Edge*);     //将参数内的路径插入图中，不允许多条路径连接相同两个结点
        bool Delete_Edge_By_Info(string);    //根据路径的详细信息删除路径，参数为目标路径的信息
        bool Delete_Edge_By_Number(int);     ///根据路径的编号删除路径，参数为目标路径的编号
        void Display_Adjacency_Matrix() const;  //输出邻接矩阵
        void Display_All_Vertices() const;  //输出图内所有结点的详细信息
        void Display_All_Edges() const;      //输出图内所有路径的详细信息
        vector<Vertex*> Get_Shortest_Route(Vertex*, Vertex*, int&, bool) const; //寻找参数两个结点之间的最短路
};
```

功能类：

```
class Service{
    private:
        Service() {};
        const string password = "sapphire";
        Undirected_Graph My_Graph;
    public:
        ~Service() = default;
        static Service* get_instance(); //单例模式函数
        template <typename T>
        bool input_judge(T&);
        bool Password_Authentication() const;   //密码解锁函数
        void Ouput_Dash(int) const;
        void Output_Function() const;   //输出主菜单
        void Insert_Vertex();   //插入景点
        void Insert_Edge(); //插入路径
        void Delete_Vertex();   //删除景点
        void Delete_Edge(); //删除路径
        void Output_Adjacency_Matrix(); //输出邻接矩阵
        void Output_All_Vertices(); //输出所有景点的详细信息
        void Output_All_Edges();    //输出所有路径的详细信息
        void Shortest_Route();  //寻找两个景点之间的最短路径
        bool Exit_Judge_Function(); //判断是否终止程序
};
```

第五部分

测试程序

测试数据 A 组：

（1）主界面登陆

```
Undirected graph constructed.
Please input the password: sapphire
Correct password. Program entered successfully.
```

（输入密码完成后进入界面）

```
Input the number to execute the corresponding function below:
--------------------------------------------------
0                                  Clear The Sreen
--------------------------------------------------
1                                      Insert Spot
--------------------------------------------------
2                                      Insert Path
--------------------------------------------------
3                                      Delete Spot
--------------------------------------------------
4                                      Delete Path
--------------------------------------------------
5                             Show Adjacent Matrix
--------------------------------------------------
6                           Show Spots Information
--------------------------------------------------
7                           Show Paths Information
--------------------------------------------------
8                              Get Shortest Route
--------------------------------------------------
9                                Exit The Program
--------------------------------------------------
Please input the number:
```

（2）插入结点

```
        Input the name of the spot: Gym

        Input the description of the spot below:
The sports gym in the Sun Yat-Sen University

        No.1 Spot "Gym" is constructed successfully.

        Spot "Gym" is inserted successfully.
```

（3）插入边：

```
Input the length of the path: 10

Input the names of the two spots that the path connects below.

The name of the first spot: Gym

The name of the second spot: library

Is this path allowed to be passed by cars? (y/n): n

No.1 Path between "Gym" and "library" is constructed successfully.

The spot is inserted successfully.
```

（4）路径信息显示

```
The information of all the paths in the map are as follows:

-----------------------------------
No.1   Path: Gym-Library-40
Car allowed: Yes.
-----------------------------------
No.2   Path: Gym-GOGO-30
Car allowed: Yes.
-----------------------------------
No.3   Path: Gym-Dorm-45
Car allowed: Yes.
-----------------------------------
No.4   Path: Library-Lab-24
Car allowed: Yes.
-----------------------------------
No.5   Path: GOGO-Lab-20
Car allowed: Yes.
-----------------------------------
No.6   Path: GOGO-Dorm-5
Car allowed: Yes.
-----------------------------------
No.7   Path: Lab-Gym-40
Car allowed: Yes.
-----------------------------------
No.8   Path: PlayGround-Pool-30
Car allowed: Yes.
-----------------------------------
No.9   Path: Dorm-Pool-16
Car allowed: Yes.
-----------------------------------
No.10  Path: Library-Dorm-28
Car allowed: Yes.
-----------------------------------
No.11  Path: Lab-Dorm-89
Car allowed: Yes.
-----------------------------------
No.12  Path: PlayGround-Lab-37
Car allowed: Yes.
-----------------------------------
```

（5）结点信息显示

```
The information of all the spots in the map are as follows:

-------------------------------------------
NO.1 Spot: Gym
Adjacent Spots: Library-40 GOGO-30 Dorm-45 Lab-40
Description: The sport gym in the Sun Yat-sen University
-------------------------------------------
NO.2 Spot: Library
Adjacent Spots: Gym-40 Lab-24 Dorm-28
Description: The library in the SYSU
-------------------------------------------
```

```
NO.3 Spot: GOGO
Adjacent Spots: Gym-30 Lab-20 Dorm-5
Description: The commercial center in the Higher Educational Mega Center
------------------------------------------
NO.4 Spot: Dorm
Adjacent Spots: Gym-45 GOGO-5 Pool-16 Library-28 Lab-89
Description: The student dormitory of SYSU
------------------------------------------
NO.5 Spot: Lab
Adjacent Spots: Library-24 GOGO-20 Gym-40 Dorm-89 PlayGround-37
Description: The laboratory in the north area
------------------------------------------
NO.6 Spot: Pool
Adjacent Spots: PlayGround-30 Dorm-16
Description: Swimming pool
------------------------------------------
NO.7 Spot: PlayGround
Adjacent Spots: Pool-30 Lab-37
Description: Two different playgrounds in the SYSU
------------------------------------------
```

（6）删除边

```
Input the name of the spot that needs deleting: Gym

No.1 Spot "Gym" is destructed successfully.

No.1 Path between "Gym" and "library" is destructed successfully.

Spot "Gym" is deleted successfully.
```

（7）删除结点

```
The information of all the paths in the map are as follows:

----------------------------------
No.1   Path: gym-library-2
Car allowed: No.
----------------------------------

Input the number of the path that you want to delete: 1

No.1 Path between "gym" and "library" is destructed successfully.

Path: gym-library-2 is deleted successfully.
```

测试数据 B 组：

（1）主界面登陆

```
Undirected graph constructed.

Please input the password: sapphire

Correct password. Program entered successfully.
```

（输入密码完成后进入界面）

```
Input the number to execute the corresponding function below:
--------------------------------------------------
0                                  Clear The Sreen
--------------------------------------------------
1                                      Insert Spot
--------------------------------------------------
2                                      Insert Path
--------------------------------------------------
3                                      Delete Spot
--------------------------------------------------
4                                      Delete Path
--------------------------------------------------
5                              Show Adjacent Matrix
--------------------------------------------------
6                            Show Spots Information
--------------------------------------------------
7                            Show Paths Information
--------------------------------------------------
8                                Get Shortest Route
--------------------------------------------------
9                                  Exit The Program
--------------------------------------------------
Please input the number:
```

（2）插入结点

```
Please input the number: 1

Input the name of the spot: mingdeyuan

Input the description of the spot below:
The mingdeyuan in the SYSU

No.1 Spot "mingdeyuan" is constructed successfully.
```

（3）最短路径信息显示

```
mingde -> GOGO -> zhishan -> Lab
The shortest distance: 59
```

（4）邻接矩阵显示

The adjacent matrix is as follows:

	mingde	zhishan	GOGO	shensi	Lab	Pool	PlayGround
mingde	0	40	30	70	0	16	0
zhishan	40	0	5	0	24	0	0
GOGO	30	5	0	0	30	0	89
shensi	70	0	0	0	40	0	0
Lab	0	24	30	40	0	0	37
Pool	16	0	0	0	0	0	30
PlayGround	0	0	89	0	37	30	0

The information of all the spots in the map are as follows:

（5）结点信息

```
The information of all the spots in the map are as follows:

-------------------------------------------
NO.1 Spot: mingde
Adjacent Spots: zhishan-40 GOGO-30 shensi-70 Pool-16
Description: The mingdeyuan in the Sun Yat-sen University
-------------------------------------------
NO.2 Spot: zhishan
Adjacent Spots: mingde-40 Lab-24 GOGO-5
Description: The zhishanyuan in the SYSU
-------------------------------------------
NO.3 Spot: GOGO
Adjacent Spots: mingde-30 Lab-30 zhishan-5 PlayGround-89
Description: The commercial center in the Higher Educational Mega Center
-------------------------------------------
NO.4 Spot: shensi
Adjacent Spots: mingde-70 Lab-40
Description: The shensiyuan of SYSU
-------------------------------------------


NO.5 Spot: Lab
Adjacent Spots: zhishan-24 GOGO-30 shensi-40 PlayGround-37
Description: The laboratory in the north area
-------------------------------------------
NO.6 Spot: Pool
Adjacent Spots: PlayGround-30 mingde-16
Description: Swimming pool
-------------------------------------------
NO.7 Spot: PlayGround
Adjacent Spots: Pool-30 GOGO-89 Lab-37
Description: Two different playgrounds in the SYSU
```

第六部分

心得体会

实现一个中大的导游图，生成具体的路径信息供游客挑选，这个需求其实就是在对图这个重要的数据结构进行更深一步的了解与使用；在课上我们了解到一个图的基本操作有插入数据、删除数据、查找数据和重要操作有遍历数据，生成最短路径等等，这次这个 project 就是对这些操作的再次熟悉。

正所谓，越接近人的程序越复杂。基于现实情况，图中结点所代表的校园内的标志性建筑拥有很多属性和行为。每个标志性建筑的地理位置、建筑物与建筑物之间的距离以及从一个地方到另一个地方的行进方式，都是在创建一个图的时候需要考虑的。干巴巴的文字显示，不如可视化图形来的实在，所以这也是我们小组这次不足的地方，没有作出可视

化界面更加清晰的去显示路线，在接下的日子，会加紧这方面的学习。值得鼓励的一方面是这次比较创新的地方在于使用了新的算法来实现最短路径的计算。

此外，面向对象的条理化编程思想这次在该家谱程序中运用到了。通过实现对边类，结点类，图类，服务类的分开实现，更加清晰与条理化的显示我们小组编程的思想与代码具体的实现方式。其实也算是对上学期在初级实训所学到的程序架构知识的应用。如此，程序的整体架构更为清晰，也大大提高了程序的可读性和可维护性。

还有，在集体开发程序中，最考验的还是团队合作能力。团体合作在某种程度上也运用到面向对象的思想。每个人分工明确——整体架构设计、代码实现、实验报告设计等等分别由不同的队员负责。每个人都不用关心其他人的工作是如何实现的，只需要知道其他人能够按要求完成，并能直接把其他人的工作成果投入实用。每个人的工作看似互不干涉、内部却相辅相成，密不可分。每个人做好自己的模块，把各个模块拼接好就是一个完整的程序。这大概也是日后开发程序、乃至解决其他大型问题的基本方法吧。

该实验考察的是图的基本操作与简单实际应用。现实生活中可能发生的情况多种多样，这次设计的程序只能很机械化的实现很简单的功能，真要让自己设计的程序运用到实际中，仍需不断完善。

编程路漫漫其修远兮，吾将上下而求索。

第七部分

项目分工与自我评价

姓名	分工	自我评分	完成度
王迎旭	实验报告，算法的实现	93	35%
王友坤	实验报告，代码的debug	94	40%
汤万鹏	代码后期的细节处理	88	25%