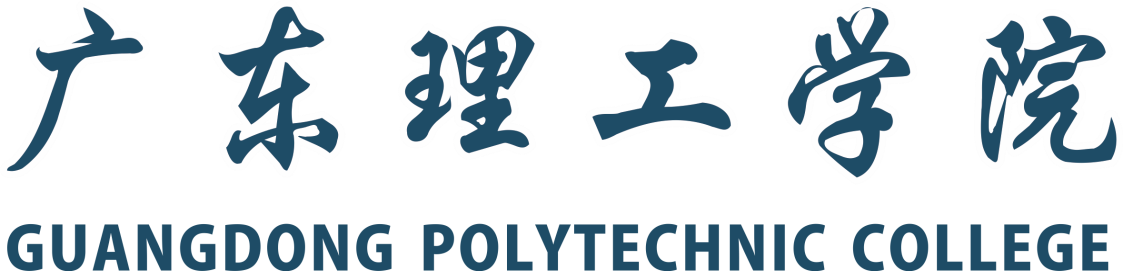
|  |  |
| --- | --- |
| **学 号** | 1812402101008 |
| 1812402101034 |
| 1812402101013 |
| 1812402101054 |

****

**数据结构与算法**

**课 程 设 计**

|  |  |
| --- | --- |
| **题 目** | **校园地图设计及其应用** |
| **学院（系）** | **信息技术学院** |
| **专 业** | **计算机科学与技术** |
| **班 级** | **2018级1班** |
| **姓 名** | **古金培** |
|  | **邱相贤** |
|  | **侯海彬** |
|  | **张世杰** |
| **指导 教师** | **郭锐** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **2020** | **年** | **07** | **月** | **09** | **日** |

摘 要

这一次的课程设计，目的是设计一个广东理工学院校园地图，为来访的客人提供各种信息查询服务，查询各地点的基本信息，查询相邻两地的路径长度，单独查询某地的基本信息，查询两点之间的最短路径。我们需要做的是，根据这些功能做出功能模块，再对每个模块进行细分，分为四个功能不一的模块，确定命名空间和各种变量和函数名，最后画出程序流程图。采用邻接矩阵的物理存储结构，再单独封装各个函数，使其保持相对独立性，部分函数需用作回调函数，以供其余函数使用，代码设计过程中将会对迪杰斯特拉算法进行改进，使其更加贴合需求。

【关键词】地图；邻接矩阵；函数封装；回调函数；迪杰斯特拉算法

Abstract

The purpose of this course design is to design a campus map of Guangdong Institute of technology to provide visitors with various information query services. The basic function of the software has been made clear. It can query the basic information of each location, the path length of adjacent two places, the basic information of a place, and the shortest path between two points. What we need to do is to make functional modules according to these functions, and then subdivide each module into four modules with different functions, determine the namespace and various variable and function names, and finally draw the program flow chart. The physical storage structure of adjacency matrix is adopted, and each function is encapsulated separately to keep its relative independence. Some functions need to be used as callback functions for other functions. In the code design process, dijestra algorithm will be improved to make it more suitable for the needs. After the completion of the code design, several tests should be carried out to improve the code logic to ensure that the code can achieve the desired purpose.

【Key Words】

Map; adjacency matrix; function encapsulation; callback function; Di Jie st algorithm

目 录

[第1章 问题描述与需求分析 1](#_Toc45271180)

[1.1 问题描述 1](#_Toc45271181)

[1.2 需求分析 1](#_Toc45271182)

[第2章 数据结构 3](#_Toc45271183)

[2.1 逻辑结构分析 3](#_Toc45271184)

[2.2 存储结构设计 5](#_Toc45271185)

[2.2.1 存储结构分析 5](#_Toc45271186)

[2.2.2预编译常量设置 5](#_Toc45271187)

[2.2.3结构体设计 6](#_Toc45271188)

[第3章 算法设计与实现 10](#_Toc45271189)

[3.1 算法设计 10](#_Toc45271190)

[3.1.1 算法思想概括 10](#_Toc45271191)

[3.1.2 各模块算法设计思路 10](#_Toc45271192)

[3.2 算法关键函数 16](#_Toc45271193)

[3.3 main函数 17](#_Toc45271194)

[3.4 算法分析 17](#_Toc45271195)

[第4章 调试分析 18](#_Toc45271196)

[4.1 测试用例设计 18](#_Toc45271197)

[4.2 运行结果 19](#_Toc45271198)

[4.3 调试过程问题分析 21](#_Toc45271199)

[第5章 总结 22](#_Toc45271200)

[5.1 收获与经验总结 22](#_Toc45271201)

[5.2 存在问题与努力方向 22](#_Toc45271202)

[参考文献 24](#_Toc45271203)

[致 谢 25](#_Toc45271204)

# 第1章 问题描述与需求分析

### 1.1 问题描述

设计一个广东理工学院校园地图，为来访的客人提供各种信息查询服务。

【基本要求】

1. 地图所含地点不少于15个。以图中顶点表示校内各地点,存放地点名称、代号、简介等信息;以边表示路径,存放路径长度等相关信息。
2. 设计地图的逻辑结构和物理结构，并分别用图形和C++语言来表示这两种结构。
3. 请用C++语言编写程序，为来访客人提供图中任意地点相关信息的查询，以及为来访客人提供图中任意地点的问路查询,即查询任意两个地点之间的一条最短的简单路径。
4. 给出相应算法的设计、实现及时间效率分析。

【测试数据】

1. 输入广东理工学院各地点名及有关该地点的基本信息，
2. 输入相邻两地点以及它们之间的路径长度。
3. 查询广东理工学院的任一地点的基本信息；
4. 查询广东理工学院任两个地点间的最短路径及路径长度；

### 1.2 需求分析

当访客进入广东理工学院时,当不知道该去哪个景点时,可利用本功能进行查询各景点的信息,从information.txt中将文件读取出来。此类问题用C++语言编写程序来解决。

此问题有以下几个需求；

1、提供校园平面图，使得能直观的了解学校。

2、提供地点信息查询，为各地点提供简短的介绍。

3、提供任意两地点间最短路径查询，并计算总路程。

根据要求，先将校园平面图信息抽象为无向网，用邻接矩阵存储

根据要求可知程序所能到达的功能：

A.图中任意景点的相关信息查询 ；模块采用一维数组将学校景点依次排放好编号G.vex[i].number=i 在选择校园介绍的时候，弹出G.vex[0]校园简介。在选择各景点信息的时候，可按编号查询 B.任意两个景点间的最短路径；利用弗洛伊德算法,求出最短路径,三维数组用来存储量景点最短路径的中间景点,二维数组用于存放两顶点间的路径长度.访客在输入自己的出发地点(或者当前所在地点),再输入目的地,方可在屏幕上显示出出发地点到达目的地的最短路径。C.任意两个景点间的所有路

D.退出该系统 。

# 第2章 数据结构

### 2.1 逻辑结构分析

1.所谓逻辑结构就是数据与数据之间的关联关系，准确的说是数据元素之间的关联关系。所有的数据都是由数据元素构成，数据元素是数据的基本构成单位。而数据元素由多个数据项构成。

2.逻辑结构有四种基本类型：集合结构，线性结构，树状结构和网络结构。也可统一的分为线性结构和非线性结构。

3.用计算机解决问题，分析并弄清数据的逻辑结构是求解问题的基础，也是求解问题的第一步。

4.数据的逻辑结构是用户根据需要建立起来的数据结构组织形式，它反映数据元素之间的逻辑关系而不是物理关系，是独立于计算机的。

5．数据中的数据元素之前可以有不同的逻辑关系。

本次课程设计内容为设计一个广东理工学院校园地图，为来访的客人提供各种信息查询服务，软件的基本功能已经是明确了出来，查询各地点的基本信息，查询相邻两地的路径长度，单独查询某地的基本信息，查询两点之间的最短路径。

学院地图设计稿为：

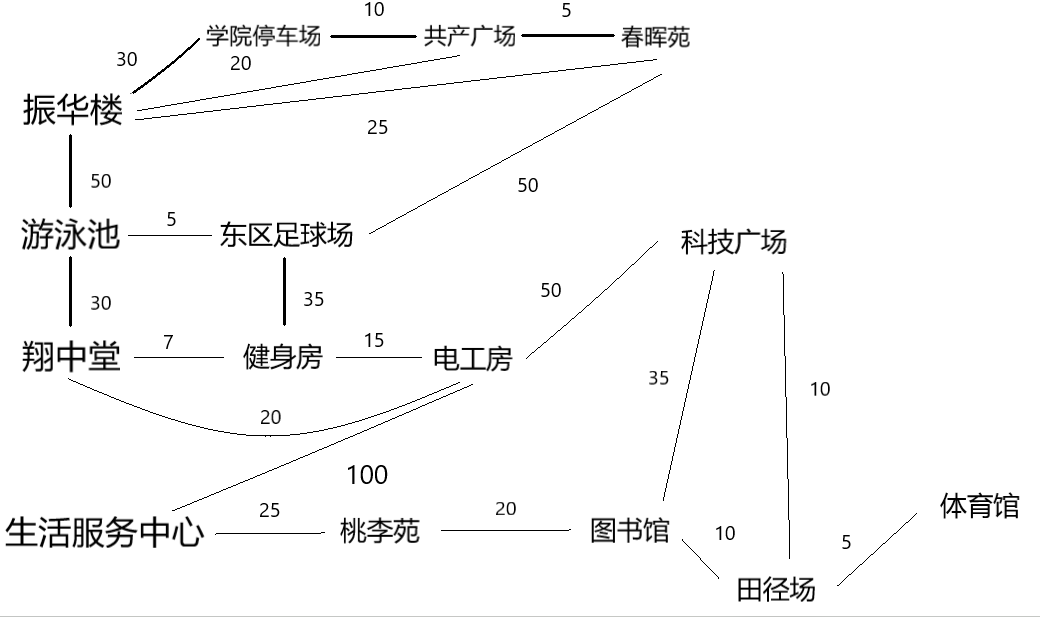


图2-1 校园地图逻辑图

表2-1 校园各地点的基本信息

|  |  |  |
| --- | --- | --- |
| 地点 | I D | 信息简介 |
| 振华楼 | Z | 信息技术学院学员的主要活动场所，内置教室数百间，室内硬件配置齐全 |
| 练车场 | L | 与学校共同合作的一个校外商家，致力于为学生提供物美价廉的考驾照之旅 |
| 美食广场 | M | 位于广东理工学院东区区域，与春晖苑相邻，其中店家琳琅满目，数不胜数，风格深受学生欢迎，是校园的一处好栖息地 |
| 春晖苑 | H | 提供了学生的日常餐饮，相对比与美食广场而言，春晖苑的装饰更为古风一些 |
| 游泳池 | Y | 泳池是室内的，所以在任何时间段都十分适合学生下水游泳 |
| 足球场 | Q | 与校游泳池相邻，有着良好的草地质感，给予运动员最舒适最安全的比赛场地 |
| 翔中堂 | X | 主要作为大型活动的场地，内部有覆盖全场的空调，观众席十分豪华 |
| 健身房 | J | 学生健身爱好者的天堂，内部硬件设施完善，安全设施齐全 |
| 电工房 | D | 位于校园的中间部分，主要保障学生日常生活用电安全 |
| 科技广场 | K | 我校的重要场所，其广场里的科技之星雕塑更是我校的标示性建筑，寓意着师生不断创新，追求真理的科技之心 |
| 服务中心 | S | 学生们称之为西区超市，因为其建筑和超市并无两样，除此之外，最大的特色就是网红街 |
| 桃李苑 | T | 称之为西区饭堂，内部的装饰还是店家的款式，都比春晖苑要好一些 |
| 图书馆 | G | 位于校门正前方，图书馆是追求知识的天堂，内部藏书齐全，种类繁多 |
| 田径场 | C | 举行校运会的主要场地，占地面积广，为学生们提供良好的锻炼场所 |
| 体育馆 | P | 与田径场相邻，这是我校唯一的室内运动馆，内有一个篮球场和四个羽毛球场 |

### 2.2 存储结构设计

### 2.2.1 存储结构分析

由于课程设计以学院地图为主，因而我们采用邻接矩阵的存储结构，图的遍历和图的应用算法设计等来设计数据结构。

1. 对于含有n个顶点的图，在采用邻接矩阵存储时，无论是有向图还是无向图。也无论边的数目是多少，其存储空间均为O（n²），所以邻接矩阵适合于存储学院地图这类边数较多的稠密图。虽然邻接表对于边数较少的稀疏图比邻接矩阵更节省空间，但学院地图这类并不属于稀疏图。
2. 在无向图中求顶点的度，使用邻接矩阵及邻接表法都很容易做到。在有向图中求顶点的度采用邻接矩阵比采用邻接表表示更方便，因而邻接矩阵更显优势。
3. 对于任一确定的无向图，邻接矩阵是唯一的（行列号与顶点编号一致），但邻接表不唯一（链接次序与顶点编号无关）。
4. 在邻接表上容易找到任意一顶点的第一个邻接点和下一个邻接点，但要判定任意两个顶点（vi，vj）之间是否有边或弧相连，则需搜索第i个或第j个链表，还不及邻接矩阵方便。
5. 对于任一确定的无向图，邻接矩阵是唯一的（行列号与顶点编号一致），但邻接表不唯一（链接次序与顶点编号无关）。

### 2.2.2预编译常量设置

预编译时定义三个常量：

分别有MAXVEX、NUMBER、INF三个

#define MAXVEX 15 //顶点数，应由用户定义

#define NUMBER 22 //边数，应由用户定义

#define INF 9999999 //定义一个无穷大数

### 2.2.3结构体设计

首先采用邻接矩阵的物理存储结构，所以需要设计两个结构体，一个有地点名称、代号和简介三个成员变量，另一个具有地点基本信息一维数组和表示各地点之间的权值的二维矩阵，并且还有代表地点个数和关联边数的两个成员变量。

结构体Vexs：

具有name、ch、str三个成员变量

typedef string VertexName; //顶点名称应由用户定义

typedef char replace; //顶点代号应由用户定义

typedef string info; //顶点简介应由用户定义

typedef struct {

VertexName name; //地点名称

replace ch; //地点代号

info str; //地点简介

}Vexs;

结构体Graph：

具有vexs[MAXVEX]、arc[MAXVEX][MAXVEX]、numVertexes、numEdges四个成员变量

typedef int EdgeType; //边上的权值类型应由用户定义

typedef struct {

Vexs vexs[MAXVEX]; //顶点表(以中文信息存储)结构体数组

EdgeType arc[MAXVEX][MAXVEX]; //邻接矩阵，可看作边

int numVertexes, numEdges; //图中当前的顶点数和边数

}Graph;

为了事先完成好校园地图信息与各地点关联关系的存储，我们使用了四个不同的辅助数组。

vexname数组：存储了所有地点各自的名称信息

VertexName vexname[15] = {

"振华楼",

"学院练车场",

"美食广场",

"春晖苑",

"学校游泳池",

"东区足球场",

"翔中堂",

"健身房",

"电工房",

"科技广场",

"生活服务中心",

"桃李苑",

"图书馆",

"田径场",

"体育馆"

};

vexch数组：存放了所有地点各自的代号信息

replace vexch[15] = {

'Z','L','M','H','Y','Q','X','J','D','K','S','T','G','C','P'

};

vexstr数组：存放了所有地点各自的简介信息

info vexstr[15] = {

"振华楼是信息技术学院学员的主要活动场所，内置教室数百间，室内硬件配置齐全，当为模范教学楼！",

"学院练车场是与学校共同合作的一个校外商家，致力于为学生提供物美价廉的考驾照之旅，练车场位于校内，十分方便学生全天候练车！",

"美食广场位于广东理工学院东区区域，与春晖苑相邻，其中店家琳琅满目，数不胜数，风格深受学生欢迎，是校园的一处好栖息地！",

"春晖苑也被称之为东区食堂，与美食广场共同提供了学生们的日常餐饮生活，相对比与美食广场而言，春晖苑的装饰更为古风一些！",

"学校游泳池位于学校的东区区域，泳池是室内的，所以在任何时间段都十分适合学生下水，当然夏日泳池的plmm才是不可多得的一道风景线！",

"东区足球场，与学校游泳池相邻，有着良好的草地质感，给予运动员最舒适最安全的比赛场地，见证了无数广理足星的诞生！",

"翔中堂也被称之为礼堂，翔中堂主要作为大型活动的场地，内部有覆盖全场的空调，观众席十分豪华，并且座椅多达两层楼之多！",

"健身房位于礼堂的对角处，它是学生健身爱好者的天堂，内部硬件设施完善，安全设施齐全,如果店家可以免费让学生体验，再好不过了！",

"电工房位于校园的中间部分，当学生们宿舍发生电路问题时，都可以求助电工房来进行修理，当然电工一般是十分忙绿的，毕竟只有一所电工房！",

"科技广场是我校的重要场所，其广场里的科技之星雕塑更是我校的标示性建筑，寓意着师生不断创新，追求真理的科技之心！",

"生活服务中心也被学生们称之为西区超市，因为其建筑和超市并无两样，除此之外，最大的特色就是网红街，试想，一个宁静的夜晚，在满是星星灯下的你和好朋友们畅聊着星辰大海，是多么惬意的事情！",

"桃李苑位于西区区域，也称之为西区饭堂，不过其内部的装饰还是店家的款式，都比春晖苑要好一些！",

"图书馆位于我校正门的正前方，正所谓一进门就得以闻其书香气，图书馆是追求知识的天堂，内部藏书齐全，种类繁多，是广理学子最热爱的学习之处！ ",

"田径场是我校举行校运会的主要场地，占地面积广，为西区的学生们提供了不可多得的课后锻炼场所！",

"体育馆与田径场相邻，这是我校唯一的室内运动馆，内有一个篮球场和四个羽毛球场，如果能再多一些室内场地，那将十分深受学生们的欢迎！",

};

特别注意：vexname、vexch、vexstr这三个数组的下标一一对应，也就是说vexname[5]只能对应vexch[5]和vexstr[5]的信息，因为这是事先设计好的地图信息，必须一一对应才能保证程序运行结果与设计稿相符合！

# 第3章 算法设计与实现

### 3.1 算法设计

### 3.1.1 算法思想概括

此次设计的程序将会把整个程序所有的流程都封装在一个function函数中，在主函数中完成调用，function函数应该具备以下两个特性：

1. 能够显示菜单栏
2. 能够完成题目所需的功能要求

采用大封装的思想，各函数之间的独立性强

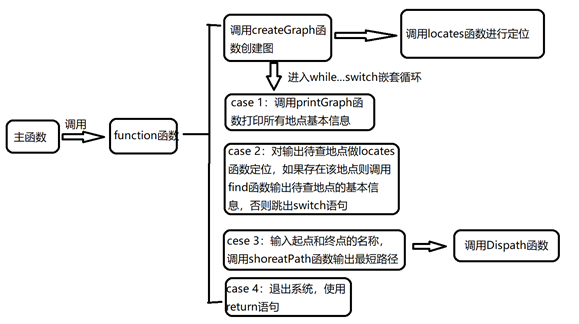


图3-1 软件整体框架图

### 3.1.2 各模块算法设计思路

根据已知需求，需要设计四个主要功能函数和四个辅助函数。这些函数均为单独封装，各函数之间保持相对独立性，部分函数需要用作回调函数供特定函数所调用。

void printGraph(Graph g)：

第一个主要功能函数的功能为打印广东理工学院所有地点的名称、代号与简介，此功能较为简单，只需要遍历一遍广东理工学院地图取出信息即可；此函数名称为printGraph，无返回值，函数类型为void，需要一个形式参数，也就是需要广东理工学院地图的结构体变量，来对广东理工学院地图的基本信息进行一个访问操作。

printGraph（Graph g）函数：

1. 遍历广东理工学院校园的各地点信息矩阵
2. 打印广东理工学院校园各个地点的基本信息
3. 程序结束

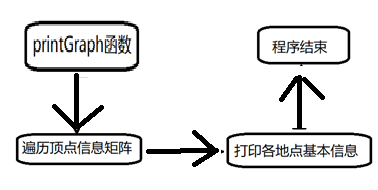


图3-2 printGraph函数流程图

void find(Graph g, string ch)：

第二个主要功能函数的功能为查询广东理工学院的任一地点的基本信息，此功能需要将待查询的地点和广东理工学院地图中已有地点做线性次比较，如果图中存在待查地点，则打印待查地点的名称、代号、简介等基本信息；如果图中并不存在该待查地点，则提示错误信息并且，转到菜单栏页面，；此函数名称为find，无返回值，函数类型为void，需要两个形式参数，第一个为广东理工学院地图的结构体变量，第二个为string类型的变量，该变量的值应该等于用户所输入的待查地点的值。

find（Graph g，string ch）函数：

1. 遍历广东理工学院校园的各地点信息矩阵
2. 判断ch的值是否存在于地点基本信息结构体数组中
3. 如果ch存在，则打印ch地点的基本信息
4. 程序结束

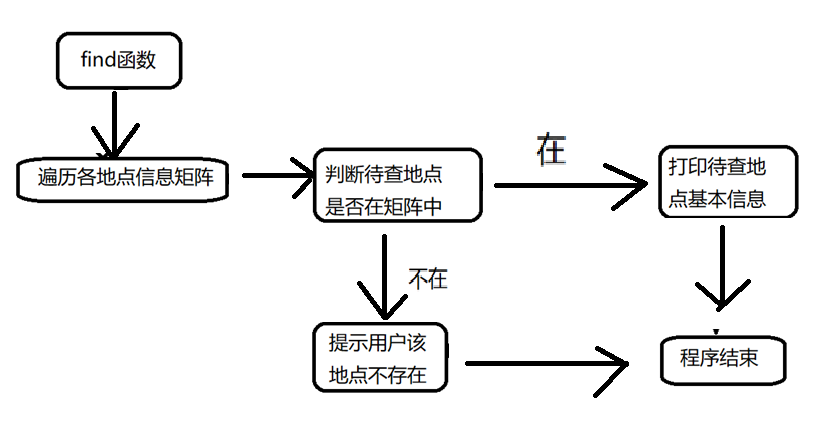


图3-3 find函数流程图

void shortestPath(Graph g, int v, int e)：

第三个主要功能函数为迪杰斯特拉算法的改进函数，使其更加贴合需求，减少一些不必要的操作；原生迪杰斯特拉算法的功能为求从一个顶点到其余各顶点的最短路径，而此次课程设计的需求为求任一两个地点之间的最短路径，需要对迪杰斯特拉算法进行部分改进方能实现。 此算法名称为shortestPath,无返回值，函数类型为void，需要三个形式参数，第一个为广东理工学院地图的结构体变量，第二个为起始地点的string变量，第二个为终止地点的string变量，这两个string变量的值都应该与用户输入的值相对应。该函数将在核心函数中进行主要讲解，再次只作概述。

void createGraph(Graph \*g)：

第四个主要功能函数的功能为将广东理工学院地图地点的基本信息存储到计算机中，也就是建立一个无向网图，此功能的实现较为简单，将广东理工学院地图的结构体变量里的变量成员一一初始化即可。此函数名称为createGraph，无返回值，函数类型为void，需要一个形式参数，即广东理工学院地图的结构体变量的指针，通过初始化形参从而初始化实参。

createGraph（Graph \*g）函数：

1. 初始化广东理工学院地点名称、代号、简介
2. 初始化广东理工学院校园的各地点权值邻接矩阵
3. 程序结束

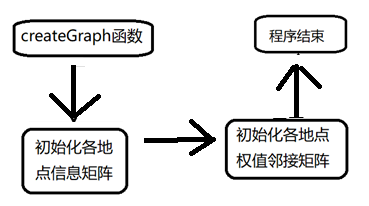


图3-4 createGraph函数流程图

int locates(Graph \*g, string ch)：

第一个辅助函数的功能为对任一一个地点名称进行定位，即定位到地点名称相对应的序号。因为广东理工学院地图地点的所有信息是采用邻接矩阵所存储的，所以每一个地点名称都将对应一个序号。所以定位函数的主要作用就是将地点名称转换为序号，以便于取到地点的相应信息进行一个操作。此函数的名称为locates，有返回值，函数返回类型为int型，需要两个形式参数，第一个为广东理工学院地图的结构体变量，第二个为string类型的变量，该变量的值应该等于用户所输入的待查地点的值。

locates（Graph \*g，string ch）函数：

1. 遍历广东理工学院校园的各地点基本信息数组
2. 判断ch是否存在于基本信息数组中
3. 若存在，则返回相对应的地点序号；若不存在，则返回-1
4. 程序结束

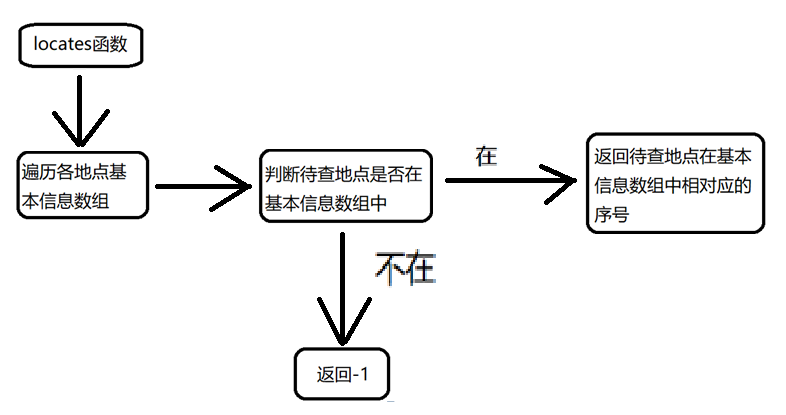


图3-5 locates函数流程图

void menum()：

第二个辅助函数的功能为显示图形界面的菜单栏信息，给与用户较好的用户体验。此函数的名称为menum，没有返回值，函数类型为void，无任何形式参数。

menum（）函数：

打印菜单栏的具体文字展示在控制台上

void find(Graph g, string ch) {

int i;

for (i = 0; i < g.numVertexes; i++) {

if (ch == g.vexs[i].name) {

cout << "地点名称:" << g.vexs[i].name << endl;

cout << "代号:" << g.vexs[i].ch << endl;

cout << "简介:" << g.vexs[i].str << endl;

cout << endl;

}

}

}

void Dispath(int dist[], int path[], int vv, int ee, Graph g)：

第三个辅助函数的功能为输出起始点与终止点之间的最短路径与路径长度，此函数会在shortestPath函数中被回调，因为这个输出动作可能会被重复执行，所以独立封装为一个函数，使代码重用。此函数名称为，disPath，无返回值，函数类型为void，需要五个形式参数，第一个参数为dist数组，该数组里存储着该出发点至各个顶点之间的最短路径长度，第二个参数为path数组，该数组存储着该出发点至各个顶点之间的最短路径，第三个参数为起始地点进行定位函数之后的序号值，第四个参数为终止地点进行定位函数之后的序号值，第五个参数为广东理工学院地图的结构体变量。

disPath（int dist[], int path[], int vv,int ee,Graph g）函数：

1. 创建一个apath数组来存放最短路径
2. 判断path数组终止点的值，也就是path[ee]的值。如果值为-1，也就是没有前驱结点，那么代表初始点到该序号对应的地点是没有最短路径的；如果值不为-1，做一个while循环，条件为path数组编号值等于vv时，也就是等于初始点的值时，循环结束
3. while循环里将起始点到终止点之间的最短路径都保存在了apath数组里面
4. 打印apath数组里的值，构成最短路径
5. 程序结束

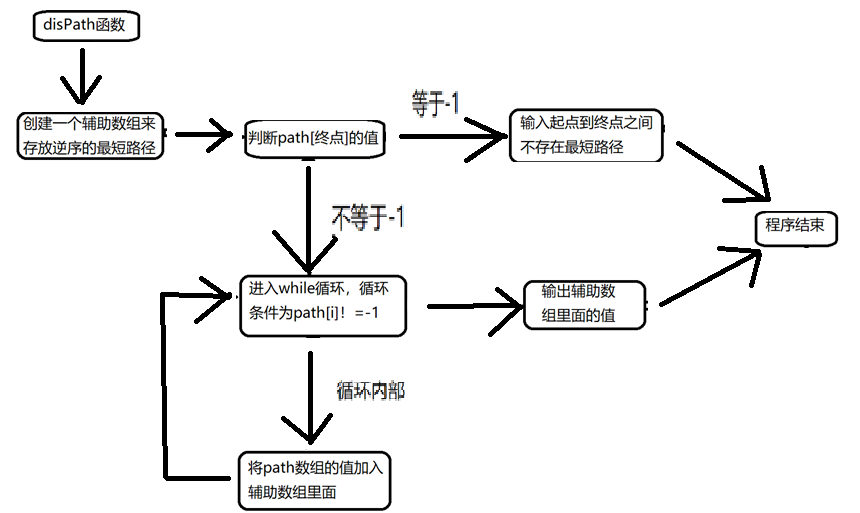


图3-6 disPath函数流程图

int function()：

第四个辅助函数的功能为将所有的程序功能、菜单、一切操作都封装在一个函数里面，让主函数尽可能的精简直观，主要用while搭配switch语句进行函数的设计。此函数的名称为function，无返回值，函数类型为int，无任何形式参数。

function（）函数：

1. 创建广东理工学院地图结构体变量g
2. 调用createGraph函数初始化广东理工学院地图
3. 使用while-switch语句循环菜单栏，每执行完一个指定操作后都会重新打印菜单栏
4. case ‘1’功能为查询广东理工学院所有地点信息，调用了printGraph函数
5. case ‘2’功能为查询广东理工学院任一地点的基本信息，调用了find函数
6. case ‘3’功能为查询广东理工学院任两个地点间的最短路径和长度，调用了shortestPath函数和disPath函数
7. case ‘4’功能为退出，使用了终止程序运行的return语句
8. 程序结束

function函数的具体结构在算法概述部分已经十分清晰，在此不做过多赘述

### 3.2 算法关键函数

shortestPath函数为核心函数，具体功能为询广东理工学院任两个地点间的最短路径和长度，其中涉及了较多的函数回调，是整个程序中较为复杂的一个函数。这里给出shortestPath函数的源代码片段

shortestPath（Graph g，int v，int e）函数：

1. 创建dist、path和F数组
2. 初始化dist、path数组，dist数组里的值为权值矩阵中行序号为v的那一行的值；如果权值邻接矩阵[v，i]（0<i<顶点个数）小于INF，则path的值为i；F数组存放的是该地点是否求出最短路径的信息，值为0或1,1代表已求出最短路径，0则反之
3. 在未求出最短路径的地点中，寻找出dist数组中的最小值暂存起来，并将该最小值地点的F数组的值置为1，表明已求出最短路径
4. 在未求出最短路径的地点中，更新dist数组的值，因为步骤2已经求出了一个地点的最短路径，那么就可以借助新确定的地点作为中转站，也就是可能发生下面这种情况（初始点-终止点的距离 > 初始点-中转站-终止点的距离），发生以上情况要将dist数组的值更新为较小的那一个；此外还需要更新path数组的值，path数组存储的是作为中转站地点通过定位函数的到的序号值
5. 重复执行（顶点数-1）次步骤3和步骤4
6. 程序结束

其中可以看到shortestPath函数调用了disPath函数，disPath函数的功能是用来打印两地点间的最短路径与长度

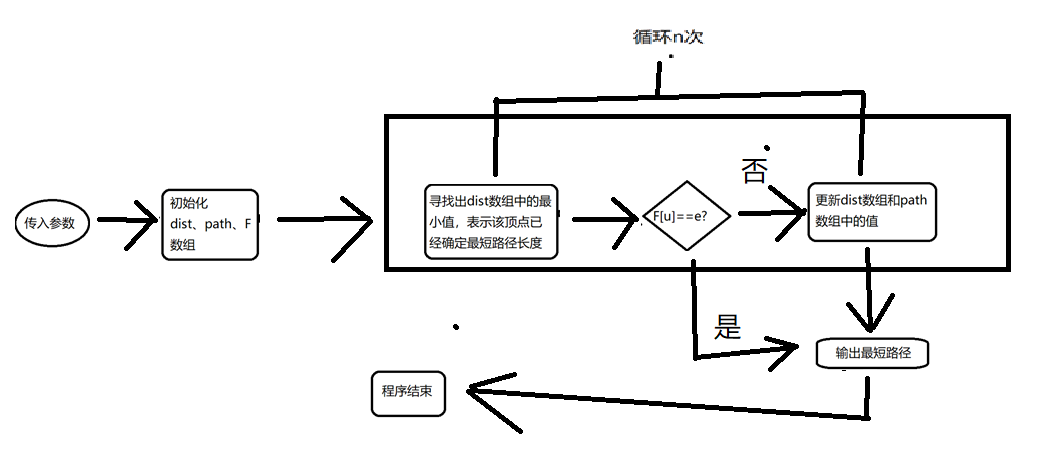


图3-7 shorestPath函数流程图

### 3.3 main函数

因为程序的功能都被function函数一起封装了起来，所以我们的主函数程序设计十分简单，即调用一个function函数就完成了主函数的编写。由于前文有较多的篇幅讲解了function里面封装且调用的所有函数，在此部分内容将不做过多赘述。

### 3.4 算法分析

由于广东理工学院校园地图的基本信息是使用邻接矩阵来进行存储的，如果说广东理工学院校园地点的规模为n，那么所需要的用到的空间最多为n\*n个单位，即整个程序的空间复杂度T（n）=n\*n；

纵观整个程序，没有哪一个函数的时间复杂度是超过n\*n的，也就是说所需最久的时间不过是n\*n而已，即整个程序的时间复杂度O（n）=n\*n；

# 第4章 调试分析

### 4.1 测试用例设计

地点选取：

选择了15个广东理工学院校园地点，分别为振华楼、学院练车场、美食广场、春晖苑、学校游泳池、东区足球场、翔中堂、健身房、电工房、科技广场、生活服务中心、桃李苑、图书馆、田径场、体育馆这些地点

地点之间的边关联关系：

关联两地点的边有22条，并且关联关系为双向的性质，如下所示：

振华楼 <->学院练车场 30

振华楼 <-> 美食广场 20

振华楼 <-> 春晖苑 25

振华楼 <-> 学校游泳池 50

学院练车场 <-> 美食广场 10

美食广场 <-> 春晖苑 5

学校游泳池 <-> 东区足球场 5

学校游泳池 <-> 翔中堂 30

东区足球场 <-> 春晖苑 50

东区足球场 <-> 健身房 35

翔中堂 <-> 健身房 7

翔中堂 <-> 电工房 20

健身房 <-> 电工房 15

电工房 <-> 科技广场 50

电工房 <-> 生活服务中心 100

科技广场 <-> 图书馆 35

科技广场 <-> 田径场 10

生活服务中心 <-> 图书馆 40

桃李苑 <-> 生活服务中心 25

图书馆 <-> 桃李苑 20

图书馆 <-> 田径场 10

田径场 <-> 体育馆 5

为了尽量模拟校园地图的智能功能，并且考虑到此程序是由用户来进行使用的，所以我们减少了用户还要将校园信息输入终端的操作，因为这些地点信息操作应该是由设计者事先完成，所以当程序运行后是没有让用户输入校园地图信息与各地点关联关系的操作部分的。

### 4.2 运行结果

测试功能1：查询广东理工学院所有地点信息

预计效果：显示所有广东理工学院地点的基本信息（地点名称、代号、简介）

实际运行结果：

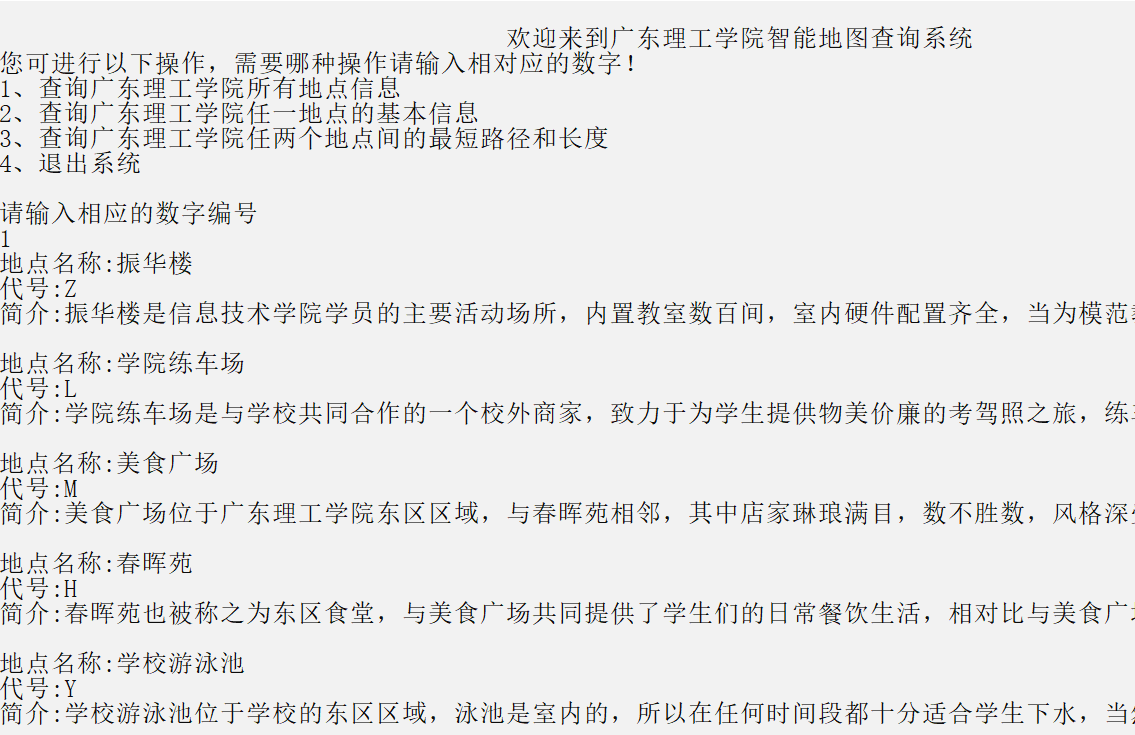


图4-1查询广东理工学院所有地点信息

测试功能2：查询广东理工学院任一地点的基本信息

测试数据：“振华楼”

预计效果：显示所有广东理工学院待查地点的基本信息（地点名称、代号、简介）

实际运行结果：

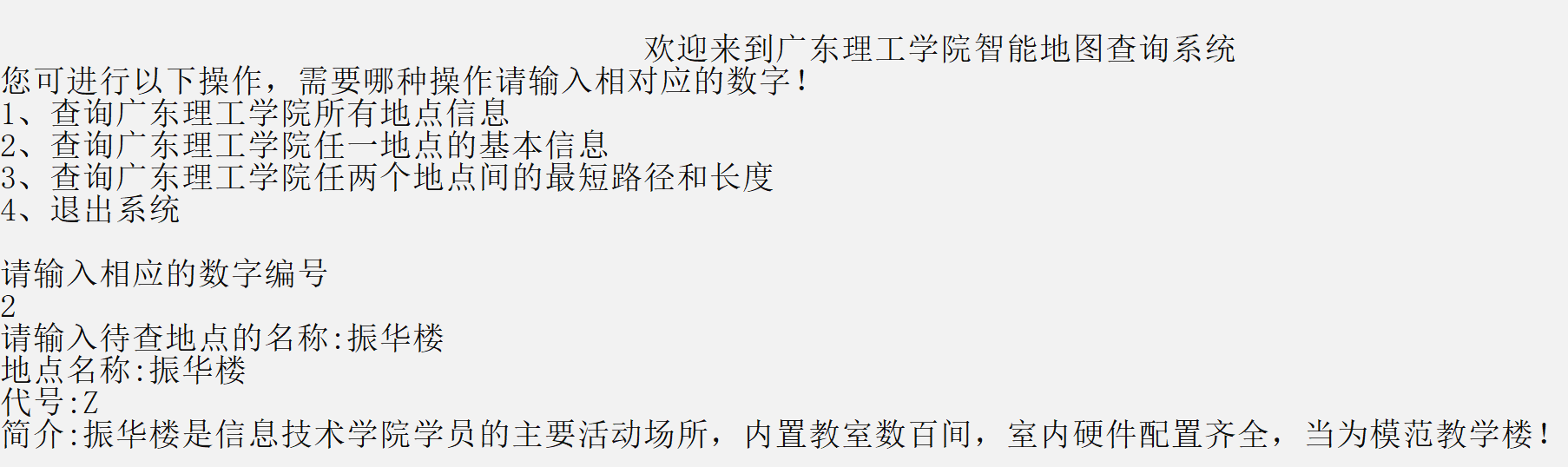


图4-2查询广东理工学院任一地点的基本信息

测试的地点为振华楼，可以看到，程序成功地将振华楼所有的基本信息都显示了出来

测试功能3：查询广东理工学院任两个地点间的最短路径和长度

测试数据：“振华楼 体育馆”

预计效果：显示待查起点与待查终点的最短路径和其长度

实际运行结果：

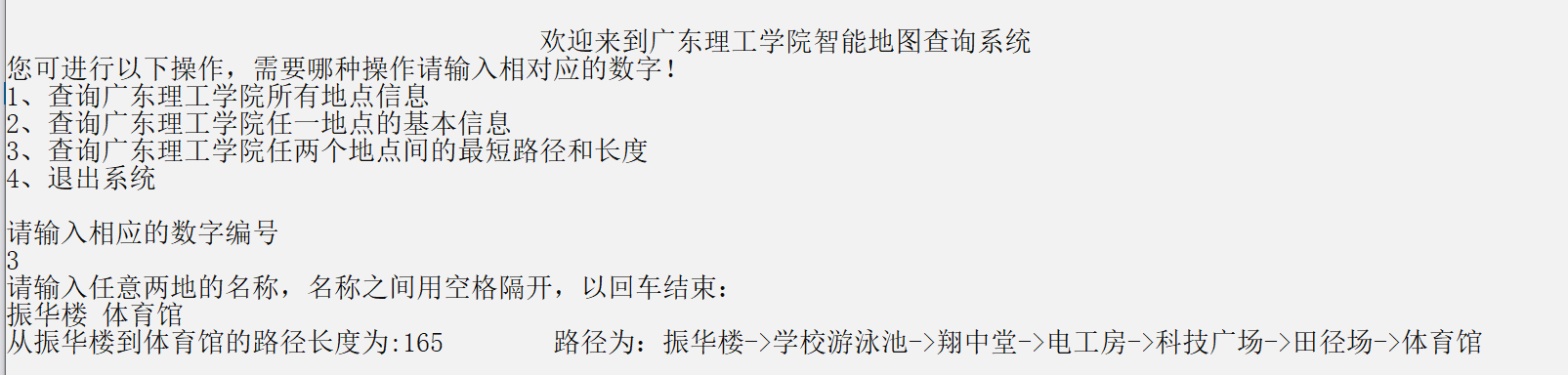


图4-3查询广东理工学院任两个地点间的最短路径和长度

测试的两个地点为振华楼和图书馆，可以看出其结果与设计稿相符合，基本功能已经实现。

测试功能4：退出系统

预计效果：显示退出此程序的提示

实际运行结果：

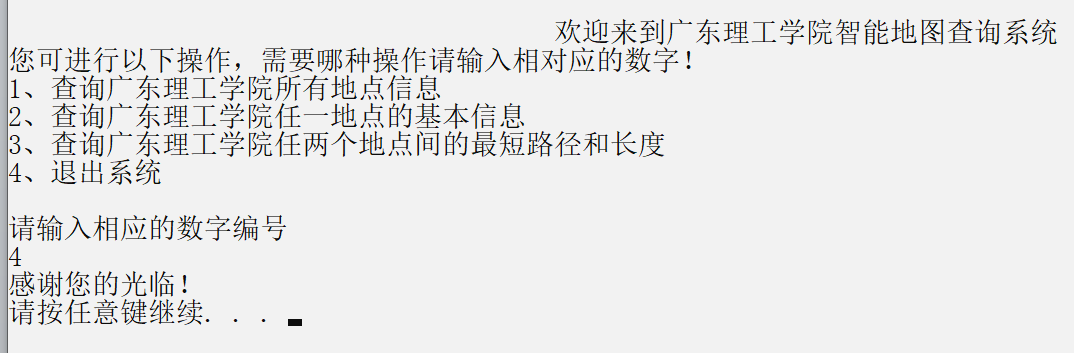


图4-4退出系统

退出系统功能也是较为简单地就成功实现了，因为在主函数中加了system（“pause”）语句，在vs编译环境中才能让窗口做多一会的停留，直至输入任意字符窗口才消失，程序结束

### 4.3 调试过程问题分析

问题描述：while循环嵌套switch语句出现输入多字符而发生程序多余执行的情况

分析：

当程序输入多字符时，多余的字符和回车会存储在输入流中，并不会自动清空，当下次输入操作来临时，输入流内的字符会自动给cin接收，而不会让用户手动输入，这就导致了程序多余执行的情况。这是由于输入流储存了多余的第一次输入字符而导致的，所以解决这类问题需要把输入流中的字符在下一次cin操作来临前都给清除掉。

解决方案：

在while循环的最末尾，也就是switch的函数体结束后加上cin.ignore(100, '\n');语句，该语句具有自动接收100个输入流里的字符的作用，从而实现清空输入流的作用

# 第5章 总结

### 5.1 收获与经验总结

在此次课程设计中，我们组从分工到最后的落实的任务都十分明确，这也奠定了我们课程设计速度如此之快的基础；除此之外，这加强了我们小组各成员之间的团队协作开发能力，在开发过程中，通过彼此之间的相互交流，很好地完成了信息交换，并达到完成程序设计的结果，相信通过这次团队协作，我们各自对团队开发概念的认知将会更进一步，这为将来从事工作的我们奠定了良好基础。回到课程设计的内容本身，我们组的课题是校园地图设计及其应用，首先进行一个程序设计最先考虑的环节必定是需求分析，在这个环节我们加深了对题目要求的理解，并将它运用于实际设计中，对我们的程序开发产生了良好的蝴蝶效应。需求分析完成之后，必定是对程序信息所需要的的存储结构进行一个选择，那么对于我们的课设而言，毫无疑问就是选择邻接矩阵最为方便，并且效率也会更高，在这个环节，我们组一致认为邻接矩阵为最佳的存储结构，当然这是因为以往程序设计的时候，使用邻接矩阵总能比邻接表的物理存储结构要方便存取与访问，事实证明，在开发过程中，邻接矩阵的方便给予了程序的高效率进行。在算法设计这一环节，我们通过对需求的分析从而一一对应了具体的封装函数，这加深了我们对面向对象编程这个概念的理解与应用。在程序调试这一块，除了出现whil语句嵌套switch语句会出现程序冗余执行的情况，没有太多的bug，所以针对这唯一的问题我们很快的找到了答案，从这个问题中我们知道了以前总是忽视的输入流的问题，相信在此次课程设计之后，我们对于程序设计的考虑角度会更为多元化与精确化。

### 5.2 存在问题与努力方向

虽然说此次课程设计总体来看是十分完美的，但是也有不足的地方。第一，在需求分析的时候，总不能及时地找出重点，并转化为我们的专业领域的问题。对于这个问题，除了多进行项目，累积经验之外，别无途径，所以，在此次课程设计结束之后，我们会更加努力地去寻找项目，并对其进行落地处理，提升自己的分析能力与总结能力。第二，在对于原生算法的中心思想上面，我们的理解显得十分粗糙。对于这个问题，只有通过不断的理解原生算法的具体思想和实践总结，才能一步一步逼近原作者对其自身算法的理解，因为只有当你对前人的算法理解程度至少和前人在同一水平线上的时候，你才可能有创造性、启发性的写出自己独特的算法；这是算法能力的养成阶段的必经之路，工作面试的时候公司也十分注重学生的算法能力，所以这一块是必须加强的；在学有余力之下，去leetcode刷题是提升自我算法能力的一种可取之道。

# 参考文献

1. 张乃孝，裘宗燕。数据结构与C++面向对象的途径。北京：高等教育出版社，1998
2. 陈慧南。数据结构—C++语言描述。北京：人民邮电出版社，2005
3. Adam Drozdek 数据结构与算法。北京：清华大学出版社，2006
4. 严蔚敏，吴伟民。数据结构。北京：清华大学出版社，1997
5. 严蔚敏,吴伟民,米宁. 数据结构题集(C语言版).北京:清华大学出版社,2007.
6. [Krishnamoorthy](https://search.dangdang.com/book/search_pub.php?category=01&key2=Krishnamoorthy&order=sort_xtime_desc) R，[Kumaravel](https://search.dangdang.com/book/search_pub.php?category=01&key2=Kumaravel&order=sort_xtime_desc" \t "https://www.icourse163.org/course/_blank) [G.I](https://search.dangdang.com/book/search_pub.php?category=01&key2=G.I&order=sort_xtime_desc). Data Strcture Using C（Photocopy Edition）, Tsinghua University Press,2009.
7. (印)[慕克吉](https://search.dangdang.com/book/search_pub.php?category=01&key2=%C4%BD%BF%CB%BC%AA&order=sort_xtime_desc) 著，[张长富](https://search.dangdang.com/book/search_pub.php?category=01&key2=%D5%C5%B3%A4%B8%BB&order=sort_xtime_desc) 等译. 数据结构(C语言版)1000个问题与解答，北京:清华大学出版社,2010.
8. [Larry Nyhoff](https://www.calvin.edu/~nyhl/index.html). [ADTs, Data Structures, and Problem Solving with C++.](https://vig.prenhall.com/catalog/academic/product/0,1144,0131409093,00.html)Second Edition, [Calvin College](https://cs.calvin.edu/),2005.
9. 沈显君.C++语言程序设计教程(第3版).北京：清华大学出版社，2015.

[10] 钱能.C++程序设计教程(第3版)通用版. 北京：清华大学出版社，2019.

[11] 谭浩强.C++程序设计(第3版).北京：清华大学出版社，2015.

# 致 谢

首先特别感谢郭锐老师在课程设计开始之前对我们组的课设引导，这让我们很容易地找到了方向，在此十分感谢郭锐老师的指引！郭锐老师让我们知道了什么样的信息改用什么样的物理存储结构，这对于信息的管理与使用提高了极大效率，也使得我们的程序开发过程变得简洁许多。在算法设计方面，郭锐老师给了我们许多点拨，例如在求任意两地点之间的最短路径与长度时，郭锐老师说将迪杰斯特拉算法加以改进就能实现，并为我们重新讲解了迪杰斯特拉算法的中心思想，通过修改迪杰斯特拉算法，我们得到了属于自己的一个最短路径算法，我们也对迪杰斯特拉算法的理解更进一步，这是令我们十分欣喜的。在程序调试出现bug时，郭锐老师总能及时给予我们精确的解答，让我们在debug的过程中学到了书本上未曾接触的新事物，这让我们的编程视野变得更加全面客观。最后，由衷地感谢郭锐老师给予本组的课设帮助与点拨，没有郭锐老师我们也做不到如此完美，谢谢！