数据结构课程设计说明书

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 姓名 | 平时 | 演示 | 答辩 | 总评 |
| 郭 朕 |  |  |  |  |
| 张永栋 |  |  |  |  |
| 黄嘉灵 |  |  |  |  |

学生姓名：郭 朕学号：1607084132

张永栋学号：1607084133

黄嘉灵学号：1607084142

学 院： 大数据学院

专 业： 网络工程

指导教师： 庞晓琼、井超

2018年1月

目录

[**1** **引言** 1](#_Toc503254938)

[**2** **设计要求及内容** 1](#_Toc503254939)

[**3** **本设计所采用的数据结构** 2](#_Toc503254940)

[**4** **各功能模块的详细设计及实现** 3](#_Toc503254941)

[**4.1** **校园平面图的创建以及显示功能模块** 3](#_Toc503254942)

[4.1.1 详细设计思想 3](#_Toc503254943)

[4.1.2 功能模块核心代码 4](#_Toc503254944)

[**4.2** **校园景点信息的查询功能模块** 5](#_Toc503254945)

[4.2.1 详细设计思想 5](#_Toc503254946)

[4.2.2 功能模块核心代码 6](#_Toc503254947)

[**4.3** **查询任意两景点的所有路径** 7](#_Toc503254948)

[4.3.1 详细设计思想 7](#_Toc503254949)

[4.3.2 功能模块核心代码 8](#_Toc503254950)

[**4.4** **查询途径多个景点的最佳路径** 10](#_Toc503254951)

[4.4.1 详细设计思想 10](#_Toc503254952)

[4.4.2 功能模块代码实现 11](#_Toc503254953)

[**4.5** **查询任意两景点之间的最短路径** 14](#_Toc503254954)

[4.5.1 详细设计思想 14](#_Toc503254955)

[4.5.2 功能模块代码实现 14](#_Toc503254956)

[**4.6** **主函数界面设计** 17](#_Toc503254957)

[4.6.1 详细设计思想 17](#_Toc503254958)

[4.6.2 功能模块代码实现 17](#_Toc503254959)

[**5** **分工情况** 21](#_Toc503254960)

[**6** **代码运行情况** 22](#_Toc503254961)

[**6.1** **校园平面图显示情况** 22](#_Toc503254962)

[**6.2** **校园景点信息查询情况** 22](#_Toc503254963)

[**6.3** **任意两景点所有路径查询情况** 22](#_Toc503254964)

[**6.4** **多个景点最佳路径查询情况** 23](#_Toc503254965)

[**6.5** **任意两景点最短路径查询情况** 23](#_Toc503254966)

[**6.6** **主函数界面显示情况** 24](#_Toc503254967)

[**7** **课程设计体会** 25](#_Toc503254968)

1. **引言**

《数据结构》课程主要介绍最常用的数据结构，阐明各种数据结构内在的逻辑关系，讨论其在计算机中的存储表示，以及在其上进行各种运算时的实现算法，并对算法的效率进行简单的分析和讨论。进行数据结构课程设计要达到以下目的：

1. 了解并掌握数据结构与算法的设计方法，具备初步的独立分析和设计能力；
2. 初步掌握软件开发过程的问题分析、系统设计、程序编码、测试等基本方法和技能；
3. 提高综合运用所学的理论知识和方法独立分析和解决问题的能力；
4. 训练用系统的观点和软件开发一般规范进行软件开发，培养软件工作者所应具备的科学的工作方法和作风。
5. **设计要求及内容**
   1. **设计要求**

设计一个校园导游咨询系统，可以为来访的客人提供各类信息的查询。比如：查询中北大学的校园平面图、图中各个景点的信息、任意两个景点之间的最短路径、任意两个景点之间的所有路径、途径多个景点的最佳路径等。

* 1. **设计内容**

1. 设计中北大学的校园平面图，所含景点不少于10个，以图中顶点表示校内各景点，存放景点名称、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. 为来访客人提供图中任意景点相关信息的查询；
3. 为来访客人提供图中任意景点的问路查询，即查询任意两相景点之间的一条最短的简单路径；
4. 提供图中任意景点问路查询，即求任意两个景点之间的所有路径；
5. 提供校园图中多个景点的最佳访问路线查询，即求途经这多个景点的最佳路径。
6. **本设计所采用的数据结构**

**3.1 逻辑结构**

图状结构——无向图

**3.2 物理结构**

顺序结构——邻接矩阵

代码如下：

//定义景点结构体

typedef struct {

int num; //编号

char name[Max]; //名称

char introduction[Max]; //简介

} beauty;

//定义边结点结构体

typedef struct {

int length; //路程长度

} edgeNode;

//定义图的邻接矩阵结构体

typedef struct {

int nodenum; //结点个数

int edgenum; //边数

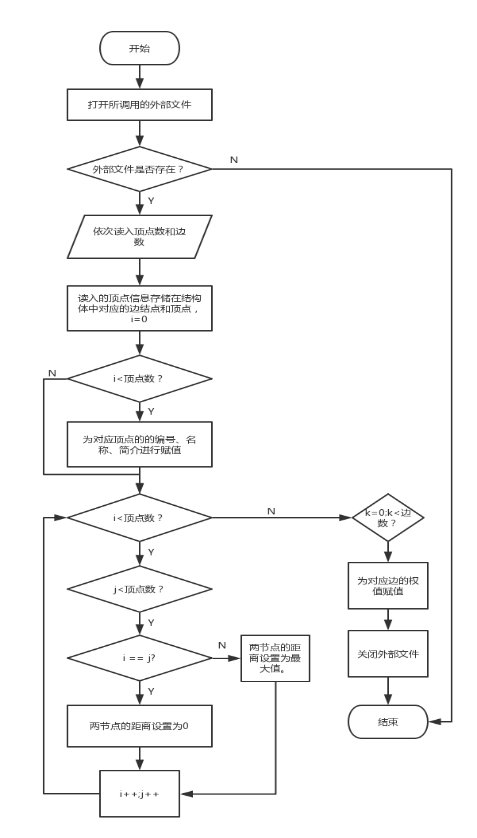
beauty beauties[Max]; //存储每个结点的信息

edgeNode edges[Max][Max]; //存储结点之间的路径长度

} GraphMatrix;

1. **各功能模块的详细设计及实现**
   1. **校园平面图的创建以及显示功能模块**
      1. 详细设计思想

使用邻接矩阵存储了校园平面图，将景点定义为结构体，结构体里面包含了景点的名称、编号和简介。边结构体中有边的长度。并且创建了地图可以查看地图大概。



* + 1. 功能模块核心代码

void create\_graph(GraphMatrix \*g, char \*s, int c){

int i;

int j;

int k;

int w;

FILE \*graph;

graph = fopen(s, "r");

if(graph){

fscanf(graph, "%d%d",&g->nodenum, &g->edgenum);

for(i = 0;i < g->nodenum; i++){

fscanf(graph, "%d", &g->beauties[i].num);

fscanf(graph, "%s", &g->beauties[i].name);

fscanf(graph, "%s", &g->beauties[i].introduction);

}

for(i = 0; i < g->nodenum; i++){

for(j = 0; j< g->nodenum; j++){

if(i == j){

g->edges[i][j].length = 0;

}

else{

g->edges[i][j].length = FINITY;

}

}

}

for(k = 0; k < g->edgenum; k++){

fscanf(graph, "%d%d%d", &i, &j, &w);

g->edges[i][j].length = w;

if(c == 0){

g->edges[j][i].length = w;

}

}

fclose(graph);

}

else{

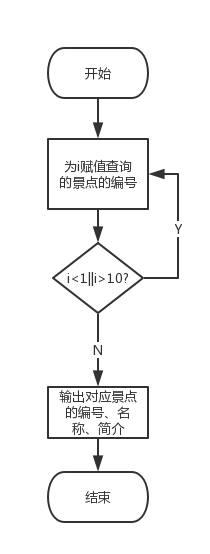
g->nodenum = 0;

}

}

* 1. **校园景点信息的查询功能模块**
     1. 详细设计思想

利用while循环判断输入的编号是否有误，如果符合条件，将该景点的编号、名称、景点介绍输出。否则，继续输入编号。



* + 1. 功能模块核心代码

void search\_graph(GraphMatrix\* g) {

int i;

printf("请输入您所需要查询的结点的编号：\n");

scanf("%d", &i);

getchar();

while (i < 1 || i > 10) {

printf("您所输入的信息有误，请重新输入！\n");

scanf("%d", &i);

getchar();

}

printf("结点编号为：\n%d\n", g->beauties[i - 1].num);

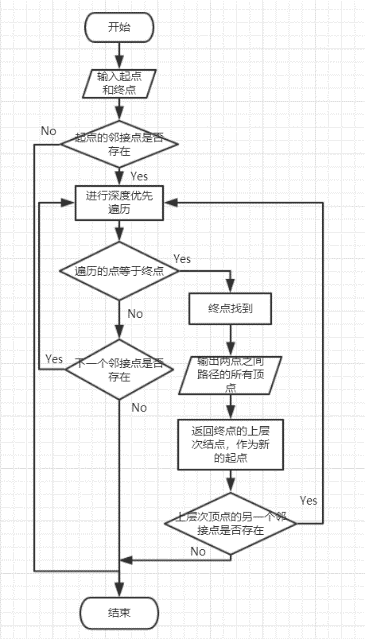
printf("结点名称为：\n%s\n", g->beauties[i - 1].name);

printf("结点简介如下：\n%s\n", g->beauties[i - 1].introduction);

}

* 1. **查询任意两景点的所有路径**
     1. 详细设计思想

在寻找两点之间的所有路径时，采用稍改动的深度优先遍历算法，首先输入起点和终点这两个结点，从起点开始进行深度优先遍历，利用递归的思想，把起点的下一个邻接点作为新的起点继续深度遍历，直到找到终点。找到终点之后，返回到终点的上一个结点，从上一个点的另外一个邻接点进行深度优先遍历，直到再次找到终点。之后再返回，直到所有的路径被找到。



* + 1. 功能模块核心代码

int visited[M];

int pre[M];

void dfspath(GraphMatrix \*g,int m,int n,int k)

{

int s;

if(pre[k]==n&&k<10)

{

for(s=0;s<k;s++)

printf("%s--->",g->beauties[pre[s]].name);

printf("%s",g->beauties[pre[s]].name);

putchar(10);

}

else

{

s=0;

while(s<g->nodenum)

{

if((g->edges[pre[k]][s].length<FINITY)&&(visited[s]==0))

{

visited[s]=1;

pre[k+1]=s;

dfspath(g,m,n,k+1);

visited[s]=0;

}

s++;

}

}

}

void printAllPath(GraphMatrix \*g)

{

int i,m,n;

printf("请输入你要查询的两个景点的编号:");

scanf("%d%d",&m,&n);

m--;n--;

pre[0]=m;

for(i=0;i<g->nodenum;i++)

visited[i]=0;

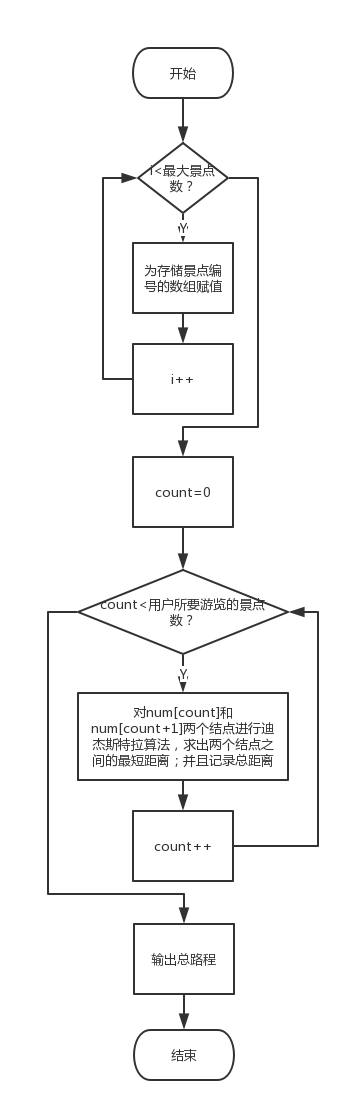
visited[m]=1;

dfspath(g,m,n,0);

}

* 1. **查询途径多个景点的最佳路径**
     1. 详细设计思想

用户依次输入节点编号，即游览顺序。根据迪杰斯特拉算法，算出第一个景点和第二个景点的最短路径，依次类推，算出多景点的最短路径。具体流程如下图所示：



* + 1. 功能模块代码实现

typedef int dist[M];

typedef int path[M];

int help (GraphMatrix g, path p, dist d, int i){

int pre;

int top = -1;

i--;

beauty st[M];

printf ("\n路程长度为:%7d, 路线为:", d[i]);

st[++top] = g.beauties[i];

pre = p[i];

while (pre != -1) {

st[++top] = g.beauties[pre];

pre = p[pre];

}

printf ("%s", st[top--].name);

while (top >= 0) {

printf(" ---> %s ", st[top--].name);

}

return d[i];

}

void travelBestPath(GraphMatrix g, path p, dist d){

int num[M];

int i;

int j;

int count = 0;

int m = 0;

label:printf("请按照您所需要游览的次序，输入景点编号：\n");

printf("注：输入“-1”即可停止选择。\n");

for(i = 0; i < M; i++){

menu();

printf("请输入您所要游览的第 %d 个景点：", i + 1);

scanf("%d", &num[i]);

system("cls");

if(num[i] == -1){

break;

}

if(num[i] < 1 || num[i] > g.nodenum){

printf("输入的数据错误！");

i--;

to\_menu();

}

}

// printf("%d", i);

j = i;

i = 0;

menu();

printf ("按照您输入的景点编号可以得出，您所要游览的景点依次为：\n");

printf ("%s ", g.beauties[num[i] - 1].name);

for (i = 1; i < j; i++){

printf("---> %s ", g.beauties[num[i] - 1].name);

}

printf("\n");

printf("-----------------------------------------------------------------------\n");

printf("请问您是否想要重新选择您的景点路线？输入“Yes”进行重置；输入“No”则不进行重置。\n");

char choice[M];

scanf("%s", choice);

if (strcmp(choice, "Yes") == 0){

system("cls");

menu();

goto label;

}

else{

system("cls");

menu();

int total\_distance = 0;

printf("\n-----------------------------------------------------------------------");

printf("\n经计算得，您的最佳旅游路线为:");

for (count = 0; count < j; count++){

dijkstra(g, num[count] , p, d);

m = count + 1;

if (num[m] == -1){

break;

}

printf("\n第 %d 步：", count + 1);

total\_distance += help(g, p, d, num[m] );

}

printf("\n-----------------------------------------------------------------------");

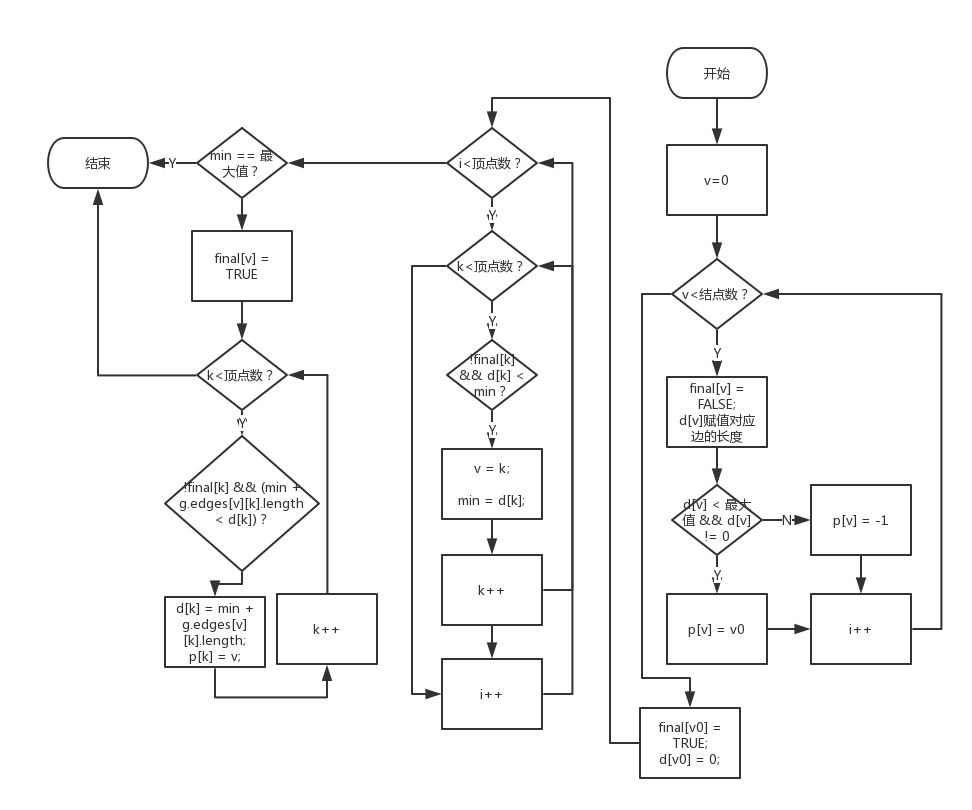
printf("\n您的旅游最佳路线的总路程长度为: %7d\n", total\_distance);

}

}

* 1. **查询任意两景点之间的最短路径**
     1. 详细设计思想

利用迪杰斯特拉算法根据需求求出用户所选择的起点到其余顶点的最短路径，然后设计一个函数仅输出用户所需要的终点对应的路径即可。



* + 1. 功能模块代码实现

typedef enum{FALSE,TRUE} boolean;

typedef int dist[M];

typedef int path[M];

void dijkstra(GraphMatrix g, int v0, path p, dist d) {

boolean final[M];

v0--;

int i, j, k, v, min, x;

//第一步初始化集合S与距离向量d

for (v = 0; v < g.nodenum; v++) {

final[v] = FALSE;

d[v] = g.edges[v0][v].length;

if (d[v] < FINITY && d[v] != 0) {

p[v] = v0;

}

else {

p[v] = -1;

}

}

final[v0] = TRUE;

d[v0] = 0;

//第二步依次找出n-1个结点加入S中

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

min = FINITY;

for (k = 0; k < g.nodenum; ++k) {

if (!final[k] && d[k] < min) {

v = k;

min = d[k];

}

}

if (min == FINITY) {

return;

}

// printf("\n%d---%d\n", g.beauties[v].num, min);

final[v] = TRUE;

//第三步修改S与V-S中各结点的距离

for (k = 0; k < g.nodenum; k++) {

if (!final[k] && (min + g.edges[v][k].length < d[k])) {

d[k] = min + g.edges[v][k].length;

p[k] = v;

}

}

}

}

void print\_g(GraphMatrix g, path p, dist d) {

int i, pre, top = -1;

beauty st[M];

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

printf("\nDistance:%7d, path:", d[i]);

st[++top] = g.beauties[i];

pre = p[i];

while (pre != -1) {

st[++top] = g.beauties[pre];

pre = p[pre];

}

printf("%s", st[top--].name);

while (top >= 0) {

printf(" ---> %s ", st[top--].name);

}

}

}

void shortPath(GraphMatrix g, path p, dist d){

int v0;

int i;

printf("please input the start point v0:\n");

scanf("%d", &v0);

printf("please input the end point i:\n");

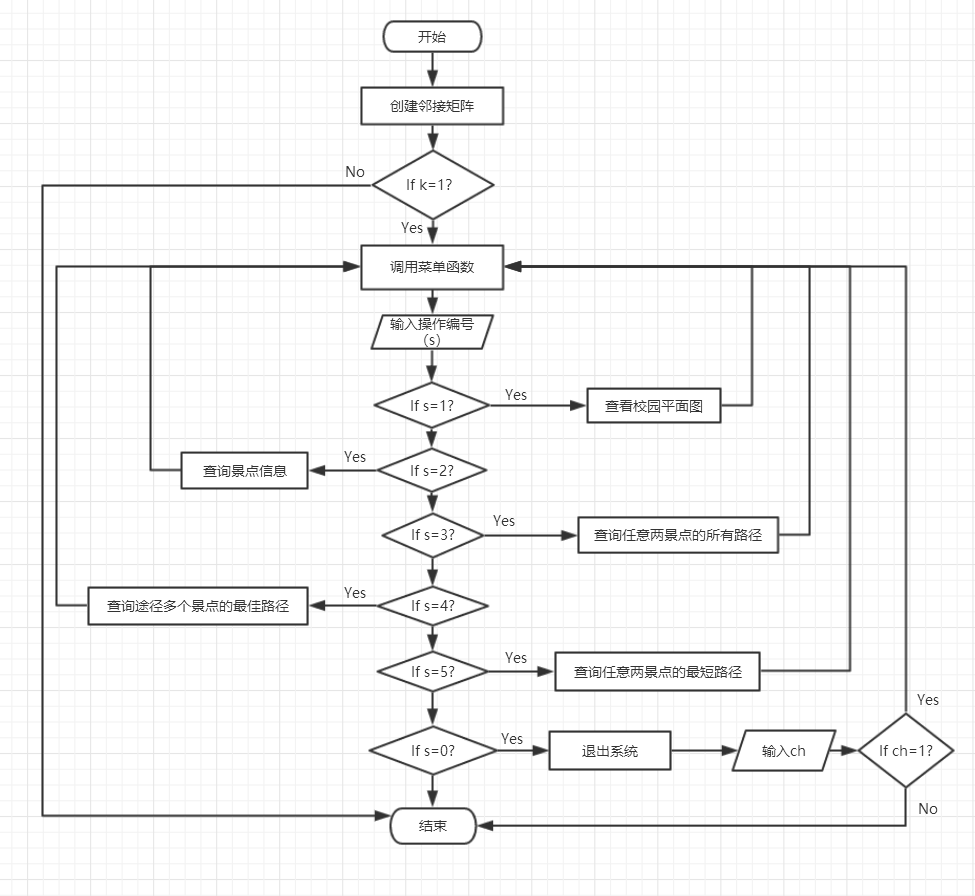
scanf("%d", &i);

dijkstra(g, v0, p, d);

help(g, p, d, i);}

* 1. **主函数界面设计**
     1. 详细设计思想

创建图的邻接矩阵，之后引用菜单meau（）函数，利用switch语句判断使用什么功能。在退出系统时，引用end（）函数，进行最后判断。



* + 1. 功能模块代码实现

int end()

{

int k=1;

int ch;

printf("确认退出吗?(0.退出 1.不退出):");

scanf("%d",&ch);

while(1)

{

if(ch==0||ch==1)

break;

else

{

printf("输入的数据不合理，请重新输入:");

scanf("%d",&ch);

}

}

if(ch==0)

{

k=0;

printf("谢谢使用本系统，欢迎下次光临!\n");

}

return k;

}

void menu() //主菜单

{

system("cls");

printf("--------------------------------------------------------\n");

printf(" 校园导航咨询系统 \n");

printf(" -made by Group 12\n");

printf("========================================================\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| 1 . 查看校园平面图 ||\n");

printf("|| 2 . 查询景点的信息 ||\n");

printf("|| 3 . 查询任意两景点的所有路径 ||\n");

printf("|| 4 . 查询途径多个景点的最佳路径 ||\n");

printf("|| 5 . 查询任意两景点的最短路径 ||\n");

printf("|| 0 . 退出本系统 ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("|| ||\n");

printf("========================================================\n");

printf("请输入要进行操作的编号：");

}

void to\_menu() //返回主菜单的函数

{

char c1,c2;

printf("\n\n\n按回车键返回主菜单...");

scanf("%c%c",&c2, &c2);

menu();

}

int main(int argc, char \*argv[]) {

GraphMatrix g;

path p;

dist d;

int s;

create\_graph(&g, "G13.txt", 0);

while(1)

{

menu();

scanf("%d", &s);

switch(s)

{

case 1: printMap();

system("cls");

break;

case 2: search\_graph(&g);

to\_menu();

system("cls");

break;

case 3: printAllPath(&g);

to\_menu();

break;

case 4: travelBestPath(g, p, d);

to\_menu();

system("cls");

break;

case 5: shortPath(g, p, d);

to\_menu();

system("cls");

break;

case 0:break;

}

}

return 0;

}

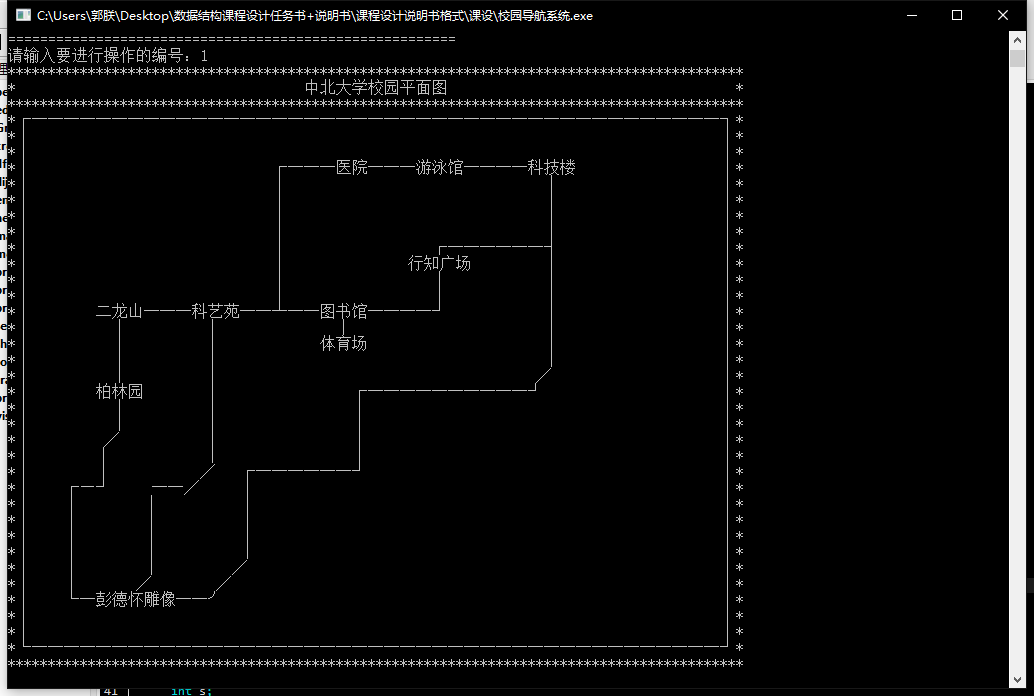
1. **分工情况**

郭 朕：结构体的建立、图的创建、多个景点最佳路径查询；

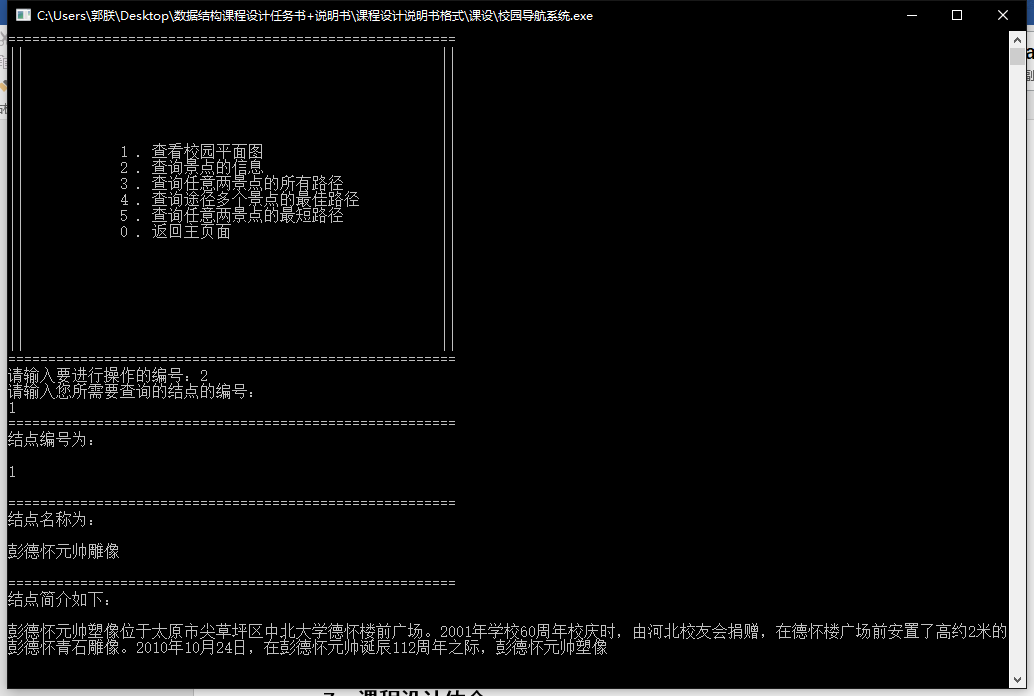
张永栋：主函数界面、任意两个景点的所有路径查询；

黄嘉灵：任意两个景点的最短路径、任意景点信息查询。

1. **代码运行情况**
   1. **校园平面图显示情况**



* 1. **校园景点信息查询情况**



* 1. **任意两景点所有路径查询情况**



* 1. **多个景点最佳路径查询情况**



* 1. **任意两景点最短路径查询情况**



* 1. **主函数界面显示情况**



1. **课程设计体会**

郭 朕：

第一次课程设计，总体感觉收获很大也有很多想法。首先，作为组长，要提前对项目所需要的基础有所了解，同时应该在项目开始之前就对项目的工作量以及内容有一个大概的估计。也就是说，要在真正开始设计之前就对整个项目的结构有一个较为清晰的认知，才能更好的分配任务和完成任务。其次，就是自己完成代码部分，第一次以小组为团队进行项目编程，在代码的整合部分出现了比较多的问题，改bug的时间要远远多于真正敲代码的时间，这里的问题应该是出在小组的讨论不够及时，不能及时的获取到组内其他成员的信息以及完成情况。最后，就是在项目设计的时候会遇到很多的自己难以解决的难题，学会灵活的使用搜索引擎，搜索自己所需要的知识点进行补足，也是非常重要的。希望，下一次课程设计的时候，这次的错误不会再犯。

张永栋：

在本次课程设计中，我对数据结构知识及应用有了更为生动的认识与感悟，同时也发现了自己知识面及编程能力的不足。对于自己所负责部分有较多的想法，但由于目前掌握知识不足，部分想法未能实现。课程设计期间，我也充分认识到搜索引擎及团队合作的重要性，许多未能解决的问题或是编程中出现的问题，利用小组讨论或是网上检索，总能激发灵感，成功解决问题。在之后的学习中，我要总结经验，再接再厉，学习专业知识，拓展自己的知识面，让本次课程设计中的错误不再出现，让好的想法完美高效实现！

黄嘉灵：

通过本次课程设计，让我更深入地了解数据结构的重要性，在上机过程中也发现了自己很多知识的不足，下来我会好好加强自己的知识储备和专业素养，为自己之后的学习打好基础！