

****

**J I A N G S U U N I V E R S I T Y**

**《数据结构与算法》课程设计**

|  |  |
| --- | --- |
| 学院名称： | 计算机科学与通信工程学院 |
| 专业班级： | 物联网工程21级 1 班 |
| 学生姓名： | \*\*\*\*\*\*\*\* |
| 学生学号： | \*\*\*\*\*\*\*\*\*\* |
| 指导教师： | 郑文怡 |

**2022年6月28日**

**一．设计要求**



以上是一份江大手绘地图，请你为2022级新生定制一个线路问询交互系统，请从地图中选择至少15个的江苏大学重要地点，其中需要包含校内公交站点和非公交站点，至少5条校车路线，这样可以设计出换乘路线，如果学校没有开通可以自行设计校车线路，例如：

京江教学楼→医学院→计算机楼→二号楼→五棵松

五棵松→教职工活动中心→职工医院

学生宿舍→计算机楼→京江学院

这样从京江教学楼到职工医院就可以在五棵松换乘；从宿舍到职工医院，可

以在计算机楼换乘；

时间、距离、费用等相关信息可以自行估计；

交通方式：步行、乘车两种；

1. 系统应该至少具有(但是不限于)以下基本功能：

管理员端：

 初始化：将所有相关信息存入文本，每次运行系统都要进行初始化操作；

 维护公交线路

1） 新增公交线路

2） 修改已有公交线路（修改部分站点信息）

3） 删除已有站点（同时应当修改该站点的公交线路信息）

4） 增加站点，添加到已有公交线路中

以上信息修改之后均需要输出信息以验证修改结果，并且将结果写回存储文件中。

学生端：

 公交线路查询

1） 输入地点，查询出该地点所经过的所有公交线路相关信息

2） 输出所有江大开设的校内公交路线；

 公交路线规划

输入起点和终点，输出所有的可达路线及花费的时间（包括从该点步行到公交车站、从公交站步行到终点的时间）

 最优路线规划

输入起点和终点，

1） 给出步行的最短距离路线

2） 给出时间最短的路线 （可以只坐公交，也可以步行+公交等，为了体现工作量，请自行设计线路，难易程度不受限制，但决定了课程设计的分数）

3） 给出换乘次数最少的路线；

4） 输出从五棵松出发到达各个地点的最短路线

 判断校内是否有环线

2、说明

菜单可以根据自己的功能设计完成，可以增加子菜单等；要求程序通过循环为用户显示，并让用户根据需求进行相应的操作，直到用户选择退出咨询系统位置；也可以根据需要增加其余功能（此部分为加分项，自行考虑和设计）

**二.问题分析和任务定义**

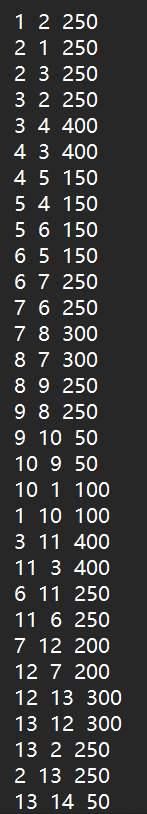
本次课程设计要求制作一个江苏大学校内交通交互问询系统，课程结束后，要求学生能够建立一个包含管理员端，学生端的双端问询系统，通过该系统，管理员用户可以通过该程序对校园内的公交站点以及公交线路进行添加和删除，学生可以通过该程序查询到所有的公共交通线路以及通过所有站点的公共交通线路，并可以使用该程序获得一个较好的路线规划。在设计该程序过程中，学生应能有效地利用本学期《数据结构与算法》课程所学的知识，并在原有知识上获得一些相应的突破。

**三.数据结构的选择和概要设计**

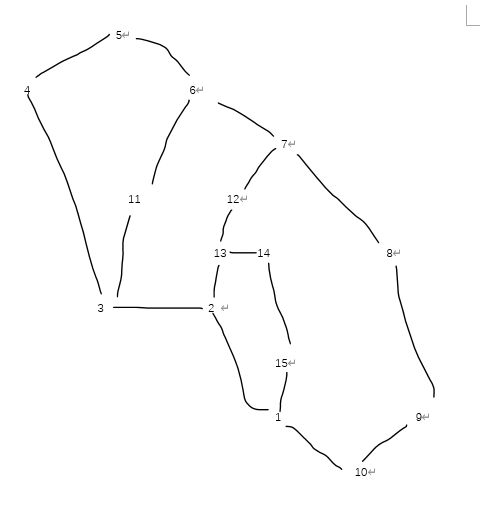
本程序将会用到数组，有向图（含AOV网）等结构，将会使用线性结构与非线性结构的结合，在线路的增删改查中，将会使用线性结构，在判断环路和最短路径时，将会使用图形结构，在程序实现中将使用迪杰斯特拉算法与拓扑排序等算法。

其中，迪杰斯特拉算法将用于有向图求解最短路径，拓扑排序将用于求解校内公交是否有环路。其他操作会使用到数组的部分知识。

事前准备：先计划出15个地标点和各个点之间通路的权值，如下图所示。



构建江苏大学地标地图。



**四.详细设计和编码**

**（1）初始化系统**

在进行初始化系统时，代码可以读入江苏大学所有地标名称，并读取校园公交路线，如果公交路线文件不合法或者数据出错，将自动清空并覆盖原文件，这样可以保证下面在读取公交线路的时候不会报错，增加了程序的稳定性。

以下为本部分实现的部分代码：

void chushihua()

{

FILE \*fp;

fp=fopen("公交路线.txt","r");

int sch[240];

int i;

for(i=0;i<240;i++)

{

sch[i]=0;

}

for(i=0;i<240;i++)

{

fscanf(fp,"%d",&sch[i]);

}

if(sch[0]!=-1)

{

fp=fopen("公交路线.txt","w");

fprintf(fp,"-1 ");

}

else ;

fclose(fp);

}

本部分在建立当初源于在一次对公交线路的输出时因文件数据错误导致输出乱码，为了提高程序的稳定性，特此建立本子函数。

**（2）新增公交线路**

如何实现新增公交线路呢？当时由于我们选择公交线路存储的结构为数组，我认为此问题可以由繁化简，直接将数据写入数组，在文件末尾直接追加这个数组就可以实现。当输入-1的时候，通过if判断结束输入，然后追加。

以下为本部分实现的部分代码：

void addline()

{

FILE \*fp1;

fp1=fopen("公交路线.txt","a+");

int i,site;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入公交路线（以输入-1作为结束）：");

for(i=0;i<15;i++)

{

scanf("%d",&site);

if(site>15||site<-1)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

break;

}

else if(site==-1)

{

fprintf(fp1,"-1 ");//-1表示输入结束。

break;

}

else fprintf(fp1," %d ",site);//循环输入站点。

} system("cls");

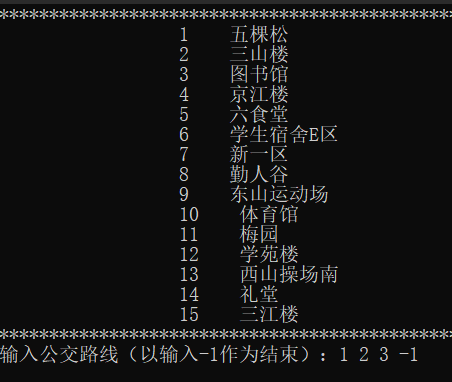
printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 程序完成 \* \n");

fclose(fp1);

}

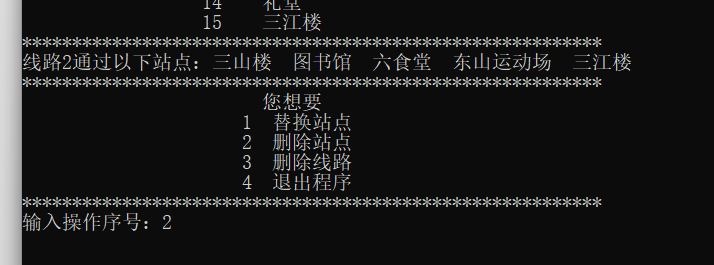
本部分在写的时候确实也遇到了一些问题，一开始我想直接用数组覆盖写入，发现覆盖写入的话程序会很繁琐，于是我将文件改为追加写入（a+），有效地解决了问题。



通过操作，我们可以看到线路追加成功。



**（3）修改指定线路的信息，包括换站点，删站点或删线路**



通过此功能，我们可以选择替换某条线路中的站点或者删除某条线路的站点，也可以将整条线路删除。原理非常的简单，就是在文件中查找到对应的线路，然后将线路中的站点序号导入数组，将数组中的指定序号替换或置空，或将整条线路置空，以达到替换/删除操作，最后将修改后的数组再次写入文件，达到修改线路的目的。

以下为本部分实现的部分代码：

void changeexist()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("您想要修改的线路是：");

int changeline;

scanf("%d",&changeline);system("cls");

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;//空数据默认为0以避免输出时输出地址影响观感。

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);//将文件里的数据导入数组。

n=0;

for(i=0;i<140;i++)

{

if(bus[i]==-1)

n++;

if(n==changeline)

{

break;

}

}

int bus2[15]={0};

int remem=i+1;

int k=0;

for(j=i+1;bus[j]!=-1;j++)

{

bus2[k]=bus[j];

k++;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("线路%d通过以下站点：",changeline);

for(i=0;bus2[i]!=0;i++)

{

print2(bus2[i]);

}

printf("\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 您想要 \n");

printf(" 1 替换站点 \n");

printf(" 2 删除站点 \n");

printf(" 3 删除线路 \n");

printf(" 4 退出程序 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入操作序号：");

int donum;

scanf("%d",&donum);system("cls");

if(donum==1)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("您想要替换掉的站点为：");

int wantchan,afterchan;

scanf("%d",&wantchan);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("替换后的站点为：");

scanf("%d",&afterchan); system("cls");

if(afterchan>15||afterchan<1||wantchan>15||wantchan<1) goto loop;

for(i=0;i<15;i++)

{

if(bus2[i]==wantchan)

bus2[i]=afterchan;

}

k=0;

for(j=remem;bus2[k]!=0;j++)

{

bus[j]=bus2[k];

k++;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 变更成功 \* \n");

help(bus);

}

else if(donum==2)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("您想要删除掉的站点为：");

int wantchan,afterchan;

scanf("%d",&wantchan); system("cls");

if(wantchan>15||wantchan<1) goto loop;

afterchan=-2;

for(i=0;i<15;i++)

{

if(bus2[i]==wantchan)

bus2[i]=afterchan;

}

k=0;

for(j=remem;bus2[k]!=0;j++)

{

bus[j]=bus2[k];

k++;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 变更成功 \* \n");

help(bus);

}

else if(donum==3)

{

for(j=remem;bus[j]!=-1;j++)

{

bus[j]=0;

}

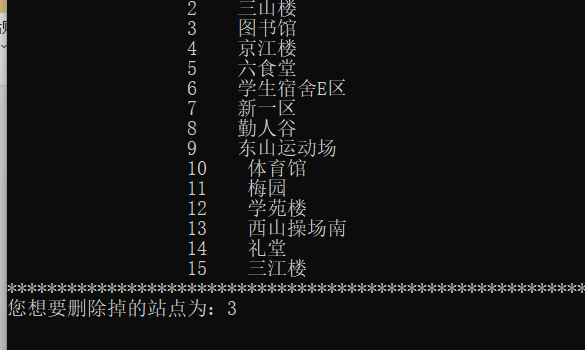
bus[remem-1]=0;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 变更成功 \* \n");

help(bus);

}



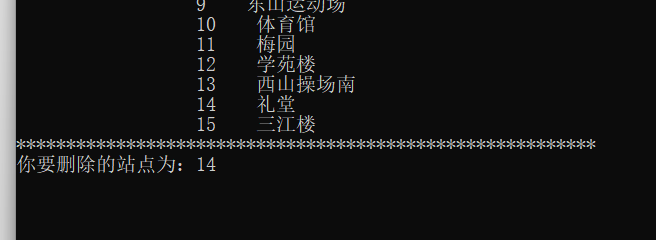
以上为从二号线中删掉图书馆站。可以看到，该站成功被删除。



本部分在编写过程中没有太大困难，直接利用数组的功能，减少了很多工作量。

**（4）删除有公交经过的某站点，使其不经过公交。**

将文件导入数组，在数组中找到要删除的站点，置空该数据，再将数组覆盖写入源文件。，达到删除站点的目的。



以下为本部分实现的部分代码：

void deletesite()

{

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;//空数据默认为0以避免输出时输出地址影响观感。

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);//将文件里的数据导入数组。

int deletesite;

fp1=fopen("公交路线.txt","w");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n你要删除的站点为：");

scanf("%d",&deletesite);system("cls");

if(deletesite<=15&&deletesite>0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 删除成功 \* \n");

}

if(deletesite>15||deletesite<=0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

}

for(i=0;i<240;i++)

{

if(bus[i]==deletesite)

bus[i]=0;//将要删除的站点置空。

}

for(i=0;i<240;i++)

if(bus[i]!=0)

fprintf(fp1,"%d ",bus[i]);

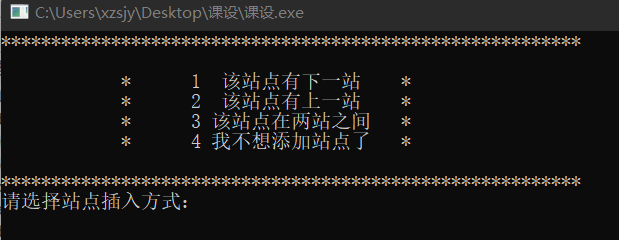
fclose(fp1);

}

本部分实现同上面的删除站点，所以实现起来不难。

**（5）增加公交站点**

从文件中读入数据并导入数组，在数组中找到你要添加站点的上一站或下一站或指定要插入站点的位置（某两站之间），将数据插入其中，并使其后面的数据后移就可以了。最后将改变后的数组写入文件以完成插入站点的操作。



以下为本部分实现的部分代码：

//下一站增站法。 找到要增加站点的下一站，在前面加上它。

void nextsta(int bus[],int creatsite)

{

start:

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("该站点下一站为：");

int next;

scanf("%d",&next);system("cls");//先给出增加站点的下一站（next节点）。

if(next>15||next<0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

goto start;

}

int i,j;

for(i=0;i<240;i++)

{

if(bus[i]==next)//找到我们输入的下一站。

{

for(j=239;j>=i;j--)

{

bus[j+1]=bus[j];//将下一站之后的站数据全部后移。

}

bus[i]=creatsite;//要增加的站点赋给i。

i++;

}

}

FILE \*fp1;

fp1=fopen("公交路线.txt","w");

for(i=0;i<240;i++)

if(bus[i]!=0&&bus[i]!=bus[i+1]) //同理，0为空，若原线路有该站点将不写入文件。

fprintf(fp1,"%d ",bus[i]);

fclose(fp1);

}

//上一站增站法。 原理同上。

void frontsta(int bus[],int creatsite)

{

start:

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("该站点上一站为：");

int front;

scanf("%d",&front);system("cls");

if(front>15||front<0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

goto start;

}

int i,j;

for(i=1;i<240;i++)

{

if(bus[i-1]==front)//找到我们输入的上一站。

{

for(j=239;j>=i;j--)

{

bus[j+1]=bus[j];//将上一站之后的站数据全部后移。

}

bus[i]=creatsite;//要增加的站点赋给i。

i++;

}

}

FILE \*fp1;

fp1=fopen("公交路线.txt","w");

for(i=0;i<240;i++)

if(bus[i]!=0&&bus[i]!=bus[i+1])

fprintf(fp1,"%d ",bus[i]);

fclose(fp1);

}

//两端增站法。原理同上。

void betweensta(int bus[],int creatsite)

{

start:

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

int front,next;

printf("该站点上一站为：");

scanf("%d",&front);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("该站点下一站为：");

scanf("%d",&next);system("cls");

if(front>15||front<0||next>15||next<0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

goto start;

}

int i,j;

for(i=1;i<240;i++)

{

if(bus[i-1]==front&&bus[i]==next)//找到我们输入的上一站和下一站。

{

for(j=239;j>=i;j--)

{

bus[j+1]=bus[j];//将上一站之后的站数据全部后移。

}

bus[i]=creatsite;//要增加的站点赋给i。

i++;

}

}

FILE \*fp1;

fp1=fopen("公交路线.txt","w");

for(i=0;i<240;i++)

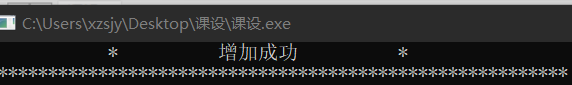
if(bus[i]!=0&&bus[i]!=bus[i+1])

fprintf(fp1,"%d ",bus[i]);

fclose(fp1);

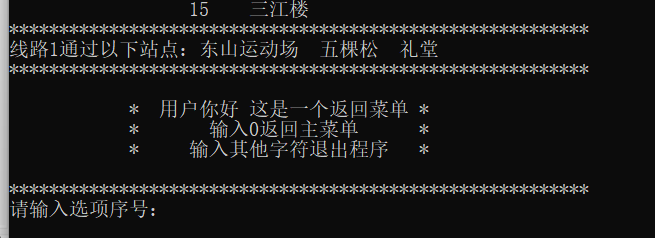
}

本部分的实现还是比较简单的，利用了数组查找和插入的原理，所以没有遇到太多问题。就是实现将数组中数据后移的时候要注意数组的大小防止末位数据丢失。



**（6）公交线路查询之查询指定线路经过的站点**

将文件中的数据导入数组，每遇到一次-1就增加一次线路序号，并获取经过站点的序号，再根据序号打印出地址。



以下为本部分实现的部分代码：

void chaxun1()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("您想要查询的线路是：");

int cxline;

scanf("%d",&cxline);system("cls");

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;//空数据默认为0以避免输出时输出地址影响观感。

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);//将文件里的数据导入数组。

n=0;

for(i=0;i<140;i++)

{

if(bus[i]==-1)

n++;

if(n==cxline)

{

break;

}

}

int bus2[15]={0};

int remem=i+1;

int k=0;

for(j=i+1;bus[j]!=-1;j++)

{

bus2[k]=bus[j];

k++;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("线路%d通过以下站点：",cxline);

for(i=0;bus2[i]!=0;i++)

{

print2(bus2[i]);

}

printf("\n");

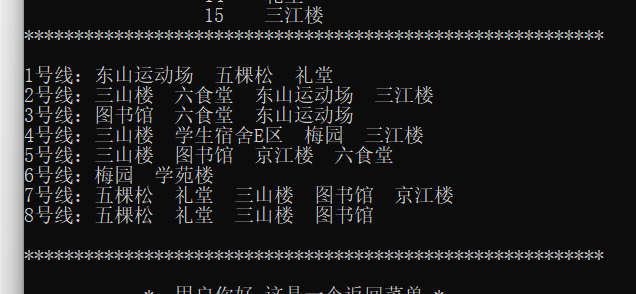
fclose(fp1);

}

本部分最初时不能显示站点名称只能显示序号，后经过老师指导，已经可以显示经过的站点名称了。改进比较顺利。

**（7）公交线路查询之输出所有开行的线路**

本部分同以上输出指定线路通过的站点，只不过不再对线路进行限制，原理和上面也是一样的。



以下为本部分实现的部分代码：

void chaxun2()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

int bus[240];

FILE \*fp1;

int i;

fp1=fopen("公交路线.txt","r");

for(i=0;i<240;i++)

bus[i]=0;

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);

int m=1;

for(i=0;i<240;i++)

{

if(bus[i]==-1&&bus[i+1]!=0)

{

printf("\n%d号线：",m);

m++;

}

else if(bus[i]!=0&&bus[i+1]!=0)

print2(bus[i]);

else

{

printf("\n\n");

break;

}

}

fclose(fp1);

}

本部分也同（6）一样进行了改进。

**（8）公交线路查询之查询经过某站点的公交线路**

构建站点查询子函数，该子函数读取公交路线文件并将其计入数组，在数组中，函数可以统计你要查询的站点，每出现一次-1公交线路序号就加一以完成线路的统计。

以下为本部分实现的部分代码：

void site()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

int bus[240];

FILE \*fp1;

int i,t=0;

fp1=fopen("公交路线.txt","r");

for(i=0;i<240;i++)

bus[i]=0;

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);

int m=1;

int search;

printf("你要查询的站点是：");

scanf("%d",&search);system("cls");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("为您查询到该站点通过以下线路:");

for(i=0;i<240;i++)

{

if(bus[i]==-1&&bus[i+1]!=0)

m++;

if(bus[i]==search)

{

printf(" %d号线 ",m-1);

t++;

}

}

if(t==0)

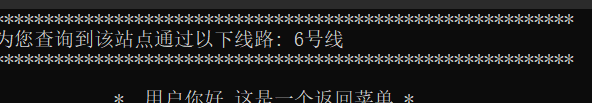
printf("没有线路通过站点！");

printf("\n");

fclose(fp1);

}

查询经过学苑楼的公交线路：



同样，用的数组查找的原理，难度不是很大。

**（9）公交路线规划：找到两点间的所有路线（仅步行）**

从文件中读取读取我们地图的邻接表，并构建一个图。

typedef struct

{

int vexs[MAXVEX];

int arc[MAXVEX][MAXVEX];

int numNodes;

int numEdges;

}Mgraph;

然后，通过遍历，找到所有能通向目的地的线路，已经经过的点从途中删掉。返回上一个有支路的点，继续进行遍历。

以下为本部分实现的部分代码：

//建立图的子函数，将起点，终点都存入G.arc里，便于直观查看。

int initgraph()

{

int i,j;

for(i=0;i<=15;i++)

for(j=0;j<=15;j++)

G.arc[i][j]=Infinity;

for(i=0;i<=15;i++)

G.arc[i][i]=0;

FILE \*fp;

int m;

fp=fopen("地标缓冲表.txt","r") ;

for(m=0;m<32662;m++)

fscanf(fp,"%d%d%d",&i,&j,&G.arc[i][j]);

fclose(fp);

return ;

}

//打印每一条路径的子函数，每遍历到一个点，就接着往下找没有遍历过的点找到终点为止，然后返回最初一个拥有两个以上可以继续遍历的点，继续遍历直到完成所有遍历。

int path(int m,int n,int k);

//输出所有路径的主程序，通过这个函数，我们可以输入起止位置并调用path子函数完成所有路径的遍历。

int allpath()

{ int k,i,j,m,n;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入您想要查询的两个地标的编号：");

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

if(m>=0&&m<=15&&n>=0&&n<=15)

{d[0]=m;//存储路径起点m（int d[]数组是全局变量）

for(k=0;k<15;k++)//全部顶点访问标志初值设为0

visited[k]=0;printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");printf("\n");

visited[m]=1;//第m个顶点访问标志设置为1

path(m,n,0);

}

else

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 输入有误 \n");

} //调用程序。k=1，对应起点d[1]==m。k为d[]数组下标

return 1;

}

//打印每一条路径的子函数，每遍历到一个点，就接着往下找没有遍历过的点找到终点为止，然后返回最初一个拥有两个以上可以继续遍历的点，继续遍历直到完成所有遍历。

int path(int m,int n,int k)

{ int s,t,time=0; //t用于存放路径上下一个顶点对应的d[]数组元素的下标

t=k+1;

if(d[k]==n&&k<16) //若d[k]是终点n且地点个数<=16,则输出该路径

{ for(s=0;s<k;s++) //计算路径长度

{

time=time+G.arc[d[s]][d[s+1]];

} //打印路径长度小于定长的路径

{ for(s=0;s<k;s++) //输出该路径。s=0时为起点m

{

printf("%d->",d[s]);

}

printf("%d ",d[s]);

printf("（时间花费%d分钟）",time/50); //输出最后一个地点（即顶点n的序号，此时s==k）

printf("\n\n");

}

}

else

{ s=1;

while(s<=15)//从第m个顶点，试探所有顶点是否有路径

{ if((G.arc[d[k]][s]<Infinity)&&(visited[s]==0))//顶点m到顶点s有边(权值不为无穷大)，且未被访问

{ visited[s]=1;

d[k+1]=s;//存储顶点编号

path(m,n,t);

visited[s]=0;//将找到的路径上顶点的访问标志重新设置为0，以用于试探新的路径

}

s++;//试探从下一个顶点s开始是否有到终点的路径

}

}

}

//调取程序的子函数，用处不是很大也可以分开调取。

int allpathtest()

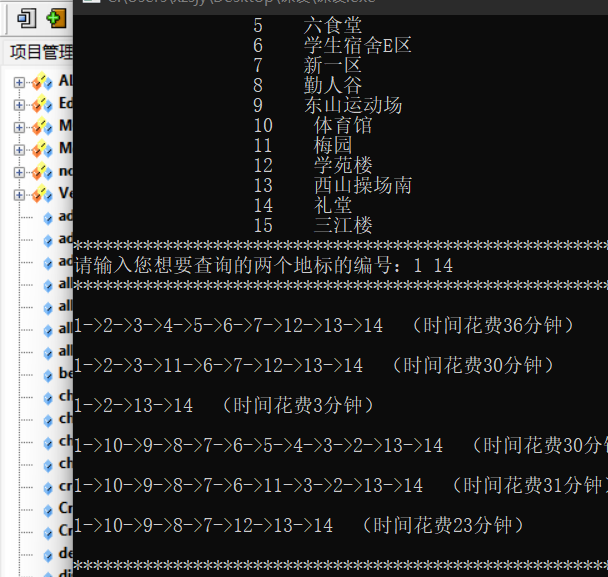
{

initgraph();

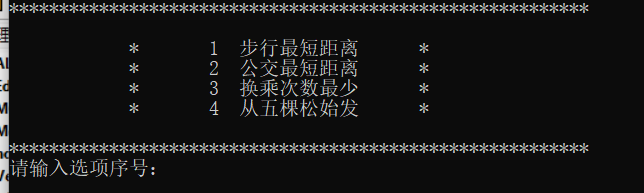
allpath();

return 0;

}



此功能实现还是比较困难的，在开始编写时因为找不好含支路节点的标记出错了许多次，后来改善了标记，问题得到了解决。

另外，本函数仍有不完善的地方，本函数无法输出相邻两点作为起止点的路径，原因我找到了是因为下一个点作为终点的时候会自动返回上一个点无法输出。

**（10）最优路线规划：找到两点间步行的最短距离并输出距离**

本算法采用迪杰斯特拉算法，将所有顶点分为已标记点和未标记点两个集合，从起始点开始，不断在未标记点中寻找距离起始点路径最短的顶点，并将其标记，直到所有顶点都被标记为止。

以下为本部分实现的部分代码：

void displayPath(int dist[],int path[],MGraph \*G,VertexType start,VertexType final)

{

int i,k,j=0;

int temp[VertexMax];//临时数组

VertexType target;//目标地点

int loc=0;

for(i=0;i<VertexMax;i++)

temp[i]=-1;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

//最短路径

for(i=0;i<G->vexnum;i++)

{

loc=i;//i每加一就对loc重置一遍。

j=0;

while(loc!=-1)

{//将loc计入数组以统计遍历过的地址。

temp[j]=loc;

loc=path[loc];

j++;

}

if(j-1==0&&G->Vertex[temp[j-1]]==start)//如果当前地址到起点是最短的，那么就可以输出当前点。

{

if(final==0||final==i+1)

{//合法的话就输出起点终点以便让用户看到。

printf("\t");

printf("始发：%d 终到：%d\n\t%d为起始点!",start,G->Vertex[i],G->Vertex[temp[j-1]]); printf("\n");

}

}

else if(j-1==0&&G->Vertex[temp[j-1]]!=start)

{//不合法的话就弹出警告。

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d\n\t%d不可达%d",start,G->Vertex[i],start,G->Vertex[temp[j-1]]); printf("\n");

}

}

else

{//如果是终点那么就可以结束循环了。

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d \n",start,G->Vertex[i]);

int clever=j-1;

printf("\t");

printf("最优线路：");

for(j=j-1;j>=0;j--)

{

if(j!=clever)printf("->");

printf("%d",G->Vertex[temp[j]]);

}

printf("(总路程长度:%d)",dist[i]); printf("\n");

}

}

for(k=0;k<20;k++)

temp[k]=-1;

}

}

//找最近路径的子函数，不断调用此子函数以找到最短路径。

int FindMinDist(int dist[],int s[],int vexnum)

{

int i;

int loc;

int min=MaxInt+1;

for(i=0;i<vexnum;i++)

{

if(s[i]==0)//只对s[i]=0的顶点进行查找

{

if(dist[i]<min)//如果找到的路径比当前遍历过的最近路径还近，就替换它直到最短。

{

min=dist[i];

loc=i;

}

}

}

return loc;//返回dist中最小元素的下标

}

void ShortestPath\_Dijkstra(MGraph \*G,VertexType start,VertexType final)

{

int i,j,num;

int loc;

int min;

int dist[VertexMax];//最短路径长度数组

int path[VertexMax];//最短路径数组

int s[VertexMax];//代表集合S（1代表该顶点已处理，属于集合S；0代表该顶点未处理，不属于集合S，属于集合V-S）

//1.初始化dist和path数组

loc=LocateVex(G,start);//获取源点的下标位置

for(i=0;i<G->vexnum;i++)

{

dist[i]=G->AdjMatrix[loc][i];

if(dist[i]!=MaxInt)

{

path[i]=loc;//标记存入i

}

else

{

path[i]=-1;

}

}

//2.初始化S数组（s数组：代表集合S，1代表该元素属于集合S(已处理的顶点)，0该元素不属于集合S(未处理的顶点)）

for(i=0;i<G->vexnum;i++)

{

s[i]=0;

}

s[loc]=1;//代表起始点(源点)以处理完毕

num=1;

while(num<G->vexnum)

{

min=FindMinDist(dist,s,G->vexnum);//在dist数组中查找其对应s[i]=0，即未处理的最小值元素

s[min]=1;//将找到的最短边所对应的的顶点加入集合S

for(i=0;i<G->vexnum;i++)//加入新的顶点后，更新dist和path数组

{

if((s[i]==0)&&(dist[i]>dist[min]+G->AdjMatrix[min][i]))

{

dist[i]=dist[min]+G->AdjMatrix[min][i];

path[i]=min;//min->i

}

}

num++;

}

displayPath(dist,path,G,start,final);//展示dist数组、path数组及最短路径

}

//找最短路径，下面的子函数应该都讲过了。

void zuiduanlujing()

{

MGraph G;

VertexType start,final;

CreateDN(&G);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入起始点：");

scanf(" %d",&start);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入终到点：");

scanf(" %d",&final);

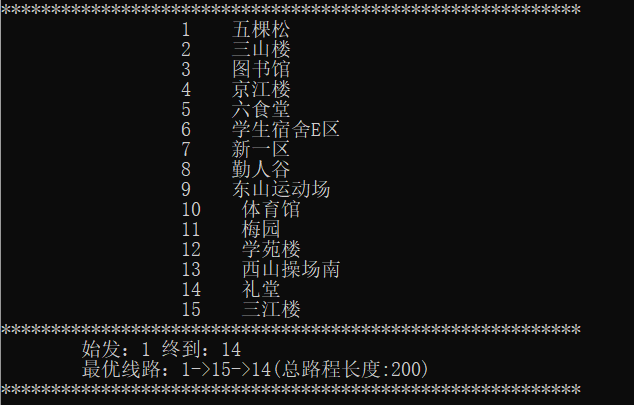
system("cls");

ShortestPath\_Dijkstra(&G,start,final);

}

该部分使用了之前上机作业的迪杰斯特拉算法的代码，因此并没有花费太多时间，只要每次找到离原来的图最近的节点就行了。

下面是从“五棵松”到“礼堂”的最短路径测试。



**（11）最优路线规划：找到两点间乘坐公交最短时间的路线并输出时间**

我的设定是公交速度为步行的5倍，在行程途中，凡是两点之间有公交的节点权值除以五再除以步行速度就是两点之间的公交时间。

先读取公交路线文件，将两点之间有公交的节点输入到表里，然后再根据表的内容从图中查找两点之间有公交的点位，对这样两点之间的权值除以五，再合并原有图的文件，使用最短路径求解（此时有公交经过的两点权值已经除以五），即找到两点之间最快的通行方法。

以下为本部分实现的部分代码：

void hangbianbiao()

{

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;//空数据默认为0以避免输出时输出地址影响观感。

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);//将文件里的数据导入数组。

int busplus[240][3];

for(i=0;i<240;i++)

for(j=0;j<2;j++)

busplus[i][j]=0;

for(i=0;i<240;i++)

busplus[i][2]=5;

j=0;

for(i=0;i<240;i++)

{

if(bus[i]!=-1&&bus[i]!=0&&bus[i+1]!=-1&&bus[i+1]!=0)

{

busplus[j][0]=bus[i];

busplus[j][1]=bus[i+1];

j++;

}

}

for(i=0;i<240;i++)

for(j=0;j<2;j++)

{

if(busplus[i][j]==0)

busplus[i][2]=0;

}

fp1=fopen("公交表.txt","w");

for(i=0;i<240;i++)

{

for(j=0;j<3;j++)

{

if(busplus[i][j]!=0)

fprintf(fp1,"%d ",busplus[i][j]);

}

fprintf(fp1,"\n");

}

fclose(fp1);

}

//将变成表的公交线路与学校地图表作对比，如果两点之间有公交，则可以自动坐公交。

void duibi()

{

FILE \*fp01;

fp01=fopen("江苏大学地标表.txt","r");

FILE \*fp02;

fp02=fopen("公交表.txt","r");

int dibiao[240][3];

int gongjiao[240][3];

int i,j;

for(i=0;i<240;i++)

for(j=0;j<3;j++)

{

dibiao[i][j]=0;

gongjiao[i][j]=0;

}

for(i=0;i<240;i++)

for(j=0;j<3;j++)

{

fscanf(fp01,"%d",&dibiao[i][j]);

}

for(i=0;i<240;i++)

for(j=0;j<3;j++)

{

fscanf(fp02,"%d",&gongjiao[i][j]);

}

for(i=0;i<240;i++)

for(j=0;j<240;j++)

{

if(i!=j&&gongjiao[i][0]==gongjiao[j][0]&&gongjiao[i][1]==gongjiao[j][1])

{

gongjiao[i][0]=0;

gongjiao[i][1]=0;

gongjiao[i][2]=1;}

}

for(i=0;i<240;i++)

for(j=0;j<240;j++)

{

if((dibiao[i][0]==gongjiao[j][0]&&dibiao[i][1]==gongjiao[j][1])&&(dibiao[i][0]!=0&&gongjiao[j][0]!=0))

dibiao[i][2]=(dibiao[i][2]/gongjiao[j][2]);

else dibiao[i][2]==dibiao[i][2];

}

FILE \*fp03;

fp03=fopen("地标缓冲表.txt","w+");

for (i=0;i<240;i++)

{

for(j=0;j<3;j++)

{

if(dibiao[i][j]!=0)

fprintf(fp03,"%d ",dibiao[i][j]);

}

fprintf(fp03,"\n");

}

fclose(fp01);

fclose(fp02);

fclose(fp03);

}

//以有公交线路的地图再次寻找最短路径（两点之间有公交就相当于两点之间变短）

void CreateDN2(MGraph \*G)

{

int i,j;

//1.输入顶点数和边数

G->vexnum=15;

G->arcnum=36;

//2.输入顶点元素

for(i=0;i<15;i++)

G->Vertex[i]=i+1;

//3.矩阵初始化

for(i=0;i<G->vexnum;i++)

for(j=0;j<G->vexnum;j++)

{

G->AdjMatrix[i][j]=MaxInt;

}

//4.构建邻接矩阵

int n,m;

VertexType v1,v2;

int w;//v1->v2的权值

FILE \*fp1;

fp1=fopen("地标缓冲表.txt","r");

for(i=0;i<G->arcnum;i++)

{

fscanf(fp1," %d%d%d",&v1,&v2,&w);

n=LocateVex(G,v1); //获取v1所对应的在Vertex数组中的坐标

m=LocateVex(G,v2); //获取v2所对应的在Vertex数组中的坐标

if(n==-1||m==-1)

{

printf("NO This Vertex!\n");

return;

}

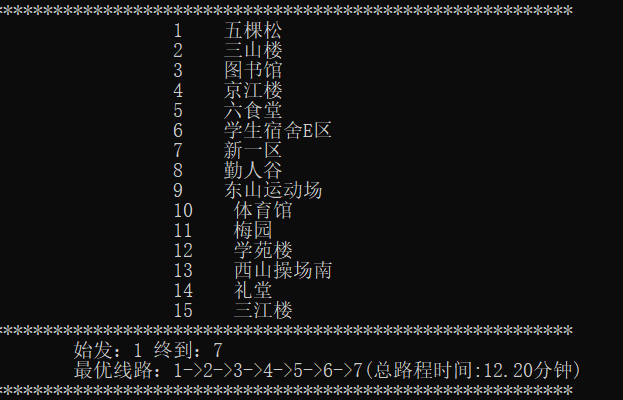
G->AdjMatrix[n][m]=w;

}

fclose(fp1);

}

以下为五棵松到新一区步行+公交的测试结果。



完成此功能设计的过程确实遇到了困难，最初设计时没有办法将公交与步行结合到一起计算，通过咨询学长，我简化了步行与公交结合的算法，于是有了步行与公交速度倍数的关系。

**（12）最优路线规划：找到两点间乘坐公交换乘次数最少的路线并输出换乘次数**

此功能通过计算途径换乘站的次数完成，若乘坐公交经过换乘站，则换乘次数加一，其中换乘站是指经过公交线路大于1的站点。

以下为本部分实现的部分代码：

void huanchengzhan()

{

FILE \*fp1;

FILE \*fp2;

fp1=fopen("公交路线.txt","r");

fp2=fopen("换乘表.txt","w");

fprintf(fp2," ");

fp2=fopen("换乘表.txt","w");

int hcb[240]={0};

int i;

for(i=0;i<240;i++)

{

fscanf(fp1,"%d",&hcb[i]);

}

int hcb2[240]={0};

for(i=0;i<240;i++)

{

hcb2[i]=hcb[i];

}

int j;

int hcb3[240]={0};

int k=0;

for(i=0;i<240;i++)

for(j=0;j<240;j++)

{

if(i==j) j++;

if(hcb[i]==hcb2[j]&&hcb[i]!=0&&hcb[i]!=-1)

{

hcb3[k]=hcb[i];

k++;

}

}

for(i=0;i<240;i++)

for(j=0;j<240;j++)

{

if(hcb3[i]==hcb3[j]&&i!=j)

hcb3[i]=0;

}

for(i=0;i<240;i++)

{

if(hcb3[i]!=0)

fprintf(fp2,"%d ",hcb3[i]);

}

fclose(fp1);

fclose(fp2);

}

//打印每一条路径的子函数，每遍历到一个点，就接着往下找没有遍历过的点找到终点为止，然后返回最初一个拥有两个以上可以继续遍历的点，继续遍历直到完成所有遍历。

int path22(int m,int n,int k);

//输出所有路径的主程序，通过这个函数，我们可以输入起止位置并调用path子函数完成所有路径的遍历。

//注意：本函数不完善没法输出相邻两点作为起止点的路径，原因我找到了是因为下一个点作为终点的时候会自动返回上一个点无法输出。

int allpath22()

{ int k,i,j,m,n;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入您想要查询的两个地标的编号：");

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

if(m>=0&&m<=15&&n>=0&&n<=15)

{d[0]=m;//存储路径起点m（int d[]数组是全局变量）

for(k=0;k<15;k++)//全部顶点访问标志初值设为0

visited[k]=0;printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");printf("\n");

visited[m]=1;//第m个顶点访问标志设置为1

path22(m,n,0);

}

else

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 输入有误 \n");

} //调用程序。k=1，对应起点d[1]==m。k为d[]数组下标

return 1;

}

//打印每一条路径的子函数，每遍历到一个点，就接着往下找没有遍历过的点找到终点为止，然后返回最初一个拥有两个以上可以继续遍历的点，继续遍历直到完成所有遍历。

int path22(int m,int n,int k)

{ int s,t,time=0; //t用于存放路径上下一个顶点对应的d[]数组元素的下标

t=k+1;

FILE \*fp;int i;

fp=fopen("换乘表.txt","r");

int hcb[240]={0};

for(i=0;i<240;i++)

{

fscanf(fp,"%d",&hcb[i]);

}

int temp=0;

if(d[k]==n&&k<16) //若d[k]是终点n且地点个数<=16,则输出该路径

{ for(s=0;s<k;s++) //计算路径长度

{

time=time+G.arc[d[s]][d[s+1]];

}

//打印路径长度小于定长的路径

{ for(s=0;s<k;s++) //输出该路径。s=0时为起点m

{

printf("%d->",d[s]);

for(i=0;i<240;i++)

{

if(d[s]==hcb[i])

{

temp++;break;}

}

}

printf("%d ",d[s]);

printf("（时间花费%d分钟,共换乘%d次）",time/50,temp); //输出最后一个地点（即顶点n的序号，此时s==k）

printf("\n\n");

}

}

else

{ s=1;

while(s<=15)//从第m个顶点，试探所有顶点是否有路径

{ if((G.arc[d[k]][s]<Infinity)&&(visited[s]==0))//顶点m到顶点s有边(权值不为无穷大)，且未被访问

{ visited[s]=1;

d[k+1]=s;//存储顶点编号

path22(m,n,t);

visited[s]=0;//将找到的路径上顶点的访问标志重新设置为0，以用于试探新的路径

}

s++;//试探从下一个顶点s开始是否有到终点的路径

}

}

}

经过测试我们发现，从“五棵松“到”新一区“，乘坐公交，至少换乘两次，共有两种换乘方法。





**（13）最优路线规划：输出从五棵松始发的到全图所有点的距离**

本功能的完成是求解两点之间最短路径的加强版本，我们将起点设为“五棵松“站，终点设为2-15之间任意一个站点并全部输出就完成了。

以下为本部分实现的部分代码：

//五棵松始发到全部地址的最短路径，求最短路径的变种，把起点改成五棵松就行。

void displayPath(int dist[],int path[],MGraph \*G,VertexType start,VertexType final)

{

int i,k,j=0;

int temp[VertexMax];//临时数组

VertexType target;//目标地点

int loc=0;

for(i=0;i<VertexMax;i++)

temp[i]=-1;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

//最短路径

for(i=0;i<G->vexnum;i++)

{

loc=i;//i每加一就对loc重置一遍。

j=0;

while(loc!=-1)

{//将loc计入数组以统计遍历过的地址。

temp[j]=loc;

loc=path[loc];

j++;

}

if(j-1==0&&G->Vertex[temp[j-1]]==start)//如果当前地址到起点是最短的，那么就可以输出当前点。

{

if(final==0||final==i+1)

{//合法的话就输出起点终点以便让用户看到。

printf("\t");

printf("始发：%d 终到：%d\n\t%d为起始点!",start,G->Vertex[i],G->Vertex[temp[j-1]]); printf("\n");

}

}

else if(j-1==0&&G->Vertex[temp[j-1]]!=start)

{//不合法的话就弹出警告。

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d\n\t%d不可达%d",start,G->Vertex[i],start,G->Vertex[temp[j-1]]); printf("\n");

}

}

else

{//如果是终点那么就可以结束循环了。

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d \n",start,G->Vertex[i]);

int clever=j-1;

printf("\t");

printf("最优线路：");

for(j=j-1;j>=0;j--)

{

if(j!=clever)printf("->");

printf("%d",G->Vertex[temp[j]]);

}

printf("(总路程长度:%d)",dist[i]); printf("\n");

}

}

for(k=0;k<20;k++)

temp[k]=-1;

}

}

void ShortestPath\_Dijkstra(MGraph \*G,VertexType start,VertexType final)

{

int i,j,num;

int loc;

int min;

int dist[VertexMax];//最短路径长度数组

int path[VertexMax];//最短路径数组

int s[VertexMax];//代表集合S（1代表该顶点已处理，属于集合S；0代表该顶点未处理，不属于集合S，属于集合V-S）

//1.初始化dist和path数组

loc=LocateVex(G,start);//获取源点的下标位置

for(i=0;i<G->vexnum;i++)

{

dist[i]=G->AdjMatrix[loc][i];

if(dist[i]!=MaxInt)

{

path[i]=loc;//标记存入i

}

else

{

path[i]=-1;

}

}

//2.初始化S数组（s数组：代表集合S，1代表该元素属于集合S(已处理的顶点)，0该元素不属于集合S(未处理的顶点)）

for(i=0;i<G->vexnum;i++)

{

s[i]=0;

}

s[loc]=1;//代表起始点(源点)以处理完毕

num=1;

while(num<G->vexnum)

{

min=FindMinDist(dist,s,G->vexnum);//在dist数组中查找其对应s[i]=0，即未处理的最小值元素

s[min]=1;//将找到的最短边所对应的的顶点加入集合S

for(i=0;i<G->vexnum;i++)//加入新的顶点后，更新dist和path数组

{

if((s[i]==0)&&(dist[i]>dist[min]+G->AdjMatrix[min][i]))

{

dist[i]=dist[min]+G->AdjMatrix[min][i];

path[i]=min;//min->i

}

}

num++;

}

displayPath(dist,path,G,start,final);//展示dist数组、path数组及最短路径

}

void wkssf()

{

MGraph G;

VertexType start,final;

CreateDN(&G);

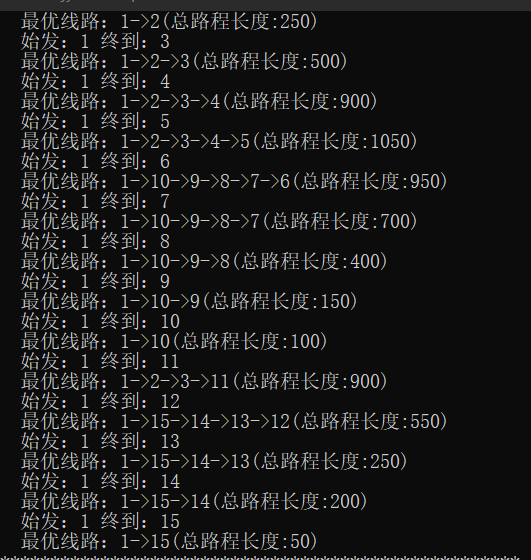
start=1;

final=0;

ShortestPath\_Dijkstra(&G,start,final);

}

本功能没有难度，再调用一遍最短路径求解就行了。



**（14）判断校内公交是否有环线**

本部分使用拓扑排序算法，实现环线判断，若图中进行拓扑排序后还剩下有环路（入度不为零的节点），那么一定就有环线了。我们不妨假设一张图上有n个点。拓扑排序的核心就是每次找入度为0的点，进入输出队列，然后将与此点相连的节点入度减1，重复做。重复做的过程中，如果存在结点不能被删除，证明存在环。

以下为本部分实现的部分代码：

//用的拓扑排序的代码，该代码可以判断环线。

void TopoSort(ALGraph \*G,int n)

{

int i,j,k,top,m=0;

EdgeNode \*p;

int \*d=(int \*)malloc(n\*sizeof(int));

for(i=0;i<n;i++) //初始化数组

{

d[i]=0;

}

for(i=0;i<n;i++) //统计各个顶点的入度情况，并把他们填入数组里面

{

p=G->adjlist[i].firstedge;

while(p!=NULL)

{

j=p->adjvex;

d[j]++;

p=p->next;

}

}

top=-1;

for(i=0;i<n;i++) //先找出里面入度是0的顶点

{

if(d[i]==0)

{

d[i]=top;

top=i;

}

}

while(top!=-1)

{

j=top;

top=d[top];

m++; //统计顶点

p=G->adjlist[j].firstedge;

while(p)

{

k=p->adjvex; //相连接的顶点

d[k]--; //相连接的顶点入度减1

if(d[k]==0) //如果发现入度为0的新顶点，从该顶点出发

{

d[k]=top;

top=k;

}

p=p->next;

}

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

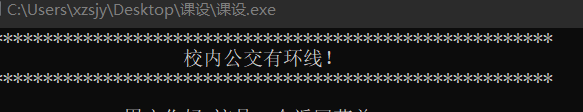
if(m<n) printf(" 校内公交有环线！\n");

else printf(" 校内公交没有环线！\n") ;

free(d);

}

以下为操作完后判断是否有环线的结果。



**（15）UI与返回界面的设计**

本系统设计了一个返回菜单用于完成某部分后返回至主菜单。这样可以提高系统使用的效率。

以下为本部分实现的部分代码：

void mainmenu()

{

start:

//UI设计

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \*这是江苏大学线路问询交互系统\* \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 请注意，本系统并未经过详细维护\n 请尽量输入合法范围内的数字避免程序崩溃。\n");

chushihua();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 1 管理员端 \* \n");

printf(" \* 2 学生端 \* \n");

printf(" \* 3 退出程序 \* \n");

printf(" \* 输入数字以进行选单 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

//选择系统代码部分

int cho1,cho2;

printf("请输入选项序号：");scanf("%d",&cho1);system("cls");

if(cho1==1)

{

administrator();

//返回菜单

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 用户你好 这是一个返回菜单 \* \n");

printf(" \* 输入0返回主菜单 \* \n");

printf(" \* 输入其他字符退出程序 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入选项序号：");scanf("%d",&cho2);system("cls");

if(cho2==0) goto start;

else exit(0);

}

else if(cho1==2)

{

student();

//返回菜单

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 用户你好 这是一个返回菜单 \* \n");

printf(" \* 输入0返回主菜单 \* \n");

printf(" \* 输入其他字符退出程序 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入选项序号：");scanf("%d",&cho2);system("cls");

if(cho2==0) goto start;

else exit(0);

}

else if(cho1==3)

exit(0);

else if(cho1==14)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* Copyright infringement \* \n");

printf(" \* 作者：孙家扬 by JSU \* \n");

goto start;

}

else

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

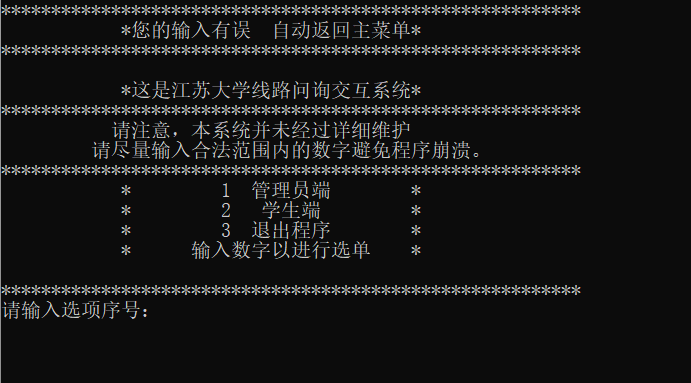
printf(" \*您的输入有误 自动返回主菜单\* \n");

goto start;

}

}

这是主菜单的设计，system("cls");可以在每次使用后清除界面达到换页效果。



**（16）一些不足与值得改进的地方**

在换乘过程中，本系统并不能给出换乘的线路以及交通方式，也不能显示换乘的站点，在输入界面，输入“\“等非合法字符会导致程序紊乱。这些是需要改进的地方。

**五．课程设计小结**

本次课程设计实践提高了我对C语言程序的应用能力，并让我将数据结构与算法课程上所学的知识得到了应用，其中遇到的一些问题，比如不能输出站点名称等，通过建立了一些中介子函数（比如输出函数，文件写入函数等）得到了比较理想的解决方案。在编码时也加深了我对学过算法的理解与认识，提高了我的逻辑认知能力，一些处理不好的地方老师也对我进行了悉心教导，问题最终得到了解决。

**六．参考文献**

1.明日科技 编著 《C语言从入门到精通》 北京 清华大学出版社 2021年9月 第五版

2.汪沁 奚李峰 编著 《数据结构与算法》 北京 清华大学出版社 2018年6月 第二版

3.谭浩强 编著 《C程序设计》 北京 清华大学出版社 2005年7月 第三版

**七．附录：完整程序代码**

#include<stdio.h>

#include<stdlib.h>

#define maxsize 16

#include<string.h>

#define MaxVertexNum 100

#define ERROR 0

#define OK 1

#define FALSE 0

#define TRUE 1

#define VertexMax 20 //最大顶点数为20

#define MaxInt 32767 //表示最大整数

#define Infinity 2000//表示无穷大

#define MAXVEX 15

#define key 15 //key为顶点个数

int shortest[15][15]; //定义全局变量存储最小路径

int pathh[15][15]; //定义存储路径

int d[30];

int qwe=0;

int visited[30];

void print2(int a)

{

if(a==1)

printf("五棵松 ");

if(a==2)

printf("三山楼 ");

if(a==3)

printf("图书馆 ");

if(a==4)

printf("京江楼 ");

if(a==5)

printf("六食堂 ");

if(a==6)

printf("学生宿舍E区 ");

if(a==7)

printf("新一区 ");

if(a==8)

printf("勤人谷 ");

if(a==9)

printf("东山运动场 ");

if(a==10)

printf("体育馆 ");

if(a==11)

printf("梅园 ");

if(a==12)

printf("学苑楼 ");

if(a==13)

printf("西山操场南 ");

if(a==14)

printf("礼堂 ");

if(a==15)

printf("三江楼 ");

}

//这是一个打印江苏大学共15个地标的子函数，该函数可以调取同文件夹的地标txt并打印。汉字以char形式打印，要读取汉字，txt必须以ANSI格式编码。

void print()

{

FILE \*fp1;

fp1=fopen("江苏大学地标.txt","r");

fp1=fopen("江苏大学地标.txt","rb");

int i;

char nam[15][100];

int num[15];

for(i=0;i<15;i++)

{

fscanf(fp1,"%d",&num[i]);

fscanf(fp1,"%s",&nam[i]);

}

for(i=0;i<15;i++)

{

printf(" ");

printf("%d ",num[i]);

printf("%s",nam[i]);

printf(" \n");

}

fclose(fp1);

}

//这是初始化子函数，读取同文件夹下的公交路线txt（该txt应以-1 开头），若文件夹为空或开头不为-1，则令文档为空并写入开头标记-1。

void chushihua()

{

FILE \*fp;

fp=fopen("公交路线.txt","r");

int sch[240];

int i;

for(i=0;i<240;i++)

{

sch[i]=0;

}

for(i=0;i<240;i++)

{

fscanf(fp,"%d",&sch[i]);

}

if(sch[0]!=-1)

{

fp=fopen("公交路线.txt","w");

fprintf(fp,"-1 ");

}

else ;

fclose(fp);

}

//这是站点查询子函数，该子函数读取公交路线文件并将其计入数组，在数组中，函数可以统计你要查询的站点，每出现一次-1公交线路序号就加一以完成线路的统计。

void site()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

int bus[240];

FILE \*fp1;

int i,t=0;

fp1=fopen("公交路线.txt","r");

for(i=0;i<240;i++)

bus[i]=0;

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);

int m=1;

int search;

printf("你要查询的站点是：");

scanf("%d",&search);system("cls");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("为您查询到该站点通过以下线路:");

for(i=0;i<240;i++)

{

if(bus[i]==-1&&bus[i+1]!=0)

m++;

if(bus[i]==search)

{

printf(" %d号线 ",m-1);

t++;

}

}

if(t==0)

printf("没有线路通过站点！");

printf("\n");

fclose(fp1);

}

//这是结构体的定义，我们接下来要进行最短路径的求取。

typedef int VertexType; //每个顶点数据类型为字符型

typedef struct

{

VertexType Vertex[VertexMax];//存放顶点元素的一维数组

int AdjMatrix[VertexMax][VertexMax];//邻接矩阵二维数组

int vexnum,arcnum;//图的顶点数和边数

}MGraph;

//这是查找子函数，该函数可以查找元素v在一维数组 Vertex[] 中的下标，并返回下标，返回下标我们就方便确定位置。

int LocateVex(MGraph \*G,VertexType v)

{

int i;

for(i=0;i<G->vexnum;i++)

{

if(v==G->Vertex[i])

{

return i;

}

}

printf("No Such Vertex!\n");

return -1;

}

//这是邻接矩阵构建子函数，该函数可以构建一个邻接矩阵，打开同文件夹的地标表文件（该文件存有路径及路径长度大小），并计入邻接矩阵。

void CreateDN(MGraph \*G)

{

int i,j;

//1.输入顶点数和边数

G->vexnum=15;

G->arcnum=36;

//2.输入顶点元素

for(i=0;i<15;i++)

G->Vertex[i]=i+1;

//3.矩阵初始化

for(i=0;i<G->vexnum;i++)

for(j=0;j<G->vexnum;j++)

{

G->AdjMatrix[i][j]=MaxInt;

}

//4.构建邻接矩阵

int n,m;

VertexType v1,v2;

int w;//v1->v2的权值

FILE \*fp1;

fp1=fopen("江苏大学地标表.txt","r");

for(i=0;i<G->arcnum;i++)

{

fscanf(fp1," %d%d%d",&v1,&v2,&w);//w是路径长度。

n=LocateVex(G,v1); //获取v1所对应的在Vertex数组中的坐标

m=LocateVex(G,v2); //获取v2所对应的在Vertex数组中的坐标

if(n==-1||m==-1)

{

printf("NO This Vertex!\n");

return;

}

G->AdjMatrix[n][m]=w;

}

fclose(fp1);

}

//这是最短路径求解子函数，依次遍历。

void displayPath(int dist[],int path[],MGraph \*G,VertexType start,VertexType final)

{

int i,k,j=0;

int temp[VertexMax];//临时数组

VertexType target;//目标地点

int loc=0;

for(i=0;i<VertexMax;i++)

temp[i]=-1;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

//最短路径

for(i=0;i<G->vexnum;i++)

{

loc=i;//i每加一就对loc重置一遍。

j=0;

while(loc!=-1)

{//将loc计入数组以统计遍历过的地址。

temp[j]=loc;

loc=path[loc];

j++;

}

if(j-1==0&&G->Vertex[temp[j-1]]==start)//如果当前地址到起点是最短的，那么就可以输出当前点。

{

if(final==0||final==i+1)

{//合法的话就输出起点终点以便让用户看到。

printf("\t");

printf("始发：%d 终到：%d\n\t%d为起始点!",start,G->Vertex[i],G->Vertex[temp[j-1]]); printf("\n");

}

}

else if(j-1==0&&G->Vertex[temp[j-1]]!=start)

{//不合法的话就弹出警告。

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d\n\t%d不可达%d",start,G->Vertex[i],start,G->Vertex[temp[j-1]]); printf("\n");

}

}

else

{//如果是终点那么就可以结束循环了。

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d \n",start,G->Vertex[i]);

int clever=j-1;

printf("\t");

printf("最优线路：");

for(j=j-1;j>=0;j--)

{

if(j!=clever)printf("->");

printf("%d",G->Vertex[temp[j]]);

}

printf("(总路程长度:%d)",dist[i]); printf("\n");

}

}

for(k=0;k<20;k++)

temp[k]=-1;

}

}

//找最近路径的子函数，不断调用此子函数以找到最短路径。

int FindMinDist(int dist[],int s[],int vexnum)

{

int i;

int loc;

int min=MaxInt+1;

for(i=0;i<vexnum;i++)

{

if(s[i]==0)//只对s[i]=0的顶点进行查找

{

if(dist[i]<min)//如果找到的路径比当前遍历过的最近路径还近，就替换它直到最短。

{

min=dist[i];

loc=i;

}

}

}

return loc;//返回dist中最小元素的下标

}

//下面是输出两点之间全部路径的子函数。我们先建立存放地址的结构体。

typedef struct

{

int vexs[MAXVEX];

int arc[MAXVEX][MAXVEX];

int numNodes;

int numEdges;

}Mgraph;

Mgraph G;

//建立图的子函数，将起点，终点都存入G.arc里，便于直观查看。

int initgraph()

{

int i,j;

for(i=0;i<=15;i++)

for(j=0;j<=15;j++)

G.arc[i][j]=Infinity;

for(i=0;i<=15;i++)

G.arc[i][i]=0;

FILE \*fp;

int m;

fp=fopen("地标缓冲表.txt","r") ;

for(m=0;m<32662;m++)

fscanf(fp,"%d%d%d",&i,&j,&G.arc[i][j]);

fclose(fp);

return ;

}

//打印每一条路径的子函数，每遍历到一个点，就接着往下找没有遍历过的点找到终点为止，然后返回最初一个拥有两个以上可以继续遍历的点，继续遍历直到完成所有遍历。

int path(int m,int n,int k);

//输出所有路径的主程序，通过这个函数，我们可以输入起止位置并调用path子函数完成所有路径的遍历。

//注意：本函数不完善没法输出相邻两点作为起止点的路径，原因我找到了是因为下一个点作为终点的时候会自动返回上一个点无法输出。

int allpath()

{ int k,i,j,m,n;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入您想要查询的两个地标的编号：");

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

if(m>=0&&m<=15&&n>=0&&n<=15)

{d[0]=m;//存储路径起点m（int d[]数组是全局变量）

for(k=0;k<15;k++)//全部顶点访问标志初值设为0

visited[k]=0;printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");printf("\n");

visited[m]=1;//第m个顶点访问标志设置为1

path(m,n,0);

}

else

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 输入有误 \n");

} //调用程序。k=1，对应起点d[1]==m。k为d[]数组下标

return 1;

}

//打印每一条路径的子函数，每遍历到一个点，就接着往下找没有遍历过的点找到终点为止，然后返回最初一个拥有两个以上可以继续遍历的点，继续遍历直到完成所有遍历。

int path(int m,int n,int k)

{ int s,t,time=0; //t用于存放路径上下一个顶点对应的d[]数组元素的下标

t=k+1;

if(d[k]==n&&k<16) //若d[k]是终点n且地点个数<=16,则输出该路径

{ for(s=0;s<k;s++) //计算路径长度

{

time=time+G.arc[d[s]][d[s+1]];

}

//打印路径长度小于定长的路径

{ for(s=0;s<k;s++) //输出该路径。s=0时为起点m

{

printf("%d->",d[s]);

}

printf("%d ",d[s]);

printf("（时间花费%d分钟）",time/50); //输出最后一个地点（即顶点n的序号，此时s==k）

printf("\n\n");

}

}

else

{ s=1;

while(s<=15)//从第m个顶点，试探所有顶点是否有路径

{ if((G.arc[d[k]][s]<Infinity)&&(visited[s]==0))//顶点m到顶点s有边(权值不为无穷大)，且未被访问

{ visited[s]=1;

d[k+1]=s;//存储顶点编号

path(m,n,t);

visited[s]=0;//将找到的路径上顶点的访问标志重新设置为0，以用于试探新的路径

}

s++;//试探从下一个顶点s开始是否有到终点的路径

}

}

}

//调取程序的子函数，用处不是很大也可以分开调取。

int allpathtest()

{

initgraph();

allpath();

return 0;

}

//这是迪杰斯特拉算法的子函数

void ShortestPath\_Dijkstra(MGraph \*G,VertexType start,VertexType final)

{

int i,j,num;

int loc;

int min;

int dist[VertexMax];//最短路径长度数组

int path[VertexMax];//最短路径数组

int s[VertexMax];//代表集合S（1代表该顶点已处理，属于集合S；0代表该顶点未处理，不属于集合S，属于集合V-S）

//1.初始化dist和path数组

loc=LocateVex(G,start);//获取源点的下标位置

for(i=0;i<G->vexnum;i++)

{

dist[i]=G->AdjMatrix[loc][i];

if(dist[i]!=MaxInt)

{

path[i]=loc;//标记存入i

}

else

{

path[i]=-1;

}

}

//2.初始化S数组（s数组：代表集合S，1代表该元素属于集合S(已处理的顶点)，0该元素不属于集合S(未处理的顶点)）

for(i=0;i<G->vexnum;i++)

{

s[i]=0;

}

s[loc]=1;//代表起始点(源点)以处理完毕

num=1;

while(num<G->vexnum)

{

min=FindMinDist(dist,s,G->vexnum);//在dist数组中查找其对应s[i]=0，即未处理的最小值元素

s[min]=1;//将找到的最短边所对应的的顶点加入集合S

for(i=0;i<G->vexnum;i++)//加入新的顶点后，更新dist和path数组

{

if((s[i]==0)&&(dist[i]>dist[min]+G->AdjMatrix[min][i]))

{

dist[i]=dist[min]+G->AdjMatrix[min][i];

path[i]=min;//min->i

}

}

num++;

}

displayPath(dist,path,G,start,final);//展示dist数组、path数组及最短路径

}

//找最短路径，下面的子函数应该都讲过了。

void zuiduanlujing()

{

MGraph G;

VertexType start,final;

CreateDN(&G);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入起始点：");

scanf(" %d",&start);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入终到点：");

scanf(" %d",&final);

system("cls");

ShortestPath\_Dijkstra(&G,start,final);

}

//五棵松始发到全部地址的最短路径，求最短路径的变种，把起点改成五棵松就行。

void wkssf()

{

MGraph G;

VertexType start,final;

CreateDN(&G);

start=1;

final=0;

ShortestPath\_Dijkstra(&G,start,final);

}

//增加公交线路子函数，因为公交可以跳站，所以默认全图联通，可任意编辑合法站点。

void addline()

{

FILE \*fp1;

fp1=fopen("公交路线.txt","a+");

int i,site;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入公交路线（以输入-1作为结束）：");

for(i=0;i<15;i++)

{

scanf("%d",&site);

if(site>15||site<-1)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

break;

}

else if(site==-1)

{

fprintf(fp1,"-1 ");//-1表示输入结束。

break;

}

else fprintf(fp1," %d ",site);//循环输入站点。

} system("cls");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 程序完成 \* \n");

fclose(fp1);

}

//辅助子函数，用于将数组写入文件，没啥用。

void help(int bus[])

{

int i;

FILE \*fp1;

fp1=fopen("公交路线.txt","w");

for(i=0;i<240;i++)

if(bus[i]!=0&&bus[i]!=-2)//换种思路，0为空，不输出0就行了 。

fprintf(fp1,"%d ",bus[i]);

fclose(fp1);

}

//修改线路子函数，原理就是直接把文件导入数组，改数组数据然后再写入文件。

void changeexist()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("您想要修改的线路是：");

int changeline;

scanf("%d",&changeline);system("cls");

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;//空数据默认为0以避免输出时输出地址影响观感。

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);//将文件里的数据导入数组。

n=0;

for(i=0;i<140;i++)

{

if(bus[i]==-1)

n++;

if(n==changeline)

{

break;

}

}

int bus2[15]={0};

int remem=i+1;

int k=0;

for(j=i+1;bus[j]!=-1;j++)

{

bus2[k]=bus[j];

k++;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("线路%d通过以下站点：",changeline);

for(i=0;bus2[i]!=0;i++)

{

print2(bus2[i]);

}

printf("\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 您想要 \n");

printf(" 1 替换站点 \n");

printf(" 2 删除站点 \n");

printf(" 3 删除线路 \n");

printf(" 4 退出程序 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入操作序号：");

int donum;

scanf("%d",&donum);system("cls");

if(donum==1)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("您想要替换掉的站点为：");

int wantchan,afterchan;

scanf("%d",&wantchan);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("替换后的站点为：");

scanf("%d",&afterchan); system("cls");

if(afterchan>15||afterchan<1||wantchan>15||wantchan<1) goto loop;

for(i=0;i<15;i++)

{

if(bus2[i]==wantchan)

bus2[i]=afterchan;

}

k=0;

for(j=remem;bus2[k]!=0;j++)

{

bus[j]=bus2[k];

k++;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 变更成功 \* \n");

help(bus);

}

else if(donum==2)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("您想要删除掉的站点为：");

int wantchan,afterchan;

scanf("%d",&wantchan); system("cls");

if(wantchan>15||wantchan<1) goto loop;

afterchan=-2;

for(i=0;i<15;i++)

{

if(bus2[i]==wantchan)

bus2[i]=afterchan;

}

k=0;

for(j=remem;bus2[k]!=0;j++)

{

bus[j]=bus2[k];

k++;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 变更成功 \* \n");

help(bus);

}

else if(donum==3)

{

for(j=remem;bus[j]!=-1;j++)

{

bus[j]=0;

}

bus[remem-1]=0;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 变更成功 \* \n");

help(bus);

}

else if(donum==4)

exit(0);

else

{

loop:

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

}

fclose(fp1);

}

//删除站点子函数，将文件里的数据导入数组并将要删除的站点变成零，在写入数组的时候不将它写入，它就被删掉了。

void deletesite()

{

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;//空数据默认为0以避免输出时输出地址影响观感。

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);//将文件里的数据导入数组。

int deletesite;

fp1=fopen("公交路线.txt","w");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n你要删除的站点为：");

scanf("%d",&deletesite);system("cls");

if(deletesite<=15&&deletesite>0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 删除成功 \* \n");

}

if(deletesite>15||deletesite<=0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

}

for(i=0;i<240;i++)

{

if(bus[i]==deletesite)

bus[i]=0;//将要删除的站点置空。

}

for(i=0;i<240;i++)

if(bus[i]!=0)

fprintf(fp1,"%d ",bus[i]);

fclose(fp1);

}

//下一站增站法。 找到要增加站点的下一站，在前面加上它。

void nextsta(int bus[],int creatsite)

{

start:

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("该站点下一站为：");

int next;

scanf("%d",&next);system("cls");//先给出增加站点的下一站（next节点）。

if(next>15||next<0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

goto start;

}

int i,j;

for(i=0;i<240;i++)

{

if(bus[i]==next)//找到我们输入的下一站。

{

for(j=239;j>=i;j--)

{

bus[j+1]=bus[j];//将下一站之后的站数据全部后移。

}

bus[i]=creatsite;//要增加的站点赋给i。

i++;

}

}

FILE \*fp1;

fp1=fopen("公交路线.txt","w");

for(i=0;i<240;i++)

if(bus[i]!=0&&bus[i]!=bus[i+1]) //同理，0为空，若原线路有该站点将不写入文件。

fprintf(fp1,"%d ",bus[i]);

fclose(fp1);

}

//上一站增站法。 原理同上。

void frontsta(int bus[],int creatsite)

{

start:

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("该站点上一站为：");

int front;

scanf("%d",&front);system("cls");

if(front>15||front<0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

goto start;

}

int i,j;

for(i=1;i<240;i++)

{

if(bus[i-1]==front)//找到我们输入的上一站。

{

for(j=239;j>=i;j--)

{

bus[j+1]=bus[j];//将上一站之后的站数据全部后移。

}

bus[i]=creatsite;//要增加的站点赋给i。

i++;

}

}

FILE \*fp1;

fp1=fopen("公交路线.txt","w");

for(i=0;i<240;i++)

if(bus[i]!=0&&bus[i]!=bus[i+1])

fprintf(fp1,"%d ",bus[i]);

fclose(fp1);

}

//两端增站法。原理同上。

void betweensta(int bus[],int creatsite)

{

start:

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

int front,next;

printf("该站点上一站为：");

scanf("%d",&front);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("该站点下一站为：");

scanf("%d",&next);system("cls");

if(front>15||front<0||next>15||next<0)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 输入有误 \* \n");

goto start;

}

int i,j;

for(i=1;i<240;i++)

{

if(bus[i-1]==front&&bus[i]==next)//找到我们输入的上一站和下一站。

{

for(j=239;j>=i;j--)

{

bus[j+1]=bus[j];//将上一站之后的站数据全部后移。

}

bus[i]=creatsite;//要增加的站点赋给i。

i++;

}

}

FILE \*fp1;

fp1=fopen("公交路线.txt","w");

for(i=0;i<240;i++)

if(bus[i]!=0&&bus[i]!=bus[i+1])

fprintf(fp1,"%d ",bus[i]);

fclose(fp1);

}

//调取增站的子函数。

void addsite()

{

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);

fclose(fp1);

int creatsite;

start:

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("你要增加的站点为：");

scanf("%d",&creatsite); system("cls");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 1 该站点有下一站 \* \n");

printf(" \* 2 该站点有上一站 \* \n");

printf(" \* 3 该站点在两站之间 \* \n");

printf(" \* 4 我不想添加站点了 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请选择站点插入方式：");

int choose1;

scanf("%d",&choose1);system("cls");

if(choose1==1)

nextsta(bus,creatsite);

else if(choose1==2)

frontsta(bus,creatsite);

else if(choose1==3)

betweensta(bus,creatsite);

else if(choose1==4)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 正试图返回主菜单 \* \n");

}

else goto start;

}

//管理员端页面。

void administrator()

{

//UI设计

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 管理员用户 你好 \* \n");

printf(" \* 1 新增公交线路 \* \n");

printf(" \* 2 修改已有线路 \* \n");

printf(" \* 3 删除已有站点 \* \n");

printf(" \* 4 增加公交站点 \* \n");

printf(" \* 5 返回上一级 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");;

int cho1;

printf("请输入选项序号：");scanf("%d",&cho1);system("cls");

if(cho1==1)

addline();

else if(cho1==2)

changeexist();

else if(cho1==3)

deletesite();

else if(cho1==4)

addsite();

else if(cho1==5)

;

else

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 您的输入不合法 \* \n");

}

printf(" \* 增加成功 \* \n");

}

//线路查询子函数，将文件数据导入数组在数组中查找相应位置线路通过的站点。

void chaxun1()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("您想要查询的线路是：");

int cxline;

scanf("%d",&cxline);system("cls");

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;//空数据默认为0以避免输出时输出地址影响观感。

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);//将文件里的数据导入数组。

n=0;

for(i=0;i<140;i++)

{

if(bus[i]==-1)

n++;

if(n==cxline)

{

break;

}

}

int bus2[15]={0};

int remem=i+1;

int k=0;

for(j=i+1;bus[j]!=-1;j++)

{

bus2[k]=bus[j];

k++;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("线路%d通过以下站点：",cxline);

for(i=0;bus2[i]!=0;i++)

{

print2(bus2[i]);

}

printf("\n");

fclose(fp1);

}

//查询所有线路的子函数，比上面的函数更简单，直接把所有线路输出一遍。

void chaxun2()

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

int bus[240];

FILE \*fp1;

int i;

fp1=fopen("公交路线.txt","r");

for(i=0;i<240;i++)

bus[i]=0;

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);

int m=1;

for(i=0;i<240;i++)

{

if(bus[i]==-1&&bus[i+1]!=0)

{

printf("\n%d号线：",m);

m++;

}

else if(bus[i]!=0&&bus[i+1]!=0)

print2(bus[i]);

else

{

printf("\n\n");

break;

}

}

fclose(fp1);

}

//将行形式的公交路线文件转化为有头节点和尾节点的表形式，行变表。

void hangbianbiao()

{

int bus[240];

FILE \*fp1;

fp1=fopen("公交路线.txt","r");

int n,i,j;

for(i=0;i<240;i++)

bus[i]=0;//空数据默认为0以避免输出时输出地址影响观感。

for(i=0;i<240;i++)

fscanf(fp1,"%d",&bus[i]);//将文件里的数据导入数组。

int busplus[240][3];

for(i=0;i<240;i++)

for(j=0;j<2;j++)

busplus[i][j]=0;

for(i=0;i<240;i++)

busplus[i][2]=5;

j=0;

for(i=0;i<240;i++)

{

if(bus[i]!=-1&&bus[i]!=0&&bus[i+1]!=-1&&bus[i+1]!=0)

{

busplus[j][0]=bus[i];

busplus[j][1]=bus[i+1];

j++;

}

}

for(i=0;i<240;i++)

for(j=0;j<2;j++)

{

if(busplus[i][j]==0)

busplus[i][2]=0;

}

fp1=fopen("公交表.txt","w");

for(i=0;i<240;i++)

{

for(j=0;j<3;j++)

{

if(busplus[i][j]!=0)

fprintf(fp1,"%d ",busplus[i][j]);

}

fprintf(fp1,"\n");

}

fclose(fp1);

}

//将变成表的公交线路与学校地图表作对比，如果两点之间有公交，则可以自动坐公交。

void duibi()

{

FILE \*fp01;

fp01=fopen("江苏大学地标表.txt","r");

FILE \*fp02;

fp02=fopen("公交表.txt","r");

int dibiao[240][3];

int gongjiao[240][3];

int i,j;

for(i=0;i<240;i++)

for(j=0;j<3;j++)

{

dibiao[i][j]=0;

gongjiao[i][j]=0;

}

for(i=0;i<240;i++)

for(j=0;j<3;j++)

{

fscanf(fp01,"%d",&dibiao[i][j]);

}

for(i=0;i<240;i++)

for(j=0;j<3;j++)

{

fscanf(fp02,"%d",&gongjiao[i][j]);

}

for(i=0;i<240;i++)

for(j=0;j<240;j++)

{

if(i!=j&&gongjiao[i][0]==gongjiao[j][0]&&gongjiao[i][1]==gongjiao[j][1])

{

gongjiao[i][0]=0;

gongjiao[i][1]=0;

gongjiao[i][2]=1;}

}

for(i=0;i<240;i++)

for(j=0;j<240;j++)

{

if((dibiao[i][0]==gongjiao[j][0]&&dibiao[i][1]==gongjiao[j][1])&&(dibiao[i][0]!=0&&gongjiao[j][0]!=0))

dibiao[i][2]=(dibiao[i][2]/gongjiao[j][2]);

else dibiao[i][2]==dibiao[i][2];

}

FILE \*fp03;

fp03=fopen("地标缓冲表.txt","w+");

for (i=0;i<240;i++)

{

for(j=0;j<3;j++)

{

if(dibiao[i][j]!=0)

fprintf(fp03,"%d ",dibiao[i][j]);

}

fprintf(fp03,"\n");

}

fclose(fp01);

fclose(fp02);

fclose(fp03);

}

//以有公交线路的地图再次寻找最短路径（两点之间有公交就相当于两点之间变短）

void CreateDN2(MGraph \*G)

{

int i,j;

//1.输入顶点数和边数

G->vexnum=15;

G->arcnum=36;

//2.输入顶点元素

for(i=0;i<15;i++)

G->Vertex[i]=i+1;

//3.矩阵初始化

for(i=0;i<G->vexnum;i++)

for(j=0;j<G->vexnum;j++)

{

G->AdjMatrix[i][j]=MaxInt;

}

//4.构建邻接矩阵

int n,m;

VertexType v1,v2;

int w;//v1->v2的权值

FILE \*fp1;

fp1=fopen("地标缓冲表.txt","r");

for(i=0;i<G->arcnum;i++)

{

fscanf(fp1," %d%d%d",&v1,&v2,&w);

n=LocateVex(G,v1); //获取v1所对应的在Vertex数组中的坐标

m=LocateVex(G,v2); //获取v2所对应的在Vertex数组中的坐标

if(n==-1||m==-1)

{

printf("NO This Vertex!\n");

return;

}

G->AdjMatrix[n][m]=w;

}

fclose(fp1);

}

void displayPath2(int dist[],int path[],MGraph \*G,VertexType start,VertexType final);

typedef int VertexType;

void ShortestPath\_Dijkstra2(MGraph \*G,VertexType start,VertexType final)

{

int i,j,num;

int loc;

int min;

int dist[VertexMax];//最短路径长度数组

int path[VertexMax];//最短路径数组

int s[VertexMax];//代表集合S（1代表该顶点已处理，属于集合S；0代表该顶点未处理，不属于集合S，属于集合V-S）

//1.初始化dist和path数组 //1079-1093行代码没啥用，我抄的以前自己写过的代码，为了避免程序不能运行就没删。

loc=LocateVex(G,start);//获取源点的下标位置

for(i=0;i<G->vexnum;i++)

{

dist[i]=G->AdjMatrix[loc][i];

if(dist[i]!=MaxInt)

{

path[i]=loc;

}

else

{

path[i]=-1;

}

}

//2.初始化S数组（s数组：代表集合S，1代表该元素属于集合S(已处理的顶点)，0该元素不属于集合S(未处理的顶点)）

for(i=0;i<G->vexnum;i++)

{

s[i]=0;

}

s[loc]=1;//代表起始点(源点)以处理完毕

num=1;

while(num<G->vexnum)

{

min=FindMinDist(dist,s,G->vexnum);//在dist数组中查找其对应s[i]=0，即未处理的最小值元素

s[min]=1;//将找到的最短边所对应的的顶点加入集合S

for(i=0;i<G->vexnum;i++)//加入新的顶点后，更新dist和path数组

{

if((s[i]==0)&&(dist[i]>dist[min]+G->AdjMatrix[min][i]))

{

dist[i]=dist[min]+G->AdjMatrix[min][i];

path[i]=min;//min->i

}

}

num++;

}

displayPath2(dist,path,G,start,final);//展示dist数组、path数组及最短路径

}

//寻找最短路径的子函数，上面已经讲过。

void displayPath2(int dist[],int path[],MGraph \*G,VertexType start,VertexType final)

{

int i,k,j=0;

int temp[VertexMax];//临时数组

VertexType target;//目标地点

int loc=0;

for(i=0;i<VertexMax;i++)

temp[i]=-1;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

//最短路径

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

for(i=0;i<G->vexnum;i++)

{

loc=i;

j=0;

while(loc!=-1)

{

temp[j]=loc;

loc=path[loc];

j++;

}

if(j-1==0&&G->Vertex[temp[j-1]]==start)

{

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d\n\t%d为起始点!",start,G->Vertex[i],G->Vertex[temp[j-1]]); printf("\n");

}

}

else if(j-1==0&&G->Vertex[temp[j-1]]!=start)

{

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d\n\t%d不可达%d",start,G->Vertex[i],start,G->Vertex[temp[j-1]]); printf("\n");

}

}

else

{

if(final==0||final==i+1)

{

printf("\t");

printf("始发：%d 终到：%d \n",start,G->Vertex[i]);

int clever=j-1;

printf("\t");

printf("最优线路：");

for(j=j-1;j>=0;j--)

{

if(j!=clever)printf("->");

printf("%d",G->Vertex[temp[j]]);

}

printf("(总路程时间:%.2f分钟)",((float)dist[i])/50); printf("\n");

}

}

for(k=0;k<20;k++)

temp[k]=-1;

}

}

//最短路径。

void zuiduanlujing2()

{

MGraph G;//写的最短路径其实是求最小时间，为减轻工作量用的最短路径的源码。

VertexType start,final;

CreateDN2(&G);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入起始点：");

scanf(" %d",&start);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入终到点：");

scanf(" %d",&final);

system("cls");

ShortestPath\_Dijkstra2(&G,start,final);

}

//最短路径。

void zuiduan2()

{

hangbianbiao();

duibi();

zuiduanlujing2();

}

//查询公交表有几行的子函数，便于确定有几条弧，以便确定有无环线。

int hangshu()

{

FILE \*fp;

fp=fopen("公交表.txt","r");

int a[240];int i;

for(i=0;i<240;i++)

a[i]=0 ;

for(i=0;i<240;i++)

fscanf(fp,"%d",&a[i]);

for(i=0;i<240;i++)

{

if(a[i]==0)

break;

}

int m;m=i/3;

return m;

fclose(fp);

}

//创建结构体以存储公交路径。

typedef int Boolean;

typedef int VertexType1;

Boolean visit[MaxVertexNum];

typedef struct node

{

int adjvex;//弧

struct node \*next;//下一个节点

}EdgeNode;

typedef struct

{

VertexType1 vertex;

EdgeNode \*firstedge;//头节点

}VertexNode;

typedef VertexNode AdjList[MaxVertexNum];

typedef struct

{

AdjList adjlist;

int n,e;//弧和节点个数

}ALGraph;

int FindVertex(ALGraph \*G ,int e,int n)

{

int i;

for(i=0;i<n;i++)

{

if(G->adjlist[i].vertex==e)

{

return i;

}

}

return -1;

}

void create(ALGraph \*G) //创建邻接表

{

int i,j,k,w,v;

EdgeNode \*s;

G->n=15;//共有15个节点。

G->e=hangshu();//共有【读取行数】个弧 。

for(i=0;i<G->n;i++)

{

fflush(stdin);

G->adjlist[i].vertex=i+1;

G->adjlist[i].firstedge=NULL;

}

FILE \*fp2;

fp2=fopen("公交表.txt","r");

int num;

for(k=0;k<G->e;k++)

{

fscanf(fp2,"%d%d%d",&i,&j,&num);

i=FindVertex(G,i,G->n);

j=FindVertex(G,j,G->n);//读入表中数据。

if(i==-1||j==-1)

{

fscanf(fp2,"%d%d%d",&i,&j,&num);

i=FindVertex(G,i,G->n);

j=FindVertex(G,j,G->n);//读入表中数据。

}

s=(EdgeNode\*)malloc(sizeof(EdgeNode));

s->adjvex=(j);

s->next=G->adjlist[i].firstedge;

G->adjlist[i].firstedge=s;//找到下一个节点。

}

fclose(fp2);

}

//用的拓扑排序的代码，该代码可以判断环线。

void TopoSort(ALGraph \*G,int n)

{

int i,j,k,top,m=0;

EdgeNode \*p;

int \*d=(int \*)malloc(n\*sizeof(int));

for(i=0;i<n;i++) //初始化数组

{

d[i]=0;

}

for(i=0;i<n;i++) //统计各个顶点的入度情况，并把他们填入数组里面

{

p=G->adjlist[i].firstedge;

while(p!=NULL)

{

j=p->adjvex;

d[j]++;

p=p->next;

}

}

top=-1;

for(i=0;i<n;i++) //先找出里面入度是0的顶点

{

if(d[i]==0)

{

d[i]=top;

top=i;

}

}

while(top!=-1)

{

j=top;

top=d[top];

m++; //统计顶点

p=G->adjlist[j].firstedge;

while(p)

{

k=p->adjvex; //相连接的顶点

d[k]--; //相连接的顶点入度减1

if(d[k]==0) //如果发现入度为0的新顶点，从该顶点出发

{

d[k]=top;

top=k;

}

p=p->next;

}

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

if(m<n) printf(" 校内公交有环线！\n");

else printf(" 校内公交没有环线！\n") ;

free(d);

}

//判断环线的子函数。

void panduanhuanxian()

{

hangbianbiao();

ALGraph \*G=(ALGraph \*)malloc(sizeof(ALGraph));

EdgeNode \*p;

create(G);

int i;

for(i=0;i<G->n;i++)

{

p=G->adjlist[i].firstedge;

while(p!=NULL)

{

p=p->next;

}

}

TopoSort(G,G->n);

}

int initgraph22()

{

int i,j;

for(i=0;i<=15;i++)

for(j=0;j<=15;j++)

G.arc[i][j]=Infinity;

for(i=0;i<=15;i++)

G.arc[i][i]=0;

FILE \*fp;

int m;

fp=fopen("地标缓冲表.txt","r") ;

for(m=0;m<32662;m++)

fscanf(fp,"%d%d%d",&i,&j,&G.arc[i][j]);

fclose(fp);

return ;

}

void huanchengzhan()

{

FILE \*fp1;

FILE \*fp2;

fp1=fopen("公交路线.txt","r");

fp2=fopen("换乘表.txt","w");

fprintf(fp2," ");

fp2=fopen("换乘表.txt","w");

int hcb[240]={0};

int i;

for(i=0;i<240;i++)

{

fscanf(fp1,"%d",&hcb[i]);

}

int hcb2[240]={0};

for(i=0;i<240;i++)

{

hcb2[i]=hcb[i];

}

int j;

int hcb3[240]={0};

int k=0;

for(i=0;i<240;i++)

for(j=0;j<240;j++)

{

if(i==j) j++;

if(hcb[i]==hcb2[j]&&hcb[i]!=0&&hcb[i]!=-1)

{

hcb3[k]=hcb[i];

k++;

}

}

for(i=0;i<240;i++)

for(j=0;j<240;j++)

{

if(hcb3[i]==hcb3[j]&&i!=j)

hcb3[i]=0;

}

for(i=0;i<240;i++)

{

if(hcb3[i]!=0)

fprintf(fp2,"%d ",hcb3[i]);

}

fclose(fp1);

fclose(fp2);

}

//打印每一条路径的子函数，每遍历到一个点，就接着往下找没有遍历过的点找到终点为止，然后返回最初一个拥有两个以上可以继续遍历的点，继续遍历直到完成所有遍历。

int path22(int m,int n,int k);

//输出所有路径的主程序，通过这个函数，我们可以输入起止位置并调用path子函数完成所有路径的遍历。

//注意：本函数不完善没法输出相邻两点作为起止点的路径，原因我找到了是因为下一个点作为终点的时候会自动返回上一个点无法输出。

int allpath22()

{ int k,i,j,m,n;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

print();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入您想要查询的两个地标的编号：");

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

if(m>=0&&m<=15&&n>=0&&n<=15)

{d[0]=m;//存储路径起点m（int d[]数组是全局变量）

for(k=0;k<15;k++)//全部顶点访问标志初值设为0

visited[k]=0;printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");printf("\n");

visited[m]=1;//第m个顶点访问标志设置为1

path22(m,n,0);

}

else

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 输入有误 \n");

} //调用程序。k=1，对应起点d[1]==m。k为d[]数组下标

return 1;

}

//打印每一条路径的子函数，每遍历到一个点，就接着往下找没有遍历过的点找到终点为止，然后返回最初一个拥有两个以上可以继续遍历的点，继续遍历直到完成所有遍历。

int path22(int m,int n,int k)

{ int s,t,time=0; //t用于存放路径上下一个顶点对应的d[]数组元素的下标

t=k+1;

FILE \*fp;int i;

fp=fopen("换乘表.txt","r");

int hcb[240]={0};

for(i=0;i<240;i++)

{

fscanf(fp,"%d",&hcb[i]);

}

int temp=0;

if(d[k]==n&&k<16) //若d[k]是终点n且地点个数<=16,则输出该路径

{ for(s=0;s<k;s++) //计算路径长度

{

time=time+G.arc[d[s]][d[s+1]];

}

//打印路径长度小于定长的路径

{ for(s=0;s<k;s++) //输出该路径。s=0时为起点m

{

printf("%d->",d[s]);

for(i=0;i<240;i++)

{

if(d[s]==hcb[i])

{

temp++;break;}

}

}

printf("%d ",d[s]);

printf("（时间花费%d分钟,共换乘%d次）",time/50,temp); //输出最后一个地点（即顶点n的序号，此时s==k）

printf("\n\n");

}

}

else

{ s=1;

while(s<=15)//从第m个顶点，试探所有顶点是否有路径

{ if((G.arc[d[k]][s]<Infinity)&&(visited[s]==0))//顶点m到顶点s有边(权值不为无穷大)，且未被访问

{ visited[s]=1;

d[k+1]=s;//存储顶点编号

path22(m,n,t);

visited[s]=0;//将找到的路径上顶点的访问标志重新设置为0，以用于试探新的路径

}

s++;//试探从下一个顶点s开始是否有到终点的路径

}

}

}

//调取程序的子函数，用处不是很大也可以分开调取。

int allpathtest22()

{

huanchengzhan();

initgraph22();

allpath22();

return 0;

}

void student()

{

start:

//UI设计

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 学生用户 你好 \* \n");

printf(" \* 1 查询公交线路 \* \n");

printf(" \* 2 规划公交线路 \* \n");

printf(" \* 3 规划最优线路 \* \n");

printf(" \* 4 判断校内环线 \* \n");

printf(" \* 5 返回上一级 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

int cho1,cho2;

printf("请输入选项序号：");scanf("%d",&cho1);system("cls");

if(cho1==1)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 1 查询指定线路 \* \n");

printf(" \* 2 查询线路列表 \* \n");

printf(" \* 3 查询过站线路 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入选项序号：");scanf("%d",&cho2);system("cls");

if(cho2==1)

chaxun1();

else if(cho2==2)

chaxun2();

else if(cho2==3)

site();

else goto start;

}

else if(cho1==2)

allpathtest();

else if(cho1==3)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 1 步行最短距离 \* \n");

printf(" \* 2 公交最短距离 \* \n");

printf(" \* 3 换乘次数最少 \* \n");

printf(" \* 4 从五棵松始发 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入选项序号：");scanf("%d",&cho2);system("cls");

if(cho2==1)

zuiduanlujing();

else if(cho2==2)

zuiduanlujing2();

else if(cho2==3)

allpathtest22();

else if(cho2==4)

wkssf();

}

if(cho1==4)

panduanhuanxian();

else;

}

void mainmenu()

{

start:

//UI设计

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \*这是江苏大学线路问询交互系统\* \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 请注意，本系统并未经过详细维护\n 请尽量输入合法范围内的数字避免程序崩溃。\n");

chushihua();

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* 1 管理员端 \* \n");

printf(" \* 2 学生端 \* \n");

printf(" \* 3 退出程序 \* \n");

printf(" \* 输入数字以进行选单 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

//选择系统代码部分

int cho1,cho2;

printf("请输入选项序号：");scanf("%d",&cho1);system("cls");

if(cho1==1)

{

administrator();

//返回菜单

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 用户你好 这是一个返回菜单 \* \n");

printf(" \* 输入0返回主菜单 \* \n");

printf(" \* 输入其他字符退出程序 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入选项序号：");scanf("%d",&cho2);system("cls");

if(cho2==0) goto start;

else exit(0);

}

else if(cho1==2)

{

student();

//返回菜单

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" \* 用户你好 这是一个返回菜单 \* \n");

printf(" \* 输入0返回主菜单 \* \n");

printf(" \* 输入其他字符退出程序 \* \n\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入选项序号：");scanf("%d",&cho2);system("cls");

if(cho2==0) goto start;

else exit(0);

}

else if(cho1==3)

exit(0);

else if(cho1==14)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \* Copyright infringement \* \n");

printf(" \* 作者：孙家扬 by JSU \* \n");

goto start;

}

else

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" \*您的输入有误 自动返回主菜单\* \n");

goto start;

}

}

int main()

{

mainmenu();

}