

**厦门大学数据结构实验报告**

**2017-2018学年第一学期**

**实验名称：实验五 全国交通咨询模拟**

**系 别： 智能系**

**学生学号： 22920162204046**

**学生姓名： 王凯**

题目： 全国交通咨询模拟

1. **需求分析**

【基本要求】

(1)提供对城市信息进行编辑(如:添加或删除)的功能

(2)城市之间有两种交通工具:火车和飞机。提供对列车时刻表和飞机航班进行编辑(增设或删除)的

(3)提供两种最优决第最快到达或最省钱到达。全程只考虑一种交通工具

(4)旅途中耗费的总时间应该包括中转站的等候时间。

(5)咨询以用户和计算机的对话方式进行。由用户输入起始站、终点站、最优决策原则和交通工具,输出信息:最快需要多长时间才能到达或者最少需要多少旅费才能到达,并详细说明依次于何时乘坐哪一趟列车或哪一次班机到何地。

**二、概要设计**

**1：抽象数据类型**

int Start();

//界面

int InitSysData();

//初始化数据

int AddCity (char \*Name);

//添加城市

int FindNumofCity (char \*name);

//查找城市，并返回城市序号

int DelCity (char \*Name);

//删除城市

int AddTrain(char \*train,char \*DepCity,char \*DesCity,int DepTime,int EndTime,int cost);

//添加火车

int AddFlight(char \*flight,char \*DepCity,char \*DesCity,int DepTime,int EndTime,int cost);

//添加飞机航班

int DelPath(char \*name);

//删除路径

void FindMinPrice\_Print(int matx[34][34],int PreCity[34],int p\_end,int ByTorByP);

//打印最低价格路线

int CalcMinPrice(int DepCity,int DesCity,int ByTorByP);

//计算费用

void FindMinPrice(int matx[34][34],int p\_start,int p\_end,int ByTorByP);

//寻找最低价格路线

int SearchMinTime(NumType City,NumType DesCity,int CurTime,int curPathNo,int ByTorByP);

//寻找最短时间路线

int CalcMinTime(int DepCity,int DesCity,int ByTorByP);

//计算时间

int SaveSysInfo();

//存储数据

**2：主程序**

int main(){

while(1){

接受命令；

处理命令；

**}**

**}**

1. **详细设计**

**1、存储结构**

typedef short int NumType;

typedef struct TrafficWay{

char name[15]; //列次/班次

int DepTime; //出发时间

int ArriveTime; //到达时间

int DesCity; //到达城市编号

int Price; //票价

}TrafficWayDat; //交通工具信息

typedef struct CityNode{

NumType city; //城市的个数

int TrainNum,FlightNum; //路过某个城市的火车/飞机的编号

TrafficWayDat Train[10];

TrafficWayDat Flight[10];

}CityNodeDat;//

typedef struct PathNode {

int City; //城市

int FlainNo; //火车或者航班的此号

}PathNodeDat; //存储路径

**2、函数/算法实现**

int Start()

//界面

{

printf("\*-----------此系统提示说明------------\*\n");

printf("\*选择你要进行的步骤: \*\n");

printf("\* 1:添加城市： \*\n");

printf("\* 2:删除城市： \*\n");

printf("\* 3:添加交通路线： \*\n");

printf("\* 4:删除交通路线： \*\n");

printf("\* 5:查找价格最低路线： \*\n");

printf("\* 6:查找时间最短路线： \*\n");

printf("\* 0:退出系统 \*\n");

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

return 1;

}

int InitSysData()

//初始化数据

{

FILE \*fp;

int i,j,hour,minute,num,cost;

char String1[15];

char String2[15];

char String3[15];

//初始化城市信息：

fp=fopen(CityFile,"r");

if(!fp){

printf("文件打开出错!\n");

return -1;

}

fscanf(fp,"%d",&NumofCity); //读入城市数目

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

{

fscanf(fp,"%s",&CityName[i]); //读入城市名字

CityInfo[i].city=i; //给城市编号

CityInfo[i].TrainNum=0; //火车数目暂定为0

CityInfo[i].FlightNum=0; //飞机数目暂定为0

}

fclose(fp);

//初始化火车路线信息：

fp=fopen(TrainFile,"r");

if(!fp){

printf("\n文件打开错误\n请先初始化系统数据!");

return -1;

}

fscanf(fp,"%d",&num); //交通干线数目

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

fscanf(fp,"%s",&String1); //读入列车号

fscanf(fp,"%s",&String2); //读入起始地点

fscanf(fp,"%s",&String3); //读入目的地

j=FindNumofCity(String2); //读取城市编号

//路过此城市的火车的目的地

CityInfo[j].Train[CityInfo[j].TrainNum].DesCity=FindNumofCity(String3);

strcpy(CityInfo[j].Train[CityInfo[j].TrainNum].name,String1);//车次赋给次列车

fscanf(fp,"%d:%d",&hour,&minute); //读入出发时间

CityInfo[j].Train[CityInfo[j].TrainNum].DepTime=hour\*60+minute;

fscanf(fp,"%d:%d",&hour,&minute); //读入到达时间

CityInfo[j].Train[CityInfo[j].TrainNum].ArriveTime=hour\*60+minute;

fscanf(fp,"%d",&cost); //读入费用

CityInfo[j].Train[CityInfo[j].TrainNum].Price=cost;

CityInfo[j].TrainNum++;

}

fclose(fp);

//初始化飞机路线信息；一切同上。

fp=fopen(FlightFile,"r");

if(!fp){

printf("\n文件打开错误!\n请先初始化系统数据!");

return -1;

}

fscanf(fp,"%d",&num);

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

fscanf(fp,"%s",&String1);

fscanf(fp,"%s",&String2);

fscanf(fp,"%s",&String3);

j=FindNumofCity(String2);

CityInfo[j].Flight[CityInfo[j].FlightNum].DesCity=FindNumofCity(String3);

strcpy(CityInfo[j].Flight[CityInfo[j].FlightNum].name,String1);

fscanf(fp,"%d:%d",&hour,&minute);

CityInfo[j].Flight[CityInfo[j].FlightNum].DepTime=hour\*60+minute;

fscanf(fp,"%d:%d",&hour,&minute);

CityInfo[j].Flight[CityInfo[j].FlightNum].ArriveTime=hour\*60+minute;

fscanf(fp,"%d",&cost);

CityInfo[j].Flight[CityInfo[j].FlightNum].Price=cost;

CityInfo[j].FlightNum++;

}

fclose(fp);

return 1;

}

int AddCity(char \*Name)

//添加城市

{

strcpy(CityName[NumofCity],Name); //第i城市名

CityInfo[NumofCity].city=NumofCity;

CityInfo[NumofCity].TrainNum=0;

CityInfo[NumofCity].FlightNum=0;

NumofCity++;

return 1;

}

int FindNumofCity (char \*name)

//查找城市的序号

{

int i;

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

if(strcmp(name,CityName[i])==0)

return i;

return -1;

}

int DelCity(char \*Name)

//删除城市

{

int city,i,j;

city=FindNumofCity(Name); //找到城市的编号

for(i=city;i<NumofCity-1;i++){ //前移

strcpy(CityName[i],CityName[i+1]); //城市名

CityInfo[i].TrainNum=CityInfo[i+1].TrainNum; //火车

CityInfo[i].FlightNum=CityInfo[i+1].FlightNum; //航班

for(j=0;j<CityInfo[i].TrainNum;j++){ //删除此地的火车信息

CityInfo[i].Train[j].Price=CityInfo[i+1].Train[j].Price;

CityInfo[i].Train[j].DesCity=CityInfo[i+1].Train[j].DesCity;

strcpy(CityInfo[i].Train[j].name,CityInfo[i+1].Train[j].name);

CityInfo[i].Train[j].DepTime=CityInfo[i+1].Train[j].DepTime;

CityInfo[i].Train[j].ArriveTime=CityInfo[i+1].Train[j].ArriveTime;

}

for(j=0;j<CityInfo[i].FlightNum;j++){ //删除此地的航班信息

CityInfo[i].Flight[j].Price=CityInfo[i+1].Flight[j].Price;

CityInfo[i].Flight[j].DesCity=CityInfo[i+1].Flight[j].DesCity;

strcpy(CityInfo[i].Flight[j].name,CityInfo[i+1].Flight[j].name);

CityInfo[i].Flight[j].DepTime=CityInfo[i+1].Flight[j].DepTime;

CityInfo[i].Flight[j].ArriveTime=CityInfo[i+1].Flight[j].ArriveTime;

}

}

NumofCity--;

return 1;

}

int AddTrain(char \*train,char \*DepCity,char \*DesCity,int DepTime,int EndTime,int cost)

//给两地添加火车

{

int i,j;

i=FindNumofCity(DepCity); //起始城市编号

j=FindNumofCity(DesCity); //目的地城市编号

CityInfo[i].Train[CityInfo[i].TrainNum].Price=cost; //费用

CityInfo[i].Train[CityInfo[i].TrainNum].DesCity=j; //从i->j

CityInfo[i].Train[CityInfo[i].TrainNum].DepTime=DepTime; //起始时间

CityInfo[i].Train[CityInfo[i].TrainNum].ArriveTime=EndTime;//到达时间

strcpy(CityInfo[i].Train[CityInfo[i].TrainNum].name,train);//列车编号

CityInfo[i].TrainNum++; //从此城市的出发的车辆+1

return 1;

}

int AddFlight(char \*flight,char \*DepCity,char \*DesCity,int DepTime,int EndTime,int cost)

//给两地添加航班

{

int i,j;

i=FindNumofCity(DepCity);

j=FindNumofCity(DesCity);

CityInfo[i].Flight[CityInfo[i].FlightNum].Price=cost;

CityInfo[i].Flight[CityInfo[i].FlightNum].DesCity=j;

CityInfo[i].Flight[CityInfo[i].FlightNum].DepTime=DepTime;

CityInfo[i].Flight[CityInfo[i].FlightNum].ArriveTime=EndTime;

strcpy(CityInfo[i].Flight[CityInfo[i].FlightNum].name,flight);

CityInfo[i].FlightNum++;

return 1;

}

int DelPath(char \*name)

//删除火车//航班

{

int i;

int j;

int flag=0;

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

//删除火车

for(j=0;j<CityInfo[i].TrainNum;j++)

if (strcmp(CityInfo[i].Train[j].name,name)==0){

flag=1;

break;

}

if(flag){

for (;j<CityInfo[i].TrainNum-1;j++){

CityInfo[i].Train[j].Price=CityInfo[i].Train[j+1].Price;

CityInfo[i].Train[j].DesCity=CityInfo[i].Train[j+1].DesCity;

strcpy(CityInfo[i].Train[j].name,CityInfo[i].Train[j+1].name);

CityInfo[i].Train[j].DepTime=CityInfo[i].Train[j+1].DepTime;

CityInfo[i].Train[j].ArriveTime=CityInfo[i].Train[j+1].ArriveTime;

}

CityInfo[i].TrainNum--;

break;

}

//删除航班

for(j=0;j<CityInfo[i].FlightNum;j++)

if(strcmp(CityInfo[i].Flight[j].name,name)==0){

flag=1;

break;

}

if(flag){

for(;j<CityInfo[i].FlightNum-1;j++){

CityInfo[i].Flight[j].Price=CityInfo[i].Flight[j+1].Price;

CityInfo[i].Flight[j].DesCity=CityInfo[i].Flight[j+1].DesCity;

strcpy(CityInfo[i].Flight[j].name,CityInfo[i].Flight[j+1].name);

CityInfo[i].Flight[j].DepTime=CityInfo[i].Flight[j+1].DepTime;

CityInfo[i].Flight[j].ArriveTime=CityInfo[i].Flight[j+1].ArriveTime;

}

CityInfo[i].FlightNum--;

break;

}

}

return 1;

}

void FindMinPrice\_Print(int matx[34][34],int PreCity[34],int p\_end,int Vehicle)

//打印最低费用费用路径

{

int track[34];

int i=0,j,k,min,tmp,end,cost=0;

j=p\_end;

track[i++]=j;

while(PreCity[j]>=0){ //求出最低费用

cost+=matx[PreCity[j]][j];

track[i++]=j=PreCity[j];

}

printf("\n旅行路线:");

if(!Vehicle){

for(i--;i>0;i--){

printf("\n%s:",CityName[track[i]]);

end=track[i-1];

min=32767;

for(k=0;k<CityInfo[track[i]].TrainNum;k++)

if(CityInfo[track[i]].Train[k].DesCity==end&&min>CityInfo[track[i]].Train[k].Price){

min=CityInfo[track[i]].Train[k].Price;

tmp=k;

}

printf("请乘坐%s次列车，起止时间:",CityInfo[track[i]].Train[tmp].name);

printf("%02d:%02d-%02d:%02d",CityInfo[track[i]].Train[tmp].DepTime/60,CityInfo[track[i]].Train[tmp].DepTime%60,CityInfo[track[i]].Train[tmp].ArriveTime/60,CityInfo[track[i]].Train[tmp].ArriveTime%60);

}

}else{

for(i--;i>0;i--){

printf("\n%s:",CityName[track[i]]);

end=track[i-1];

min=32767;

for(k=0;k<CityInfo[track[i]].FlightNum;k++)

if(CityInfo[track[i]].Train[k].DesCity==end&&min>CityInfo[track[i]].Flight[k].Price){

min=CityInfo[track[i]].Flight[k].Price;

tmp=k;

}

printf("请乘坐%s次航班，起止时间:",CityInfo[track[i]].Flight[tmp].name);

printf("%02d:%02d-%02d:%02d",CityInfo[track[i]].Flight[tmp].DepTime/60,CityInfo[track[i]].Flight[tmp].DepTime%60,CityInfo[track[i]].Flight[tmp].ArriveTime/60,CityInfo[track[i]].Flight[tmp].ArriveTime%60);

}

}

printf("\n%s:已到达目的地",CityName[track[0]]);

printf("\n最低价格:%d\n",cost);

}

void FindMinPrice(int matx[34][34],int p\_start,int p\_end,int Vehicle)

//寻找最低费用路径,(迪杰斯特拉算法)

{

int PreCity[34];

int i,j,min,pre,pos;

for(i=0;i<NumofCity;i++) //初始化将所有城市标记为-1

PreCity[i]=-1;

PreCity[p\_start]=-2; //将起始城市标记为-2

while(PreCity[p\_end]==-1){

min=-1;

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

if(PreCity[i]!=-1){ //起始城市

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

if(PreCity[j]==-1&&matx[i][j]>0&&(min<0||matx[i][j]<min)){

pre=i;

pos=j;

min=matx[i][j];

}

}

PreCity[pos]=pre;

}

FindMinPrice\_Print(matx,PreCity,p\_end,Vehicle);

}

int CalcMinPrice(int DepCity,int DesCity,int Vehicle)

//计算最低费用,用到迪杰斯特拉算法

{

int ma[34][34]; //存储两地之间的价格

int i,j,min,end;

for(i=0;i<NumofCity;i++) //初始化,两地之间的价格赋值-1；

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

ma[i][j]=-1;

//相当于赋权值

if(Vehicle==0){ //乘坐火车

//计算存在信息的任意两个城市之间的最低价格（直达）

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

min=32767;

j=0;

while(j<CityInfo[i].TrainNum){

min=32767;

end=CityInfo[i].Train[j].DesCity;

//找i城市->j城市之间的最低价格(可以存在多趟列车，价格不一)

while(end==CityInfo[i].Train[j].DesCity&&j<CityInfo[i].TrainNum){

if(CityInfo[i].Train[j].Price<min)

min=CityInfo[i].Train[j].Price;

j++;

}

ma[i][end]=min;

}

}

}else{ //乘坐飞机

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

min=32767;

j=0;

while(j<CityInfo[i].FlightNum){

min=32767;

end=CityInfo[i].Flight[j].DesCity;

while(end==CityInfo[i].Flight[j].DesCity&&j<CityInfo[i].FlightNum){

if(CityInfo[i].Flight[j].Price<min)

min=CityInfo[i].Flight[j].Price;

j++;

}

ma[i][end]=min;

}

}

}

FindMinPrice(ma,DepCity,DesCity,Vehicle);

return 1;

}

int SearchMinTime(NumType City,NumType DesCity,int CurTime,int curPathNo,int Vehicle)

//寻找最小时间,(迪杰斯特拉算法)

{

int i;

if(City==DesCity){

if(MinTime>CurTime-DepTime){

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

MinPath[i].City=TemPath[i].City;

MinPath[i].FlainNo=TemPath[i].FlainNo;

curPath=curPathNo;

}

if(CurTime<DepTime)

MinTime=CurTime+1440-DepTime;

else

MinTime=CurTime-DepTime;

}

}else{

curPathNo++;

TemPath[curPathNo].City=City;

if(!Vehicle){//火车

for(i=0;i<CityInfo[City].TrainNum;i++){

if((CityInfo[City].Train[i].DepTime>=(CurTime%1440))&&(CityInfo[City].

Train[i].ArriveTime+(CurTime/1440)\*1440-DepTime<MinTime)){

TemPath[curPathNo].FlainNo=i; SearchMinTime(CityInfo[City].Train[i].DesCity,DesCity,CityInfo[City].

Train[i].ArriveTime+(CurTime/1440)\*1440,curPathNo,Vehicle);

} if((CityInfo[City].Train[i].DepTime<(CurTime%1440))&&(CityInfo[City].Train[i].ArriveTime+(CurTime/1440)\*1440-DepTime<MinTime)){

TemPath[curPathNo].FlainNo=i; SearchMinTime(CityInfo[City].Train[i].DesCity,DesCity,CityInfo[City].Train[i].ArriveTime+(CurTime/1440+1)\*1440,curPathNo,Vehicle);

}

}

}else{//飞机

for(i=0;i<CityInfo[City].FlightNum;i++){

if((CityInfo[City].Flight[i].DepTime>=CurTime)&&(CityInfo[City].

Flight[i].ArriveTime+(CurTime/1440)\*1440-DepTime<MinTime)){

TemPath[curPathNo].FlainNo=i; SearchMinTime(CityInfo[City].Flight[i].DesCity,DesCity,CityInfo[City].Flight[i].ArriveTime+(CurTime/1440)\*1440,curPathNo,Vehicle);

} if((CityInfo[City].Flight[i].DepTime<CurTime)&&(CityInfo[City].Flight[i].ArriveTime+(CurTime/1440)\*1440-DepTime<MinTime)){

TemPath[curPathNo].FlainNo=i;

SearchMinTime(CityInfo[City].Flight[i].DesCity,DesCity,CityInfo[City].Flight[i].ArriveTime+(CurTime/1440+1)\*1440,curPathNo,Vehicle);

}

}

}

}

return 1;

}

int CalcMinTime(int DepCity,int DesCity,int Vehicle)

//计算最小时间

{

int i;

MinTime=32767; //最大值

curPath=0;

TemPath[0].City=DepCity;

if(!Vehicle){ //火车

for(i=0;i<CityInfo[DepCity].TrainNum;i++){

TemPath[0].FlainNo=i;

DepTime=CityInfo[DepCity].Train[i].DepTime; SearchMinTime(CityInfo[DepCity].Train[i].DesCity,DesCity,CityInfo[DepCity].Train[i].ArriveTime,0,Vehicle);

}

}else{ //飞机

for(i=0;i<CityInfo[DepCity].FlightNum;i++){

TemPath[0].FlainNo=i;

DepTime=CityInfo[DepCity].Flight[i].DepTime; SearchMinTime(CityInfo[DepCity].Flight[i].DesCity,DesCity,CityInfo[DepCity].Flight[i].ArriveTime,0,Vehicle);

}

}

if(MinTime==32767){

printf("\n系统中无该路线！");

return 0;

}

printf("\n旅行路线:\n");

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

if(!Vehicle)

printf("%s:请乘坐%s次列车起止时间:",CityName[MinPath[i].City],

CityInfo[MinPath[i].City].Train[MinPath[i].FlainNo].name);

//打印城市名和车次

else

printf("%s:请乘坐%s次航班",CityName[MinPath[i].City],

CityInfo[MinPath[i].City].Flight[MinPath[i].FlainNo].name);

//打印城市名和飞机航班

printf("%02d:%02d---%02d:%02d\n",CityInfo[MinPath[i].City].Train[MinPath[i].FlainNo].DepTime/60,CityInfo[MinPath[i].City].Train[MinPath[i].FlainNo].DepTime%60,CityInfo[MinPath[i].City].Train[MinPath[i].FlainNo].ArriveTime/60,CityInfo[MinPath[i].City].Train[MinPath[i].FlainNo].ArriveTime%60);

}

printf("%s:已到达目的地",CityName[DesCity]);

printf("\n旅途耗费时间:%02d:%02d",MinTime/60,MinTime%60);

return 1;

}

int SaveSysInfo()

//保存信息

{

FILE \*fp;

int i,j,total;

//写入城市信息

fp=fopen(CityFile,"w");

fprintf(fp,"%d\n",NumofCity);

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

fprintf(fp,"%s\n",CityName[i]);

fclose(fp);

//写入火车的信息

total=0;

fp=fopen(TrainFile,"w");

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

total+=CityInfo[i].TrainNum;

fprintf(fp,"%d\n",total);

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

for(j=0;j<CityInfo[i].TrainNum;j++){

//写入列车名、起始城市、终点城市

fprintf(fp,"%s %s %s ",CityInfo[i].Train[j].name,

CityName[i],CityName[CityInfo[i].Train[j].DesCity]);

//写入起始时间

fprintf(fp,"%2d:%2d ",CityInfo[i].Train[j].DepTime/60,

CityInfo[i].Train[j].DepTime%60);

//写入到达时间、消费

fprintf(fp,"%2d:%2d ",CityInfo[i].Train[j].ArriveTime/60,

CityInfo[i].Train[j].ArriveTime%60);

//写入消费

fprintf(fp,"%d\n",CityInfo[i].Train[j].Price);

}

}

fclose(fp);

//写入飞机的信息

total=0;

fp=fopen(FlightFile,"w");

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

total+=CityInfo[i].FlightNum;

fprintf(fp,"%d\n",total);

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

for(j=0;j<CityInfo[i].FlightNum;j++){

fprintf(fp,"%s %s %s ",CityInfo[i].Flight[j].name,

CityName[i],CityName[CityInfo[i].Flight[j].DesCity]);

fprintf(fp,"%2d:%2d ",CityInfo[i].Flight[j].DepTime/60,

CityInfo[i].Flight[j].DepTime%60);

fprintf(fp,"%2d:%2d ",CityInfo[i].Flight[j].ArriveTime/60,

CityInfo[i].Flight[j].ArriveTime%60);

fprintf(fp,"%d\n",CityInfo[i].Flight[j].Price);

}

}

fclose(fp);

return 1;

}**调试分析**

1. 此系统实现了稀疏矩阵的基本运算，界面较为友好。
2. 实现了可以多次进行运算的功能,并且有提醒机制，还可以在适当时候退出系统。
3. 再调试过程中，发现有些情况下当输入错误时不能正确的处理。尽管在代码中已经尽量对不规范输入做了限制，但还是存在问题。
4. 输入过程较为繁琐。
5. 算法时间复杂度分析

除了int DelCity(char \*Name)

int DelPath(char \*name)

void FindMinPrice\_Print(int matx[34][34],int PreCity[34],int p\_end,int Vehicle)

void FindMinPrice(int matx[34][34],int p\_start,int p\_end,int Vehicle)

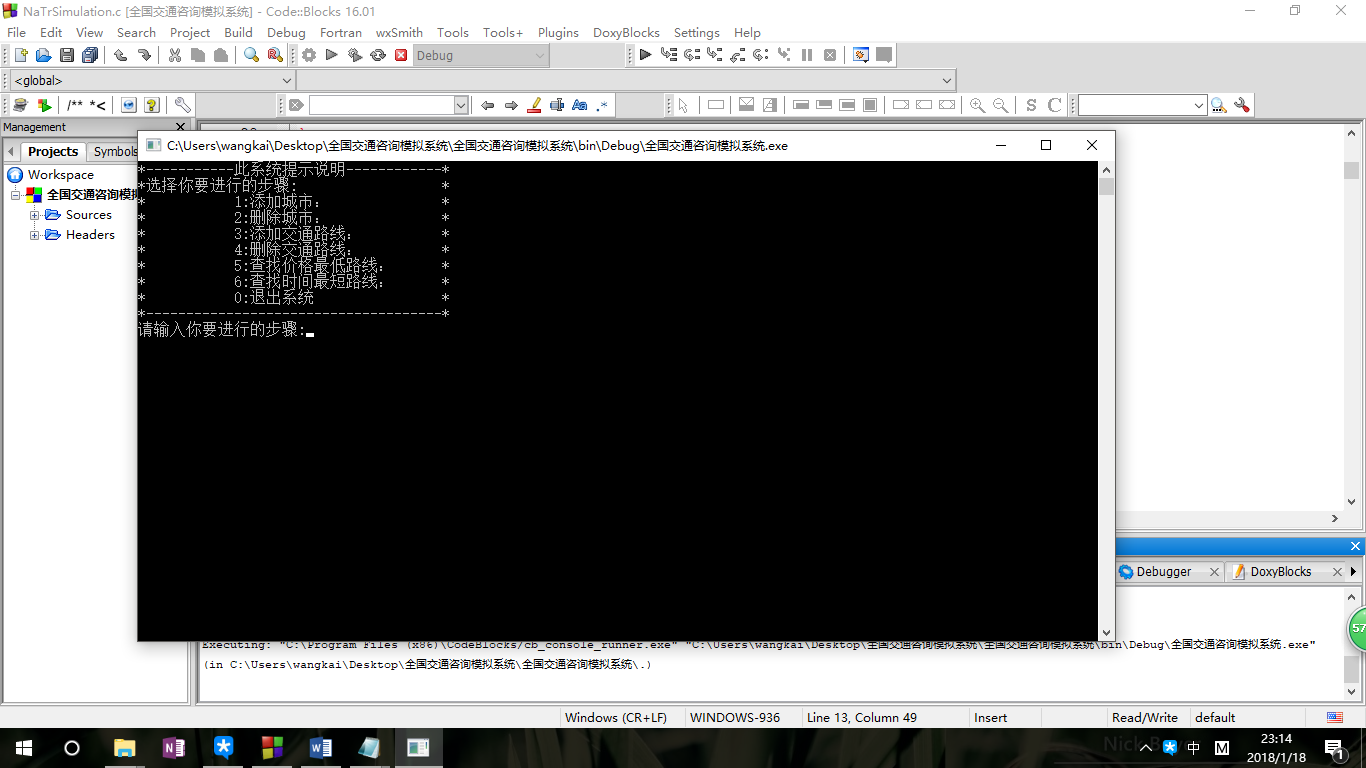
int CalcMinPrice(int DepCity,int DesCity,int Vehicle)

int SaveSysInfo()

这几个函数的时间复杂度为O(n2)外，其余函数的时间复杂度均为O(N)/O(1).

1. **用户使用说明**

用户只需根据系统的的提醒操作，下面系统菜单界面：

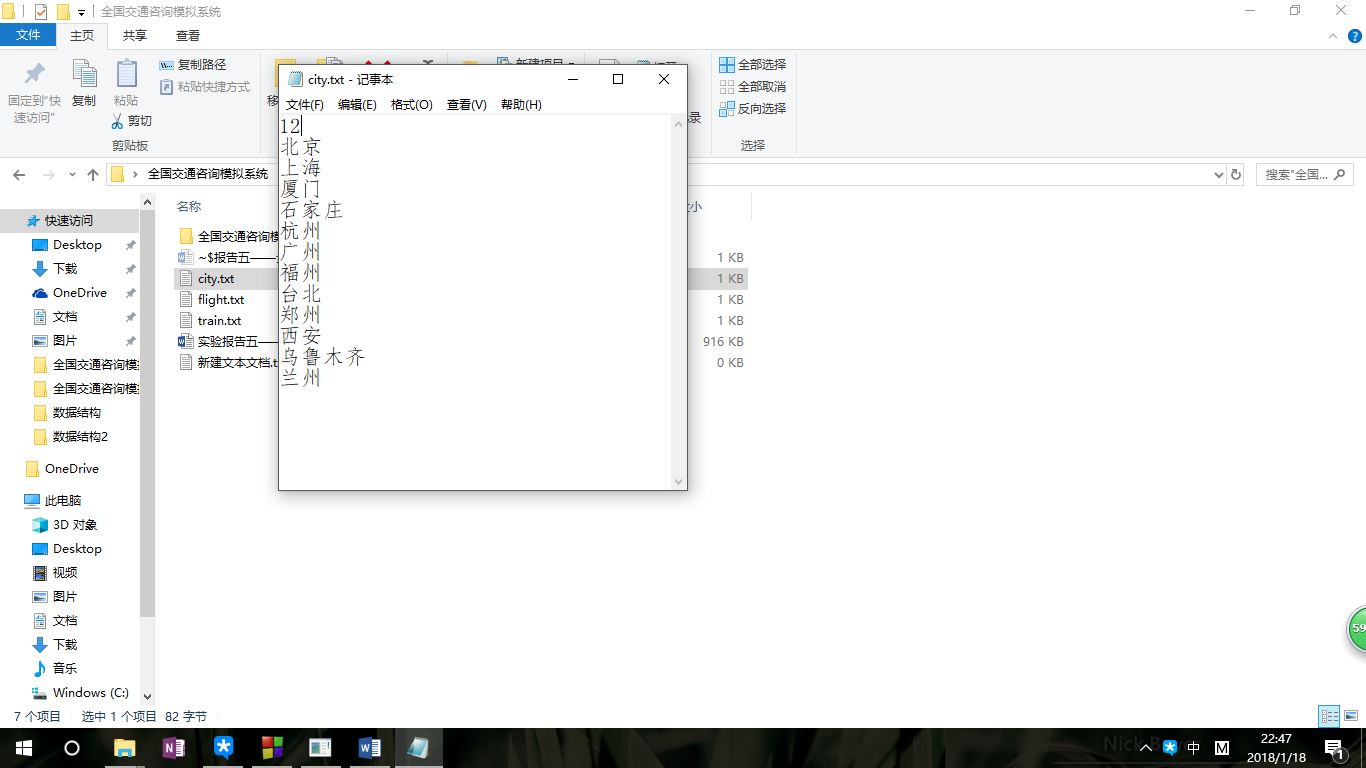
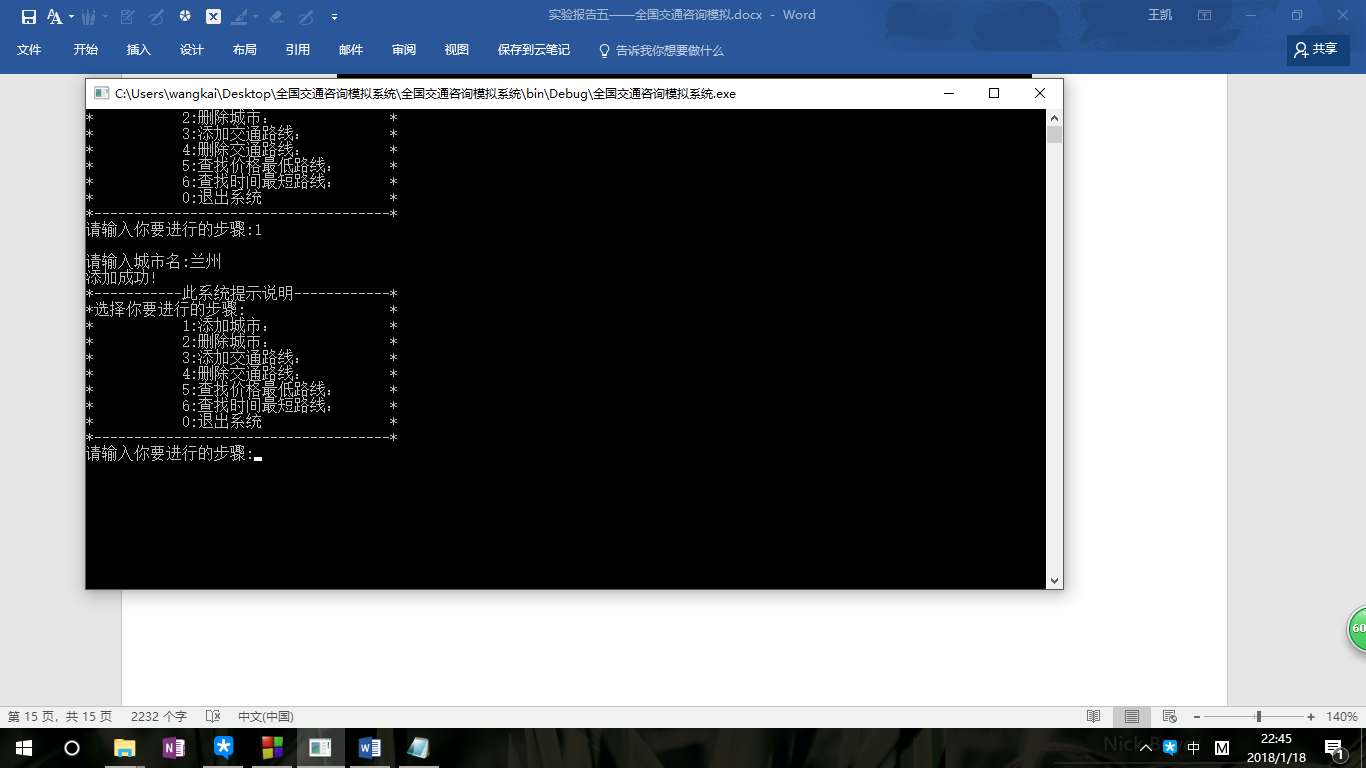
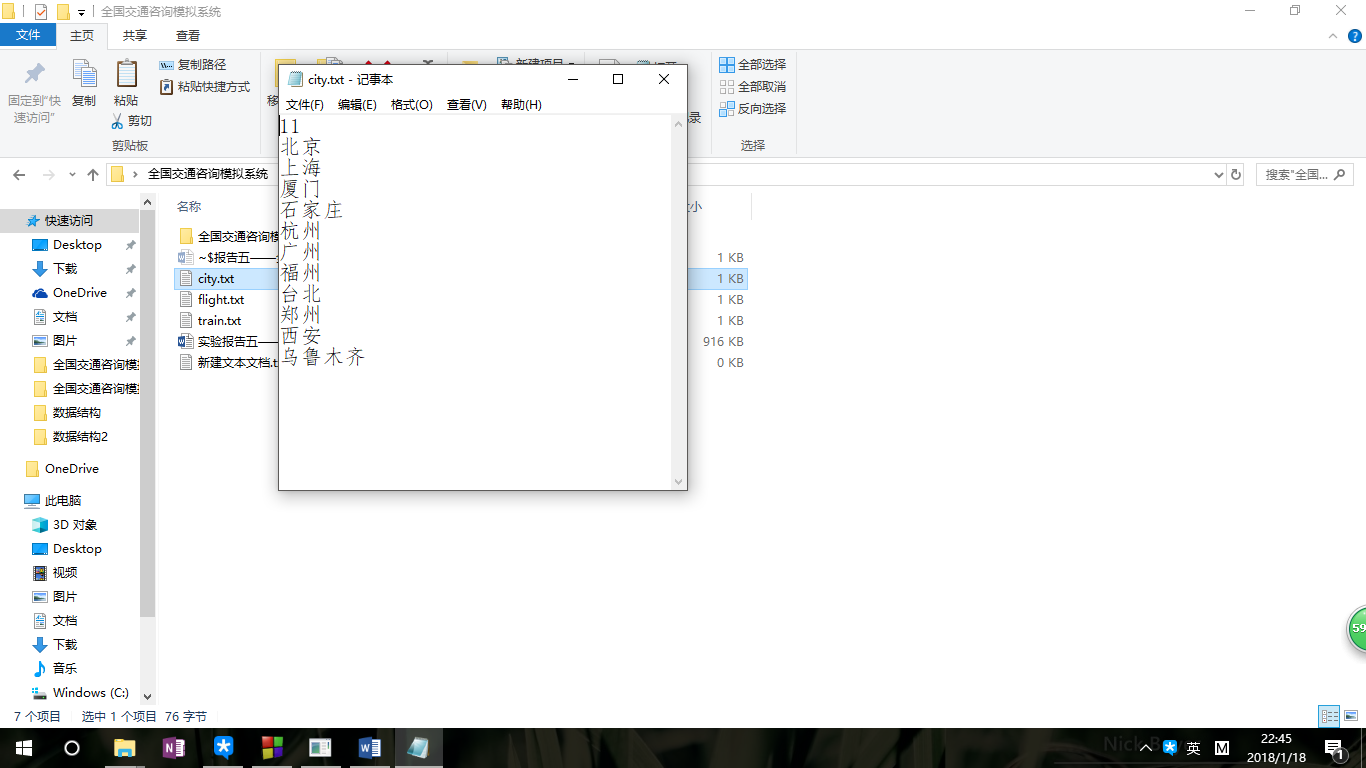


然后根据系统提醒，进行后面的事项。

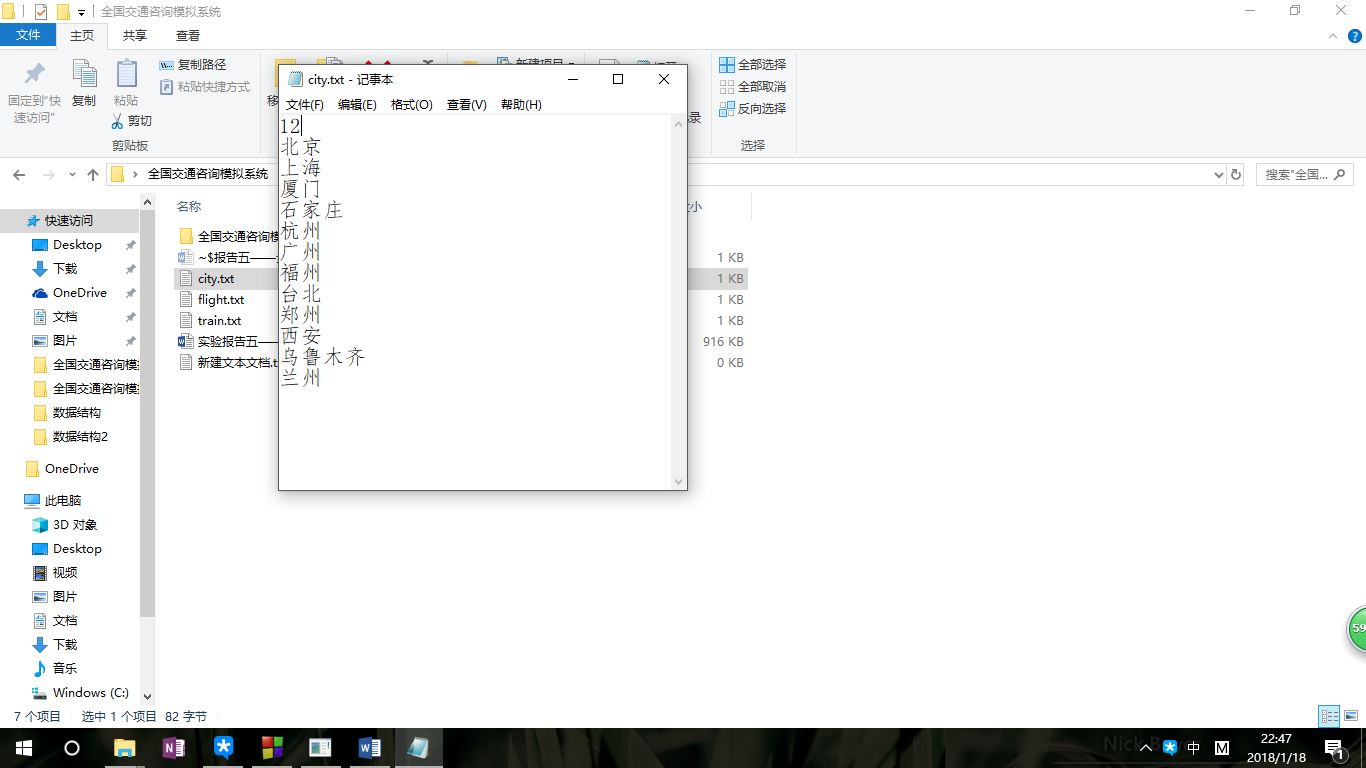
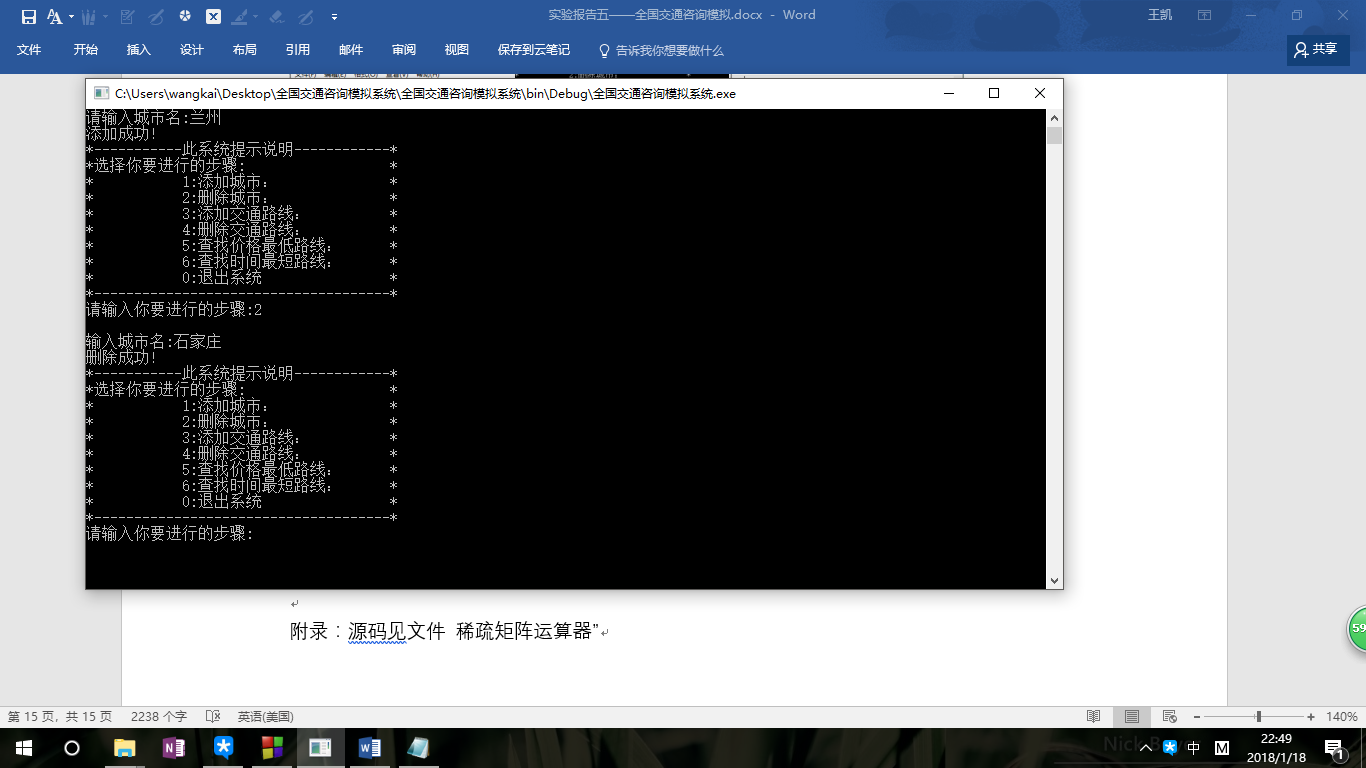
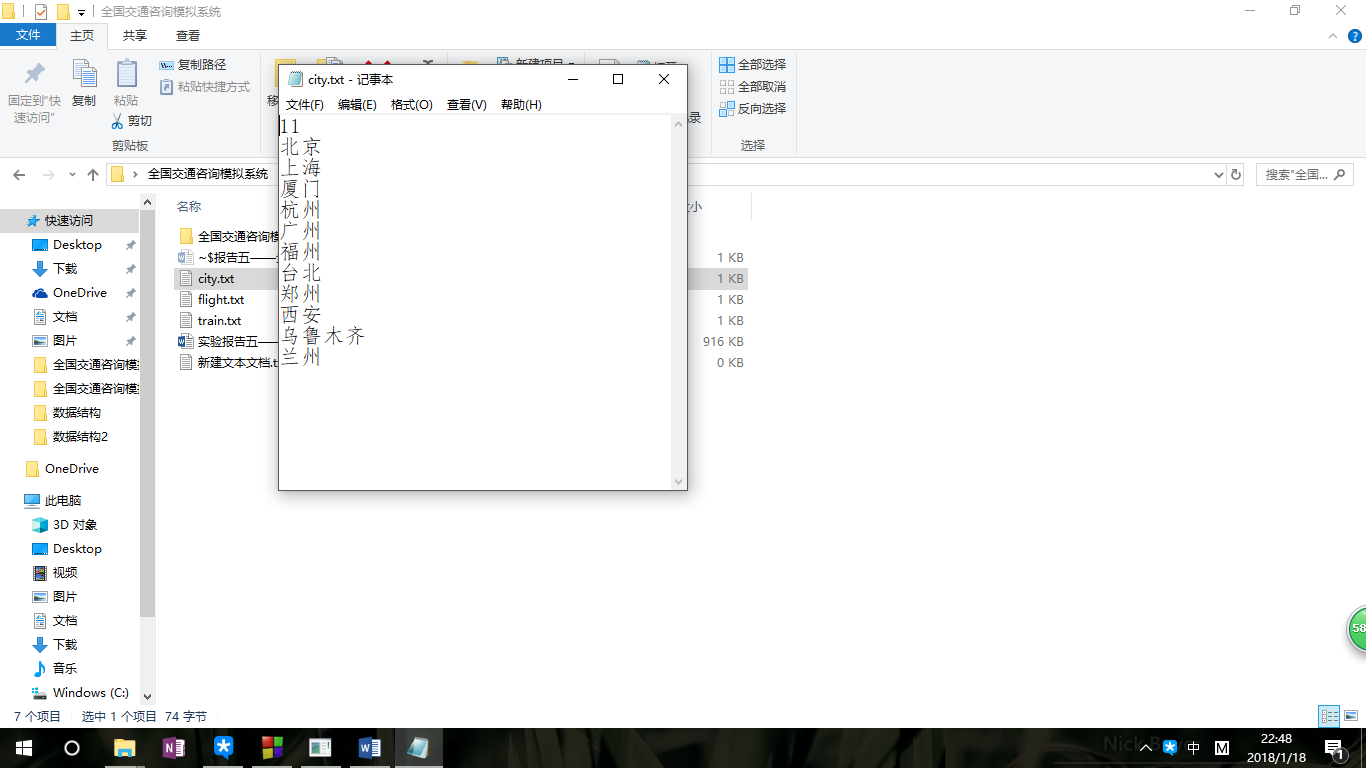
数字、字母采用英文输入法。

若因为错误操作使得txt文件中的内容错误，需自行修改txt文件，再次运行程序。

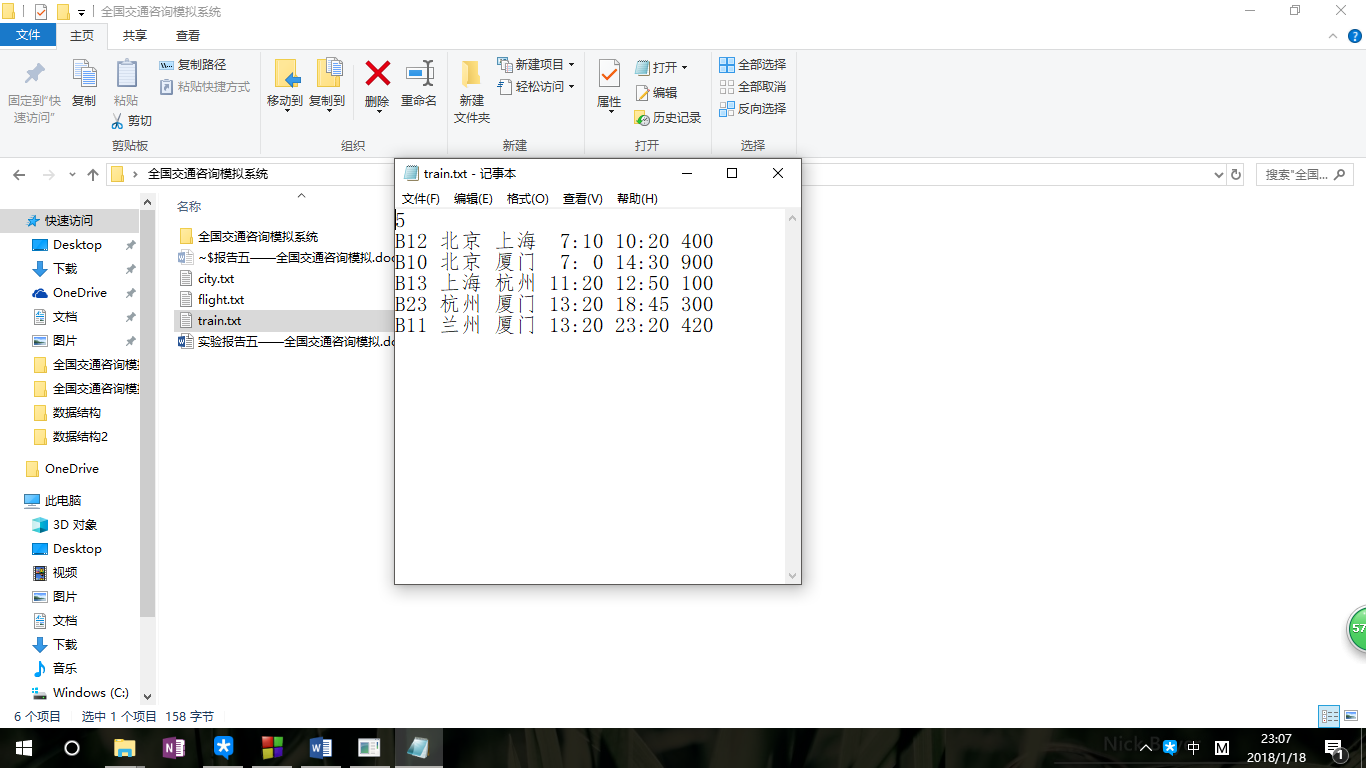
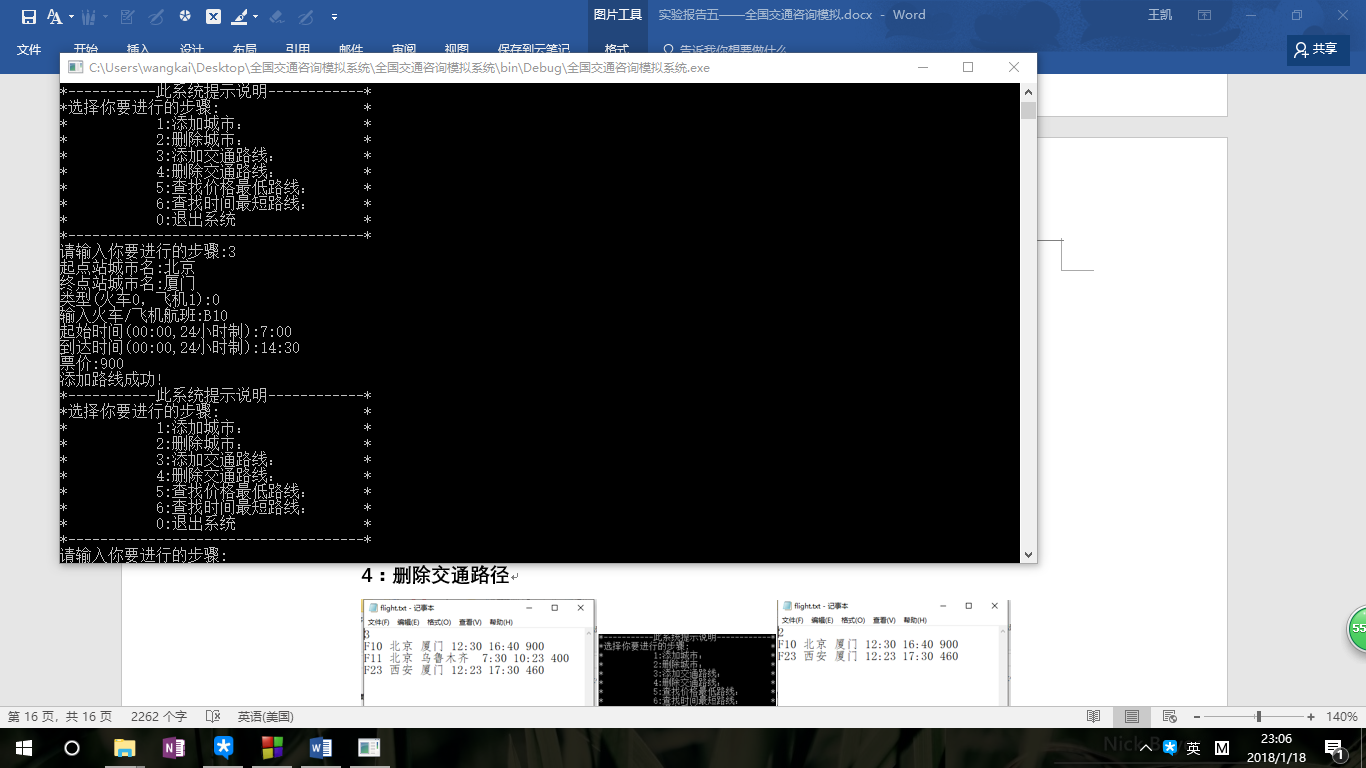
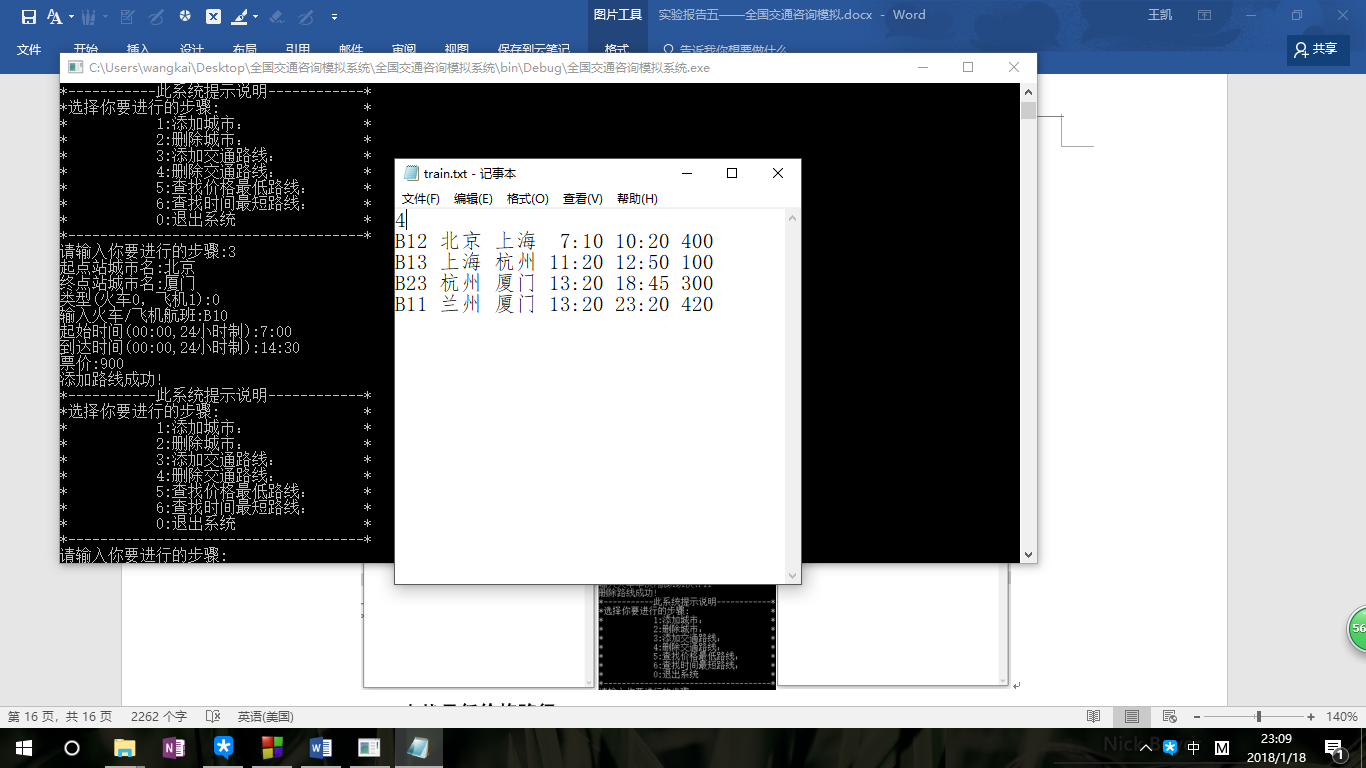
1. **测试结果**
2. **添加城市**



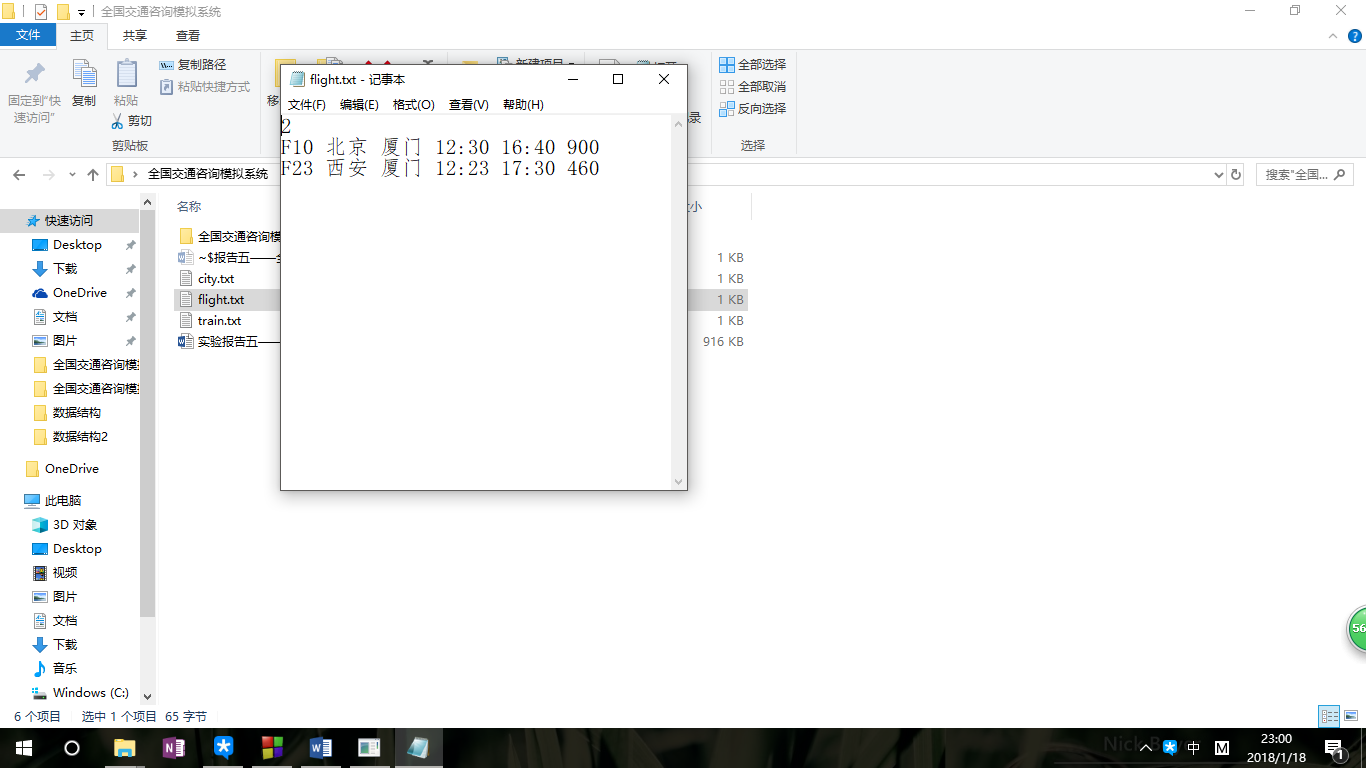
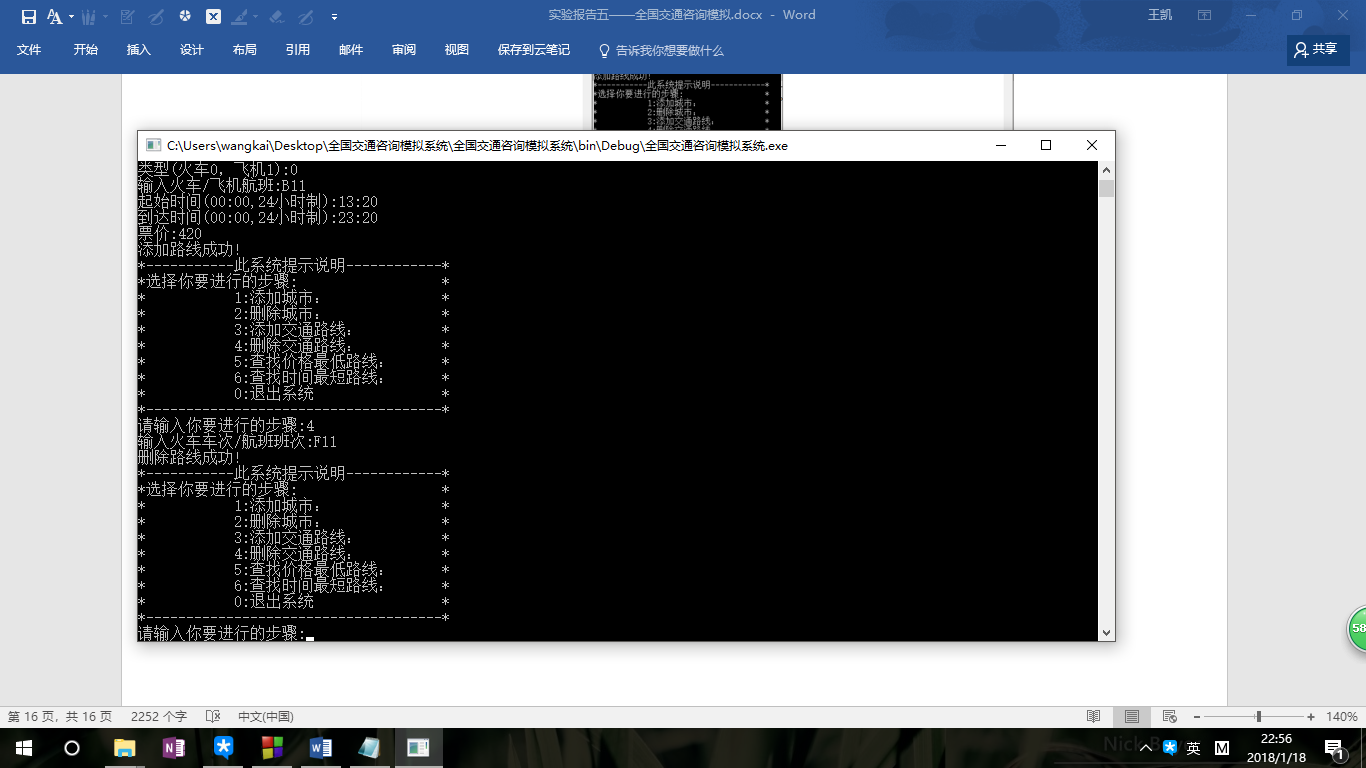
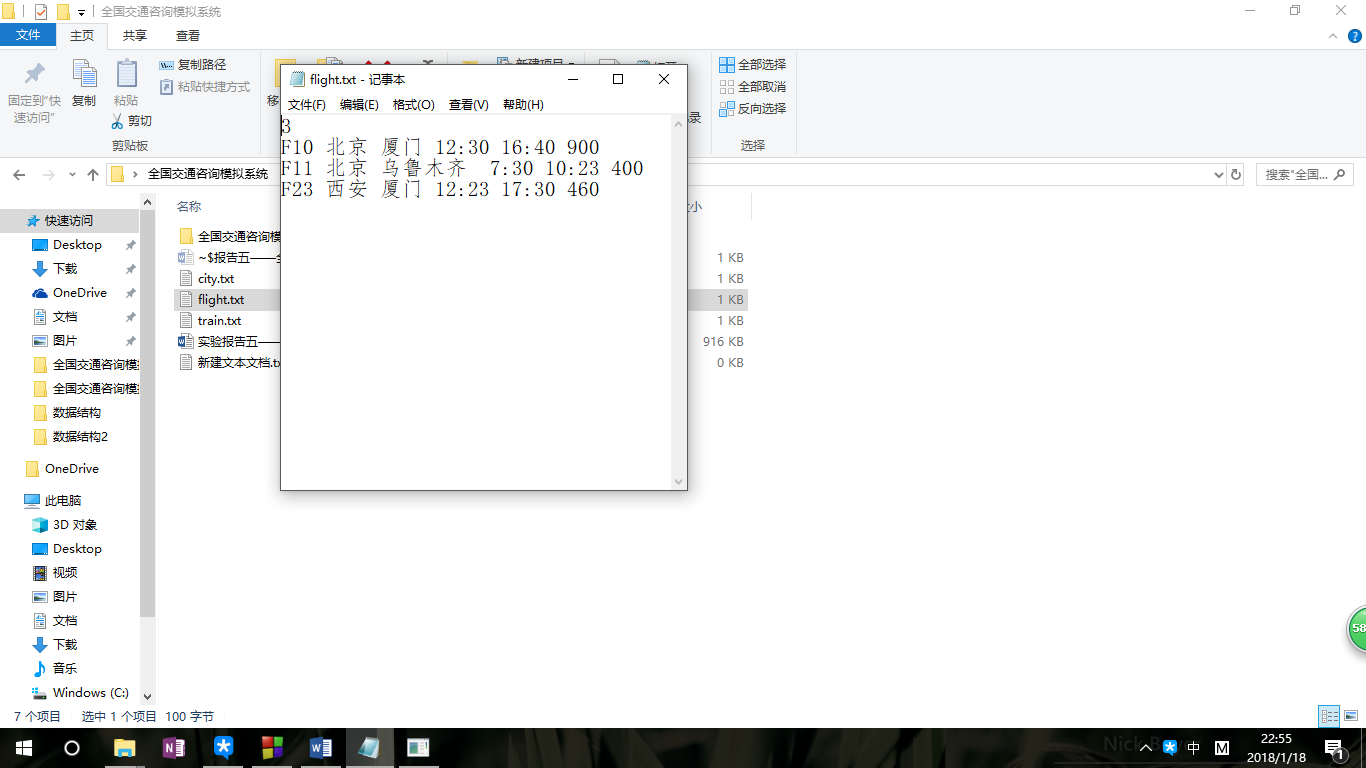
1. **删除城市**

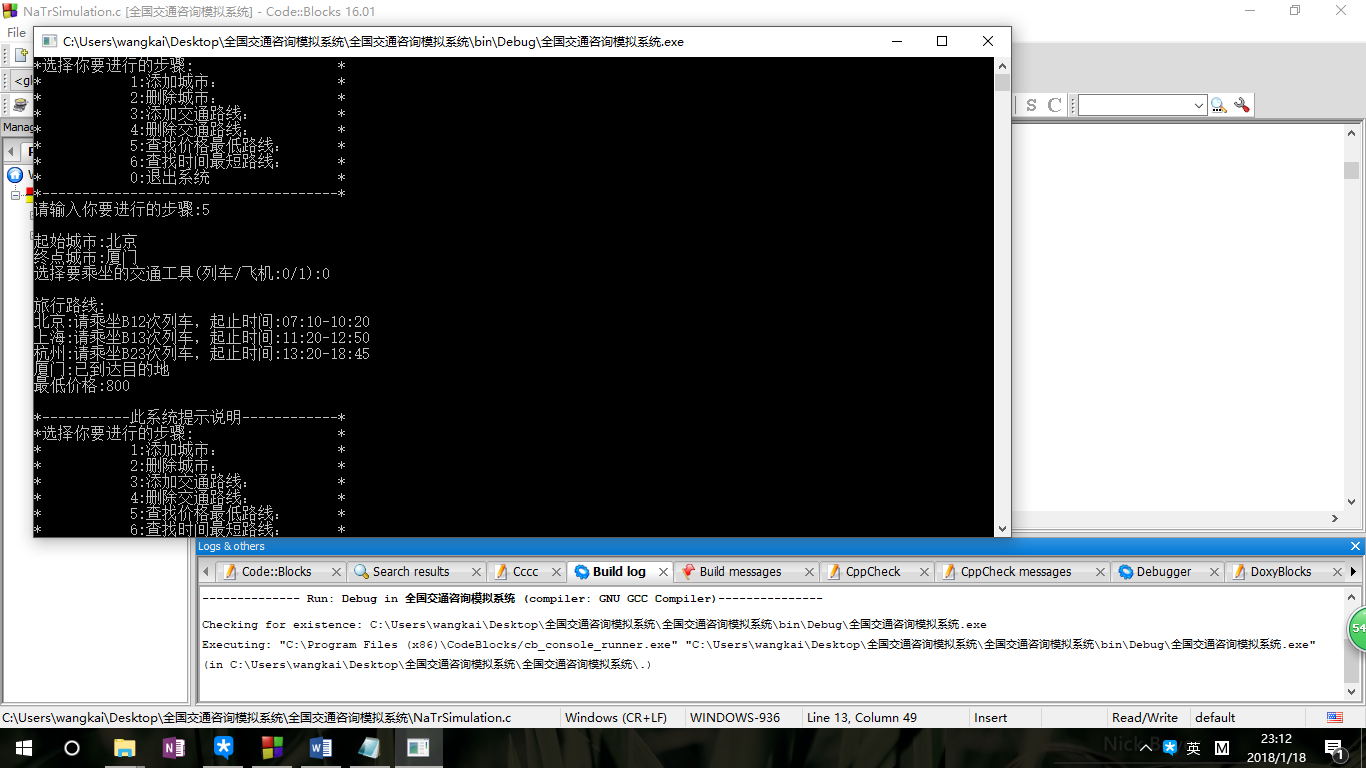
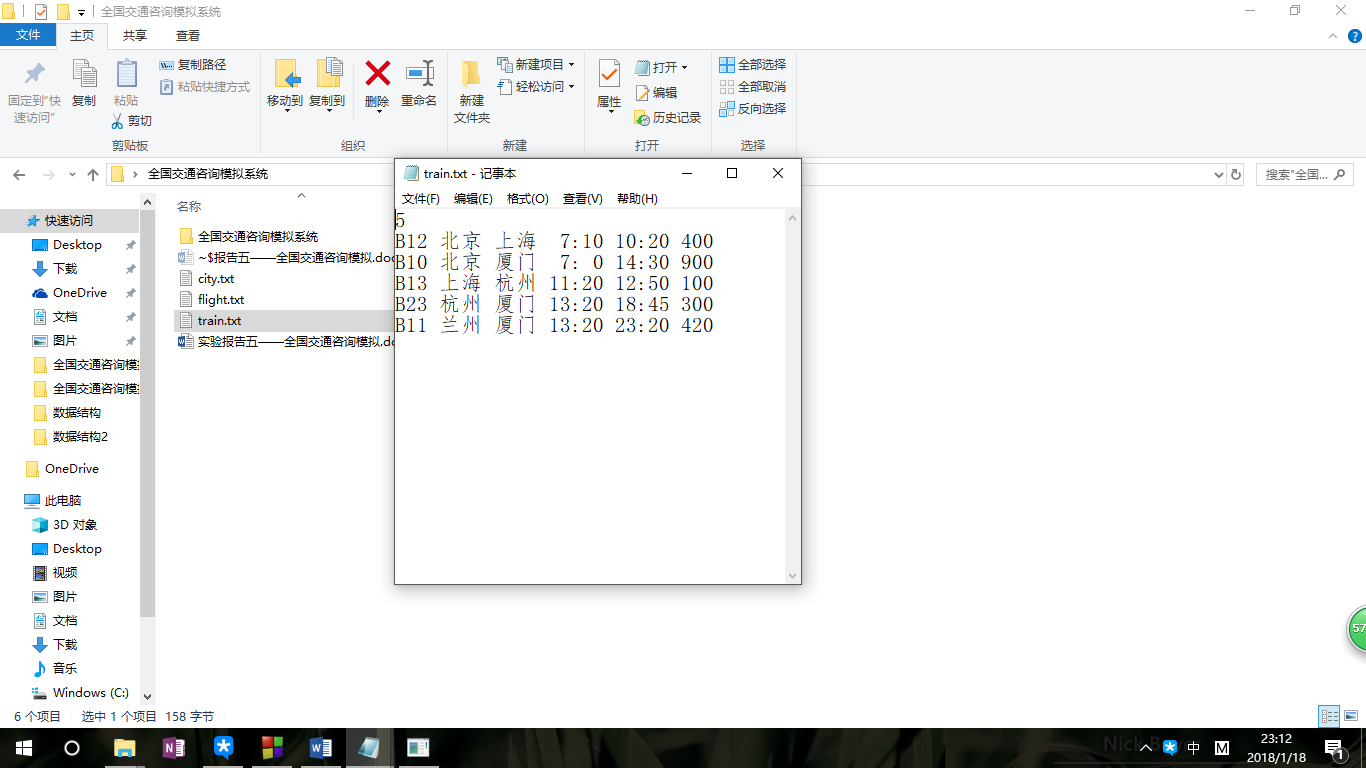
1. **添加路径**



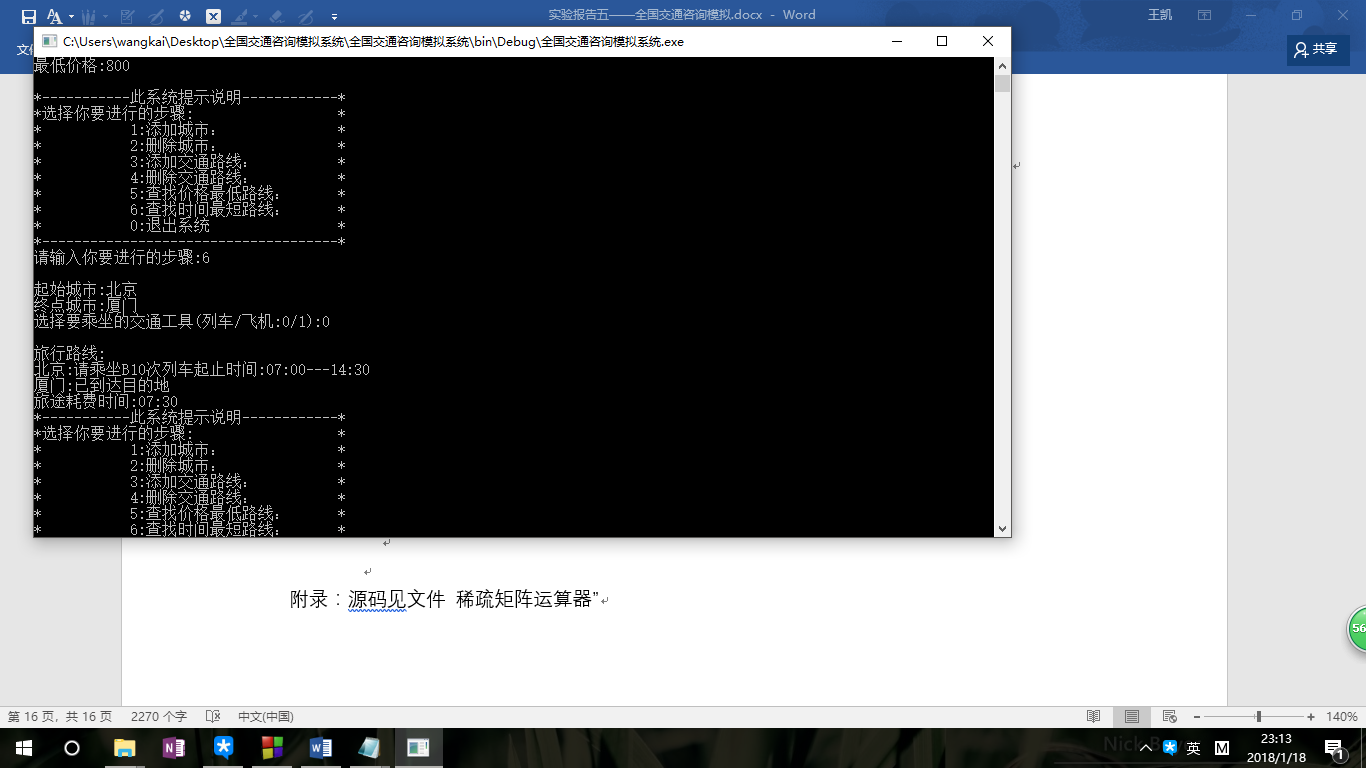
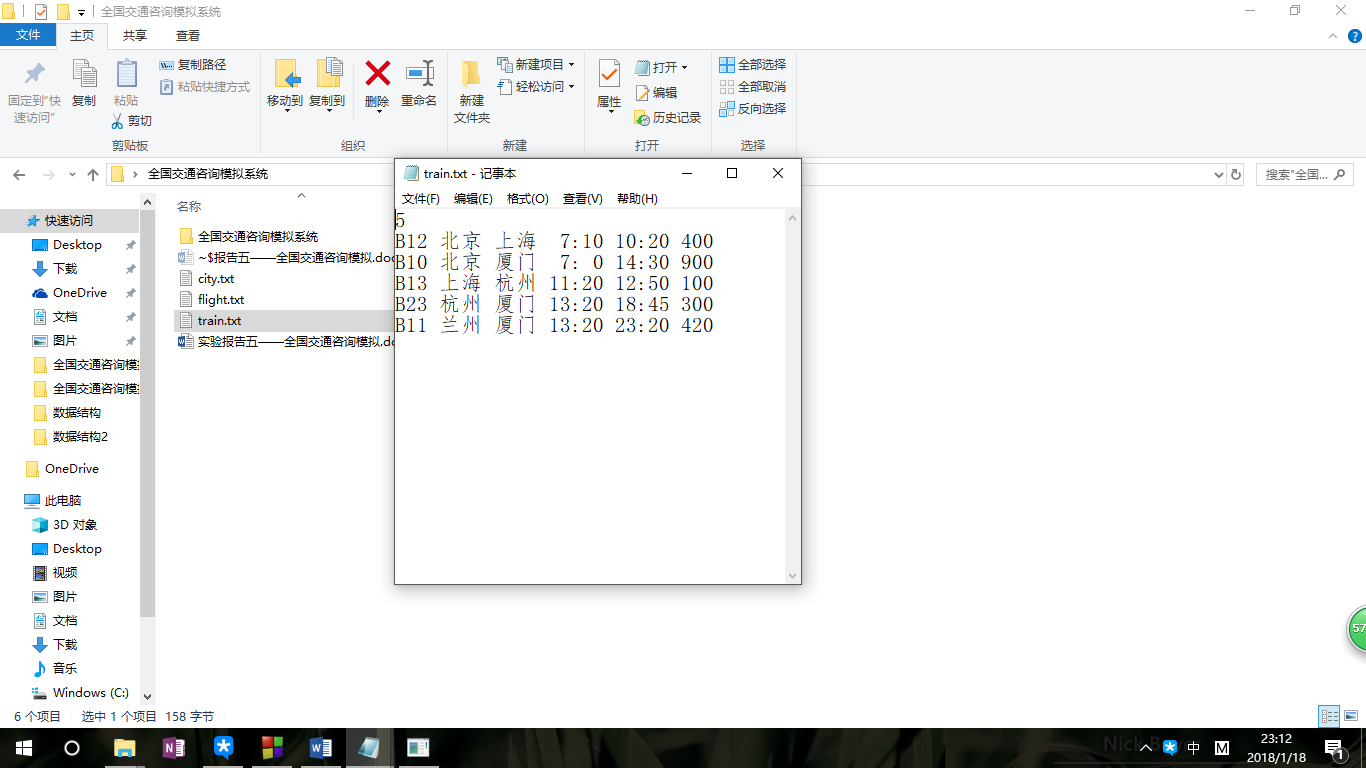
**4：删除交通路径**



**5：查找最低价格路径**



**6：时间最短路径**



附录：源码见文件 全国交通咨询模拟