# 课程编号：B080109010

数据结构课程设计

**总结报告**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **姓名** | **燕江弟** | **学号** | | **20144671** |
| **班级** | **软件1404班** | **指导教师** | | **刘益先** |
| **实验名称** | **数据结构课程设计** | | | |
| **开设学期** | **2016-2017第一学期** | | | |
| **开设时间** | **第10周——第12周** | | | |
| **报告日期** | **2016-11-25** | | | |
| **评定成绩** |  | | **评定人** |  |
| **评定日期** | **2016-11-28** |

**东北大学软件学院**

第一章需求分析

**1.1 建立主程序应用菜单选项**

主程序应用菜单选项包含所实现的所有功能，并且对选项采用数字标识进行选择，对其他错误输入可以进行判别，提示输入错误。

**1.2 导游线路图的创建级景区分布图的输出**

用邻接链表存储景点分布图的信息，（带权无向）图的邻接链表。输出景区景点分布图（邻接矩阵）。图中边的权值∞用32767表示。

**1.3 输出导游线路图**

景区旅游信息管理系统中制订旅游景点导游线路策略，首先通过遍历景点，给出一个入口景点，建立一个导游线路图，导游线路图用有向图表示。

**1.4 输出导游线路图中是否有回路**

景区旅游信息管理系统中，创建好导游路线图后，判断该图中是否存在回路。

**1.5 查找及排序**

* 查找功能： 可以根据用户输入的关键字进行景点的查找，关键字可以在景点名称也可以在景点介绍中。查找成功则返回景点的相关简介，如果查找不成功请给予正确提示。
* 排序功能：按景点欢迎度，景点的岔路数对景点进行排序并打印出来排序顺序。

**1.6 输出两个景点之间最短路径和最短距离**

求出两个景点间的最短路径和最短距离，并且输出道路修建规划图。 算法采用迪杰斯特拉算法。

**1.7 输出道路修建规划图**

道路建设首先要保证能连通所有景点，但又要花最小的代价。

**1.8 输出车辆的进出信息**

**1.8.1 具体需求：**

停车场是一个可以停放n辆汽车，且只有一个大门可供汽车进出。汽车在停车场内按车辆到达时间的先后顺序，依次排列，若车场内已停满n辆车，后来的车只能在门外的便道上等候，一旦有车开走，则排在便道上的第一辆车即可开入；当停车场内某辆车要离开时，在它之后进入的车辆必须先退出车场为它让路，待该辆车开出大门外，其它车辆再按原次序进入车场，每辆停放在车场的车在它离开停车场时必须按它停留的时间长短交纳费用。输出每辆车到达后的停车位置（停车场或便道上），以及某辆车离开停车场时应缴纳的费用和它在停车场内停留的时间。

**1.8.2 停车场的管理流程如下：**

A.当车辆要进入停车场时，检查停车场是否已满，如果未满则车辆进入停车场；如果停车场已满，则车辆进入便道等候。

B.当车辆要求出栈时，先让在它之后进入停车场的车辆退出停车场为它让路，再让该车退出停车场，让路的所有车辆再按其原来进入停车场的次序进入停车场。之后，再检查在便道上是否有车等候，有车则让最先等待的那辆车进入停车场。

**1.8.3 车辆出入清单：**

每一组输入数据包括三个数据项：汽车“到达”或“离去”信息、汽车牌照号码以及到达或离去的时刻。对每一组输入数据进行操作后的输出信息为：若是车辆到达，则输出汽车在停车场内或便道上的停车位置；若是车辆离去，则输出汽车在停车场内停留的时间和应交纳的费用（在便道上停留的时间不收费）。

**1.9**

退出整个程序。

第二章系统设计

2.1总体设计：

2.1.1：**具体数据结构定义**

首先需要创建节点类，邻接边类，无向图类以及停车类。节点类包括了存储的景点名称，景点介绍，景点的欢迎度，景点有误休息区，景点有无厕所以及指向下一条邻接边的指针。邻接边类包括了邻接点的序号，边的权值（即是距离）以及指向下一条边的节点指针。无向图类包括了该图中所需要的节点个数，所需要的邻接边数以及存储具体节点和边的指针。具体如下：

class ArcNode {

public:

int adjvex;

ArcNode \*nextarc;

double weight;

};

class VNode {

public:

string data1;

string data2;

int wel;

bool wc;

bool rest;

ArcNode \*firstarc;

};

class ALGraph {

public:

VNode \*vertices;

int vexnum, arcnum;

ArcNode \*arcNode;

};

class zanlind{

public:

int number;

string time;

};

**2.1.2 ：具体功能实现方法：**

a. 景区景点分布图的创建：

int LocateVex(ALGraph G, string v)

void CreateUDN(ALGraph &G);

b 输出景区景点分布图：

void PrintAdjList(ALGraph &G)，

void OutputGraph(ALGraph G)。

c. 输出导游线路图：

void DFSTraverse(ALGraph G)

d. 输出导游线路图中是否有回路：

void FindInDegree( ALGraph &g)，

void JudgeCir(ALGraph G)。

e.查找及排序：

int LocateW(ALGraph G, int wel)，

void LocateVex2(ALGraph G, string v)，

void Search(ALGraph G,string s)，

void FindInDegree(G)，

void SortWel(ALGraph G)，

bool isInN(ALGraph G,int a[MAX],int n)

void SortN(ALGraph G)，

f. 输出两个景点之间最短路径和最短距离：

void ShortestPath\_DIJ(ALGraph G,int v0, int p[][MAX], int D[])，

bool isInVe(ALGraph G,string va)

void printShortestPath(ALGraph G)。

g.输出道路修建规划图：

void build(ALGraph &G)

void prim(ALGraph G,int v,double arr[MAX][MAX])。

h.输出车辆的进出信息：

bool isInZan(int za[MAX],int number)，

int indexZ(int z[],int n)

void goIn()，

void goOut()。

2.2程序设计

**a. 景区景点分布图的创建:**

void CreateUDN(ALGraph &G) {

G.arcNode=new ArcNode[MAX];

G.vertices=new VNode[MAX];

fstream file("C:\\Users\\22291\_000\\Desktop\\数据结构\\info.txt");

if(file.fail()){

cout << "error open!" << endl; }

int j;

ArcNode \*s, \*t;

cout<<"请输入顶点数和边数：";

cin>>G.vexnum>>G.arcnum;

int i=0;

cout<<endl;

while(!file.eof(){

file >>G.vertices[i].data1>>G.vertices[i].data2 >>

G.vertices[i].wel>> G.vertices[i].rest>>G.vertices[i].wc;

G.vertices[i].firstarc = NULL;

i++;}

cout<<endl;

fstream file1("C:\\Users\\22291\_000\\Desktop\\数据结构\\edge.txt");

if(file.fail()){

cout << "error open!" << endl; }

while(!file1.eof()){

int weight;

string v1, v2;

file1>>v1>>v2>>weight;

int i = LocateVex(G, v1);

int j = LocateVex(G, v2);

s = new ArcNode();

t = new ArcNode();

s->adjvex = j;

s->nextarc = G.vertices[i].firstarc;

s->weight=weight;

G.vertices[i].firstarc =s;

t->adjvex = i;

t->nextarc = G.vertices[j].firstarc;

t->weight=weight;

G.vertices[j].firstarc =t; }}

**b.深度优先遍历算法：具体如下：**

void DFSTraverse(ALGraph G){

bool sta[20];

int v;

for (v = 0; v<G.vexnum; v++){

sta[v] =true; }

stack<int>status;

int n=0;

int num = -1;

int pk;

ArcNode \*e;

cout << G.vertices[0].data1 << "->";

sta[0] =false;

status.push(0);

int aa, bb;

aa = 0;

while (n < G.vexnum-1){

e = NULL;

num = status.top();

e = G.vertices[num].firstarc;

while (e){

if (sta[e->adjvex] == false){

e = e->nextarc; }

else{

status.push(e->adjvex);

cout << G.vertices[e->adjvex].data1<<"->";

aa = e->adjvex;

sta[e->adjvex] = false;

n++;

break; } }

if (e == NULL){

pk = status.top();

bb = pk;

if (aa != bb){

cout << G.vertices[pk].data1<<"->";}

status.pop();}

if (status.top() == 0){

cout << G.vertices[0].data1 << "->";}}

cout << endl; }

**c. 输出车辆进出信息：**

void goIn(){

zanlind zan;

cout<<"车牌号为：";

cin>>zan.number;

cout<<endl;

time\_t t = time(0);

char tmp[64];

strftime(tmp,sizeof(tmp),"%X ",localtime(&t));

zan.time=tmp;

cout<<"进场时间为：";

cout<<tmp<<endl;

if(parking.size()<MAX2){

parking.push(zan);

z[k++]=zan.number;

cout<<"该车已进入停车场在： 1号车道";}

else{

cout<<"停车场已满，请等待其他车辆离开：";

waits.push(zan); }}

void goOut(){

if(parking.size()<=0){

cout<<"停车场为空，没有车要离开！";}

else{

cout<<"请输入您的车牌号：";

int number;

cin>>number;

if(isInZan(z,number)) {

while(parking.top().number!=number) {

cars.push(parking.top());

parking.pop();}

time\_t t = time(0);

char tmp[64];

strftime(tmp,sizeof(tmp),"%X ",localtime(&t));

cout<<"车牌号为："<<parking.top().number<<"的车要离开了,离开时间为"<<tmp;

parking.pop();

while(!cars.empty() && parking.size()<MAX2)

{

parking.push(cars.top());

cars.pop();}

while(parking.size()<MAX2) {

parking.push(waits.front());

waits.pop();}}

else{

cout<<"没有该辆车！请输入正确的车牌号："<<endl; }}}。

第三章系统实现与调试

3.1 景区景点分布图的创建

实现过程：是通过定义的数据结构将节点数据和邻接边数据信息读取后存储在其数据结构上，节点通过头插法进行插入邻接边。

难点：如何定义合理的数据结构来存储数据。

解决方案：定义了三种数据结构来实现，包括节点类，邻边类和无向图类。

3.2输出导游路线图及其图中的回路部分

实现过程：导游路线路采取了深度优先遍历算法来进行所有节点的遍历，然后将遍历顺序值压入声明好的栈中，在访问下一个节点之前输出栈中的元素。输出回路时，在创建时记录下所有节点的度，然后在判断回路时，将其度为0或者是1 的节点压入队列，记录下具体数目，然后将所有与度为1节点相连的节点的度减少1 ，记录下数目。如果前面数目与该数目相加小于总结点数目，则存在回路，反之没有回路。

难点：在输出导游路线图时如何遍历回去，再进行下一个节点的遍历。

解决方案：采用了库函数里的数据结构栈，将前面深度优先遍历顺序记录下来，然后再下一个节点遍历之前将其弹栈输出即可解决。

3.3输出两个景点之间最短路径和最短距离

实现过程：采用了迪杰斯特拉算法，进行距离比较，选出最短路径与距离。

难点：如何进行比较与下一个节点连接时，距离的大小。

解决方案：用一个数组来存储下当前所有遍历路径上的距离之和，每次连接之时进行比较。

3.4输出道路修建规划图

实现过程：采用了无向图最小生成树的思想，用prim算法实现建立已存无向图的最小生成树。即是在图中中选取权值最小的边<u, v>，其中u为集合Vnew中的元素，而v不在Vnew集合当中，并且v∈V（如果存在有多条满足前述条件即具有相同权值的边，则可任意选取其中之一），然后将v加入集合Vnew中，将<u, v>边加入集合Enew中，最后使用集合Vnew和Enew来描述所得到的最小生成树。

难点：在创建中需要反复判断是否回路。

解决方案：每次加入新的节点后判断有无回路。

3.5查找及排序

实现过程：查找时让游客输入含有景点信息的关键字，然后逐一从图的节点遍历，若节点的名称或者是简介中包含此关键字，则输出该景点所有信息，若是多个，则将多个按遍历顺序输出所有节点所有信息。排序时将欢迎度或者是岔路数（即是节点度数）按照冒泡排序进行排序。然后输出节点名称的排序顺序。

难点：如何在节点名称和简介中判断是否含有关键字。

解决方案：采用了C++库函数<cstring>中的find()函数来实现。

3.6输出车辆的进出信息

实现过程：采用栈与队列的思想，当一个汽车来时，都以栈的形式进入停车场（即是谁先来谁在最里面，先进后出），如果停车场满了，那么就在路边等候空位出现，在等待时采用队列思想，谁先来谁先走（即是谁先进入空位）。如果某辆车要离开，则位于前面的车都以出栈的形式出来，先停在空闲处为其让位，在让位时采用进栈思想，谁在前面，谁先进去。然后的离开后，依次出栈的思想进入停车场，这样就不会弄乱次序。

难点：当停车场满了后，有车离开时，如何让等待的车按顺序进入直到又满为止。

解决方案：采用队列的思想，让等待的车存储在队列中，当满后有车离开时，队列中的车依次出队列进入停车场，在此过程中顺便判断停车场是否已满。

第四章系统测试

**4.1. 测试方法：**

输入不同特征的几组数据，验证输出结果是否符合需要。

**4.2. 测试用例（应该给出几组具有不同特征的数据进行测试）：**

(1)节点信息：

a qwer 3 1 1 b asdfg 1 0 1 c rtyuio 4 1 0

邻接边信息：

a b 1 a c 2 b c 5.

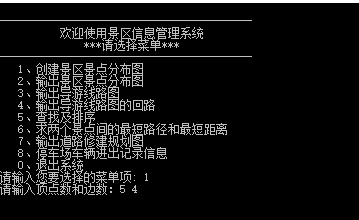
(2)节点信息：

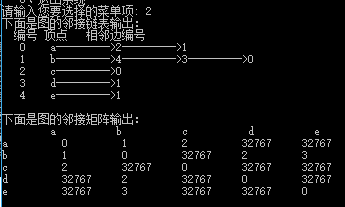
a asdfg 3 1 1 b sdfgh 4 1 0 c asdfg 1 0 1 d werty 2 1 0 eqwert 5 0 0

邻接边信息：

a b 1 a c 2 b d 2 b e 3

**4.3. 测试结果：**

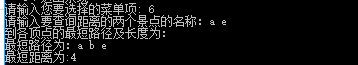


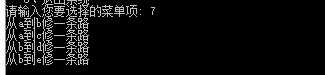


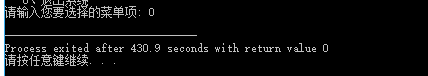












**4.4. 出现的问题及解决办法：**

（1）在输出导游路线图时，栈的存储顺序有问题导致输出顺序存在偏差

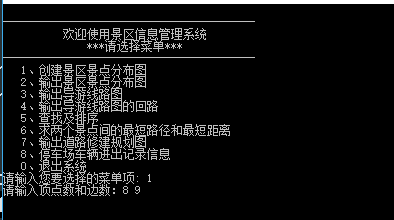
解决办法：深度遍历时，栈存储之前判断有没有访问过该节点，若没有则压栈存储进去。

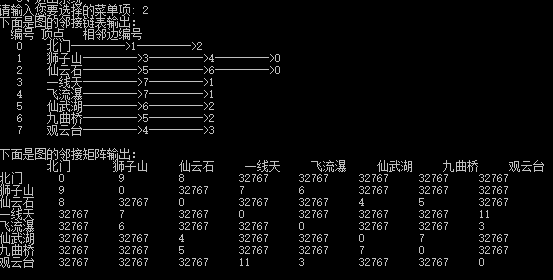
（2）查找两个节点最短距离及输入菜单选项时，输入有误时程序直接终止。

解决方法：在每次输入后，先判断输入是否合理。

第五章结论

**5.1.程序运行的最终结果：**

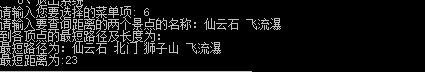


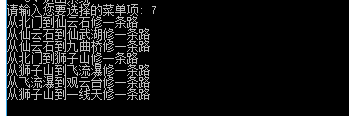


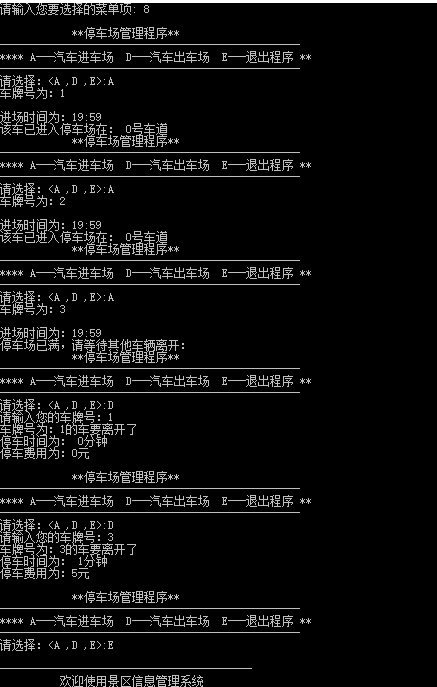


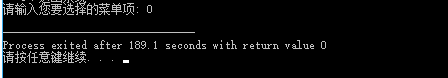












**5.2. 系统实现的功能**：

（1）输出菜单界面

（2）导游线路图的创建：

（3） 输出景区景点分布图：

(4）输出导游线路图及线路图中是否有回路：

（5）.查找及排序：

（6）输出两个景点之间最短路径和最短距离：

（7）输出道路修建规划图：

（8）输出车辆的进出信息：

（9）退出系统。

**5.3. 主要创新点：**

（1）在创建图时采用了头插法在节点之后插入邻接边。

（2）在测试数据时直接从文件里读取，避免了手动反复多次输入。

（3）在输出路线图时自己增加了输出邻接链表。

（4）在输出导游路线图时用了栈存储访问顺序，到返回时直接弹栈输出即可。

（5）在所有输入后，先判断输入是否合理和正确，以增强代码的健壮性。

（6）在停车场问题上，用一个一维数组来标记已在车场中的车牌号。

参考文献

例：

[1] 数据结构、算法与应用：C++语言描述 [Data Structures,Algorithms,and Applications in C++][M].[机械工业出版社](http://book.jd.com/publish/%E6%9C%BA%E6%A2%B0%E5%B7%A5%E4%B8%9A%E5%87%BA%E7%89%88%E7%A4%BE_1.html)出版时间：2000-01-01.

[2] 数据结构(C语言版) [M].北京: 中国铁道出版社, 2011-08-01.

**附录：**

**《数据结构课程设计》实验成绩评定表**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **评价内容** | **具 体 要 求** | **分值** | **得分** |
|
| **平时表现** | **课程设计过程中，无缺勤、迟到、早退现象，学习态度积极。课设过程未做与课设无关的事情。** | **10** |  |
| **报告质量** | **实验报告格式规范，体例符合要求；报告内容充实、正确，实验目的归纳合理到位，思考题回答准确。** | **40** |  |
| **创新性** | **在实践内容实现上，有自己独立的想法，并在功能实现上有一定的创新性。** | **10** |  |
| **实验内容** | **能够按实验要求合理设计并开发出程序，认真记录实验数据，原理及实验结果分析准确，归纳总结充分。程序代码书写规范，简洁。代码块区别显著，有重要位置的注释内容。** | **40** |  |
| **总 分** | | |  |