数据结构实践课报告

题目：最小生成树问题

班级：17052317 姓名：於文卓 学号17061833 完成日期：2018年11月30日

1. **需求分析**
2. 问题描述

若要在n个城市之间建设通信网络，只需要假设n-1条线路即可，如何以最低的经济代价建设这个通信网，是一个网的最小生成树问题

1. 基本要求

1.利用克鲁斯卡尔算法求最小生成树

2.实现MFSet抽象数据类型

3.输出数中各条边以及他们的权值

1. 测试数据：

　/\*

8

a b c d e f g h

14

a b 4

a c 3

b c 5

b d 5

c d 5

b e 9

c h 5

d e 7

d h 4

d f 6

d g 5

e f 3

f g 2

h g 6

\*/

1. **概要设计**
2. 数据结构

|  |
| --- |
| struct Node  {  int x;  int y;  int distance;  char start;  char end;  }; |

1. 基本操作

|  |
| --- |
| int findboss(int x)//找到主导的节点  void merge(int a, int b)//合并  bool cmp( Node a, Node b)//比较 |

1. **本程序包含3个模块：**

(1)主程序模块：

int main(){

输入数据；

处理命令；

输出结果；

return 0;

}

(2)图的存储模块——包含起始点，重点，长度的结构体数组；

（3）操作单元模块——实现合并和查找；

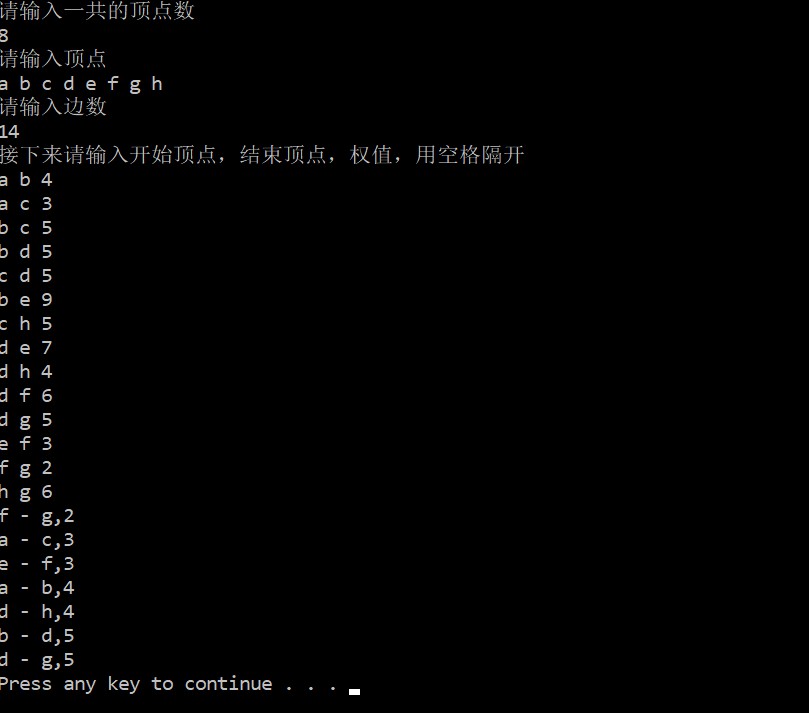
各模块之间的调用关系如下：

主程序模块 图的存储模块 操作单元模块

1. **运算单元模块部分实现代码**

|  |
| --- |
| **int findboss(int x)**  **{**  **int boss = x;**  **while(boss\_list[boss] != boss)**  **{**  **boss = boss\_list[boss];**  **}**  **return boss;**  **}**  **void merge(int a, int b)**  **{**  **a = findboss(a);//a说我问问老板的意见**  **b = findboss(b);//b说我问问老板的意见**  **if (a != b)//如果a和b的老板是同一个 那么他们本来就在一个集合里**  **{**  **boss\_list[a] = b;//否则的话就合并了**  **}**  **}**  **bool cmp( Node a, Node b)**  **{**  **return a.distance < b.distance;**  **}** |

1. **算法分析和代码调试**
2. **运用到了并查集的思想**
3. **用到了c++的map，方便直接给输入的字符定位**
4. **注意要用getchar（）吸收掉多余的回车，防止干扰**
5. **用户手册**
6. **本程序运行环境为win10操作系统，执行文件为main.exe**
7. **用户界面和测试数据如下**

****

1. **附件源代码**
2. /\*
3. \* @Author: TD21forever
4. \* @Date: 2018-11-29 21:48:02
5. \* @Last Modified by: TD21forever
6. \* @Last Modified time: 2018-12-01 17:03:50
7. \*/
8. #include<stdio.h>
9. #include<stdlib.h>
10. #include<map>
11. #include<string>
12. #include <algorithm>
13. using namespace std;
14. /\*
15. 8
16. a b c d e f g h
17. 14
18. a b 4
19. a c 3
20. b c 5
21. b d 5
22. c d 5
23. b e 9
24. c h 5
25. d e 7
26. d h 4
27. d f 6
28. d g 5
29. e f 3
30. f g 2
31. h g 6
32. \*/
33. struct Node
34. {
35. int x;
36. int y;
37. int distance;
38. char start;
39. char end;
40. };
41. Node arc[5000];
42. int boss\_list[1050];//用来检测是否属于同一个
43. int findboss(int x)
44. {
45. int boss = x;
46. while(boss\_list[boss] != boss)
47. {
48. boss = boss\_list[boss];
49. }
50. return boss;
51. }
52. void merge(int a, int b)
53. {
54. a = findboss(a);//a说我问问老板的意见
55. b = findboss(b);//b说我问问老板的意见
56. if (a != b)//如果a和b的老板是同一个 那么他们本来就在一个集合里
57. {
58. boss\_list[a] = b;//否则的话就合并了
59. }
60. }
61. bool cmp( Node a, Node b)
62. {
63. return a.distance < b.distance;
64. }
65. int main(int argc, char const \*argv[])
66. {
67. int N, n;//N是顶点的数量，n是边的数量
68. char start, end;
69. int weight;
70. map<char, int> mp;
71. map<char, int> :: iterator it;
72. printf("请输入一共的顶点数\n");
73. scanf("%d", &N);
74. printf("请输入顶点\n");
75. for (int i = 1; i <= N; ++i)
76. {
77. char vex;
78. scanf("%s", &vex);//用%s
79. mp[vex] = i;//为方便操作 这里用map给顶点名赋一个int类型的位置
80. }
81. for (int i = 1; i <= N; ++i)
82. {
83. boss\_list[i] = i;//初始化 每一个顶点都是独立的顶点
84. }
85. printf("请输入边数\n");
86. scanf("%d", &n);
87. printf("接下来请输入开始顶点，结束顶点，权值，用空格隔开\n");
88. for (int j = 0; j < n; ++j)
89. {
90. getchar();//吸收掉多余的回车
91. scanf("%c", &start);
92. getchar();
93. scanf("%c", &end);
94. scanf("%d", &weight);
95. arc[j].x = mp[start];
96. arc[j].y = mp[end];
97. arc[j].distance = weight;
98. arc[j].start = start;
99. arc[j].end = end;
100. }
101. sort(arc, arc + n, cmp);//对节点进行排序
102. for (int i = 0; i < n ; ++i)
103. {
104. if (findboss(arc[i].x) != findboss(arc[i].y))//因为已经从小到大排序了 只要他们的老板不是同一个 就拉他们入伙
105. {
106. merge(arc[i].x, arc[i].y);
107. printf("%c - %c,%d\n", arc[i].start, arc[i].end, arc[i].distance);
108. }
109. }
110. return 0;
111. }