项目说明文档

数据结构课程设计

——排课软件

作 者 姓 名： 伊啸

学 号： 1951220

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

目 录

[1 分析 1](#_Toc495668153)

[1.1 背景分析 1](#_Toc495668154)

[1.2 功能分析 1](#_Toc495668155)

[2 设计 2](#_Toc495668156)

[2.1 数据结构设计 2](#_Toc495668157)

[2.2 类结构设计 2](#_Toc495668158)

[2.3 成员与操作设计 2](#_Toc495668159)

[2.4 系统设计 4](#_Toc495668160)

[3 实现 4](#_Toc495668161)

[3.1 插入功能的实现 4](#_Toc495668162)

[3.1.1 插入功能流程图 4](#_Toc495668163)

[3.1.2 插入功能核心代码 5](#_Toc495668164)

[3.1.3 插入功能截屏示例 5](#_Toc495668165)

[3.2 删除功能的实现 7](#_Toc495668166)

[3.2.1 删除功能流程图 7](#_Toc495668167)

[3.2.2 删除功能核心代码 7](#_Toc495668168)

[3.2.3 删除功能截屏示例 8](#_Toc495668169)

[3.3 查找功能的实现 10](#_Toc495668170)

[3.3.1 查找功能流程图 10](#_Toc495668171)

[3.3.2 查找功能核心代码 10](#_Toc495668172)

[3.3.3 查找功能截图示例 11](#_Toc495668173)

[3.4 修改功能的实现 13](#_Toc495668174)

[3.4.1 修改功能流程图 13](#_Toc495668175)

[3.4.2 修改功能核心代码 13](#_Toc495668176)

[3.4.3 修改功能截屏示例 14](#_Toc495668177)

[3.5 统计功能的实现 15](#_Toc495668178)

[3.5.1 统计功能流程图 15](#_Toc495668179)

[3.5.2 统计功能核心代码 15](#_Toc495668180)

[3.5.3 统计功能截屏示例 16](#_Toc495668181)

[3.6 总体系统的实现 16](#_Toc495668182)

[3.6.1 总体系统流程图 16](#_Toc495668183)

[3.6.2 总体系统核心代码 17](#_Toc495668184)

[3.6.3 总体系统截屏示例 17](#_Toc495668185)

[4 测试 19](#_Toc495668186)

[4.1 功能测试 19](#_Toc495668187)

[4.1.1 插入功能测试 19](#_Toc495668188)

[4.1.2 删除功能测试 19](#_Toc495668189)

[4.1.3 查找功能测试 20](#_Toc495668190)

[4.1.4 修改功能测试 20](#_Toc495668191)

[4.1.5 统计功能测试 21](#_Toc495668192)

[4.2 边界测试 21](#_Toc495668193)

[4.2.1 初始化无输入数据 21](#_Toc495668194)

[4.2.2 删除头结点 22](#_Toc495668195)

[4.2.3 删除后链表为空 22](#_Toc495668196)

[4.3 出错测试 23](#_Toc495668197)

[4.3.1 考生人数错误 23](#_Toc495668198)

[4.3.2 操作码错误 23](#_Toc495668199)

[4.3.3 插入位置不存在 23](#_Toc495668200)

[4.3.4 删除考号不存在 24](#_Toc495668201)

[4.3.5 查找考号不存在 24](#_Toc495668202)

[4.3.6 修改考号不存在 25](#_Toc495668203)

# 1 分析

## 1.1 背景分析

大学的每个专业都要进行排课。假设任何专业都有固定的学习年限，每学年含两学期，每个专业开设的课程都是确定的，而且课程在开设时间的安排必须满足先修关系。每门课程有哪些先修课程是确定的。每门课恰好占一个学期，假定每天上午与下午各有5节课。是在这样的前提下设计一个教学计划编制程序。

## 1.2 功能分析

1 输入数据包括：每个学期所开的课程数（必须使每学期所开的课程数之和与课程总数相等），课程编号，课程名称，周学时数，指定开课学期，先决条件。如指定开课学期为0，表示有电脑自行指定开课学期。

2 如输入数据不合理，比如每学期所开的课程数值和与课程总数不相等，应显示适当的提示信息。

3 用文本文件存储输入数据，并且读入计算机。

4 用文本文件存储产生的各学期的课表。

排课要求与课程信息：

假设周一至周五上课，每天上10节课，第1大节为第1-2节课，第二大节为第3-5节课，第3大节为第6-7节课，第4大节为8-10节课，在排课时，如一门课程有3节课，则优先安排3节课连续上；如3节课连续无法按排，再优先安排两节课连续上，最后再安排单节课上的情况；如果一门课程需要安排上两天，为教学效果较好，最好不安排在相邻的两天，比如优先安排相隔2天上课，设weekday表示当前安排上课的工作日期，下一次排课的工作日是：weekday=（weekday+2-5）？（weekday+2-5）：（weekday+2）；

存储课程信息的文本文件中，输出结果也附在文件中。

# 2 设计

## 2.1 数据结构设计

如上功能分析所述，该系统要求大量的增加、删除、修改操作，而链表进行增加、删除等操作十分简便，因此考虑使用链表数据结构。同时，为了实现简易，在第一个结点之前附加一个头结点，这样就使得增加或者删除头结点与处理其他结点方法相同，使得程序简洁。

## 2.2 类结构设计

主要的有存储课程信息的节点，struct类型，便于取每个课程的信息，以及图的类，存储课程节点，以及进行各种图的操作。

## 2.3 成员与操作设计

**链表结点类（Course）**

**公有成员：**

int \_Weekday1 = -1; //课程的第一个所在工作日

int \_Time1 = -1; //课程的第一个所在时间段

int \_Weekday2 = -1;

//课程的第二个所在工作日（有些课程有，有些没有）

int \_Time2 = -1; //课程的第二个所在时间段

int count = 0;

//入度数组兼入度为零顶点栈，即该课程的先修课程数量，用于拓扑排序

int counter = 0;

//值和count相同，但用于安排课程，因为count在拓扑排序时值已经改变

string courseNumber; //课程序号

string courseName; //课程名称

string advancedCourse1, advancedCourse2;//先修课名称，若没有则为""

int courseHours; //课时

ScheduledTime scheduledTime; //ScheduledTime类型，用于存储课程学期

Edge\* adj; //邻接表的头指针

**公有操作：**

Course() :courseHours(0), adj(NULL) {} //构造函数

Course(ScheduledTime scheduled\_time, string number = "NULL", string name = "NULL", string advanced\_course1 = "NULL",

string advanced\_course2 = "NULL", int hours = 0, Edge\* edge = NULL);

//构造函数

Course(const Course& vertex) { //复制构造函数

courseNumber = vertex.courseNumber;

courseName = vertex.courseName;

advancedCourse1 = vertex.advancedCourse1;

advancedCourse2 = vertex.advancedCourse2;

courseHours = vertex.courseHours;

adj = vertex.adj;

adj->dest = vertex.adj->dest;

adj->link = vertex.adj->link;

scheduledTime = vertex.scheduledTime;

}

bool operator==(const Course& vertex) {//判断两课程节点是否相同

return courseNumber == vertex.courseNumber ? true : false;

}

**链表类（Graphlnk）**

**私有成员：**

int maxVertices; //最大节点数量

int numEdges; //边数量

int numVertices; //节点数量

Course\* NodeTable; //存储各个课程的数组

Course\* TopologicalArray; //拓扑排序后存储拓扑序列的数组

int getVertexPos(const string number) {

//通过课程序号得到课程在NodeTable数组中的位置

for (int i = 0; i < numVertices; i++)

if (NodeTable[i].courseNumber == number)

return i;

return -1;

}

**公有操作：**

Graphlnk(int sz = DefaultVertices); //构造函数

~Graphlnk(); //析构函数

void readFile(); //读文件并

bool insertVertex(const Course& course);//插入新课程节点

bool insertEdge(int v, int w); //在两课程节点间插入边，由v指向w

int getFirstNeightbor(int v);

//取下标为v课程的邻接表的第一个节点的下标

int getNextNeighbor(int v, int w);

//取下标为v课程的邻接表的除w以外的第一个节点的下标

## void Topological();

## //进行拓扑排序并安排课表，写入文本文件

## 2.4 系统设计

系统首先读取文本文件中的数据，将其存储到各个课程节点中，再将节点构建图，进行拓扑排序，排序后再根据排课的逻辑进行安排。

# 3 实现

## 3.1 拓扑排序的实现

### 3.1.1 拓扑排序功能流程图



### 3.1.2 拓扑排序核心代码

for (i = 0; i < n; i++) { //循环n次，n课程总的数量

if (top == -1) { //中途栈空

cout << "There are loops in the network!" << endl;

return;

}

else {

v = top;

top = NodeTable[top].count; //退栈

TopologicalArray[number] = NodeTable[v]; //存入拓扑数组

number++;

w = getFirstNeightbor(v); //取v的第一个邻接顶点

while (w != -1) {

if (--NodeTable[w].count == 0) { //扫描出边表

NodeTable[w].count = top;

top = w; //入度为零的节点入栈

}

w = getNextNeighbor(v, w);

}

}

}

## 3.2 排课功能的实现

### 3.2.1 排课功能流程图

若课时为3，则优先排入时间段为3的课；

流程图以课时为5为例，4和6类似



### 3.2.2 排课功能核心代码

int count = 0, target = 0, record\_j = -1, record\_k = -1;

/\*count表示安排好了几个时间段，为2时退出；target表示查找了几轮，如果target>2，说明查找了3轮都安排不下该课，则要安排到下一学期\*/

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

if (j > 4) {

j = j - 5;

target++;

}

for (k = 0; k < 4; k++) {

if (Time[j][k] == 0) {

if (!count) {

Time[j][k] = 1;

record\_k = k; //记录工作日和时间段

record\_j = j;

count++;

break;

}

else {

if (record\_k % 2) {

if (k % 2 == 0) {

Time[j][k] = 1;

count++;

break;

}

}

else {

if (k % 2 == 1) {

Time[j][k] = 1;

count++;

break;

}

}

}

}

}

if (count == 1) {

for (l = 0; l < 4; l++) {

int t = ((j + 2 - 5) >= 0) ? (j + 2 - 5) : (j + 2);

//检查隔一天后是否有空余时间段安排该课程

if (Time[t][l] == 0 && l % 2 != record\_k % 2)

break;

}

if (l == 4) {

count--;

Time[j][k] = 0;

}

}

if (count == 1)

j++; //通过循环使j变为两天后

if (target > 2) {

//循环3遍仍然没有结果，说明这个学期已经安排不下了

Time[record\_j][record\_k] = 0;

TopologicalArray[i].scheduledTime.Semester++;

//该课程学期加一

TopologicalArray[i].scheduledTime.target = 0;

//target变为0，说明其学期未安排好

break;

}

if (count == 2) {

//已经安排好了两天的课程，记录两个工作日和时间段

TopologicalArray[i].\_Weekday1 = record\_j;

TopologicalArray[i].\_Time1 = record\_k;

TopologicalArray[i].\_Weekday2 = j;

TopologicalArray[i].\_Time2 = k;

break;

}

}

break;

## 3.3 总体系统的实现

### 3.3.1 总体系统流程图



### 3.3.2 总体系统核心代码

ifstream infile("C:\\Users\\yixiao\\Desktop\\9\_1951220\_伊啸\\courseInformation.txt", ios::in);

if (!infile.is\_open()) {

cout << "File opening failed" << endl;

return;

}

string line;

while (getline(infile, line)) { //一行一行读文件

string courseNumber, courseName, advancedCourse1, advancedCourse2;

int courseHours, courseSemester;

istringstream iss(line);

iss >> courseNumber >> courseName >> courseHours >> courseSemester >> advancedCourse1 >> advancedCourse2;

ScheduledTime scheduledtime(courseSemester);

Course course(scheduledtime, courseNumber, courseName, advancedCourse1, advancedCourse2, courseHours);//构造课程节点

if (insertVertex(course)) {//插入课程节点

if (advancedCourse1 != "") {

int position1 = getVertexPos(advancedCourse1);

insertEdge(position1, numVertices - 1); //插入边

NodeTable[numVertices - 1].count++;

NodeTable[numVertices - 1].counter++;

}

if (advancedCourse2 != "") {

int position2 = getVertexPos(advancedCourse2);

insertEdge(position2, numVertices - 1);

NodeTable[numVertices - 1].count++;

NodeTable[numVertices - 1].counter++;

}

}

}

infile.close();

return;

# 4 测试

## 4.1 功能测试

                                          第1学期  
            周一                周二                周三                周四                周五  
 1-2节课  : 英语                程序设计基础                                                                  
3-5节课   : 大学语文            计算机文化          英语                线性代数                                
6-7节课   :                                                                                                        
8-10节课  : 数据通信            线性代数            数据通信            程序设计基础                          
  
                                          第2学期  
            周一                周二                周三                周四                周五  
 1-2节课  : 英语                                    汇编语言                                                    
3-5节课   : 高等数学            计算机组成原理      英语                计算机组成原理      汇编语言            
6-7节课   :                                                                                                       
8-10节课  :                     离散数学          高等数学              离散数学                                
  
                                          第3学期  
            周一                周二                周三                周四                周五  
 1-2节课  : 英语                普通物理                                普通物理                                
3-5节课   : 数值分析            计算机系统结构      英语                计算机系统结构                       
6-7节课   :                     数据结构算法                            数据结构算法                          
8-10节课  :                                         数值分析                                                    
  
                                          第4学期  
            周一                周二                周三                周四                周五  
 1-2节课  : 英语                操作系统原理        微机原理            操作系统原理                          
3-5节课   : 单片机应用          数据库原理          英语                计算机网络                             
6-7节课   : 微机原理            计算机网络          编译原理            数据库原理                             
8-10节课  : 编译原理            面向对象程序设计    Java                VC++                                        
  
                                          第5学期  
            周一                周二                周三                周四                周五  
 1-2节课  : 英语                JSP程序设计         C#.net              VB.net                                      
3-5节课   : C#.net              VB.net              英语                JSP程序设计                             
6-7节课   : PowerBuilder        Delphi              ASP程序设计         C++Builder                                  
8-10节课  : ASP程序设计         C++Builder          PowerBuilder        Delphi                                      
  
                                          第6学期  
            周一                周二                周三                周四                周五  
 1-2节课  : 英语                                    算法设计                                                    
3-5节课   :                                         英语                                                          
6-7节课   : 算法设计                                                                                            
8-10节课  :                                                                                                       
  
                                          第7学期  
            周一                周二                周三                周四                周五  
 1-2节课  : 英语                                                                                                  
3-5节课   :                                         英语                                                          
6-7节课   :                                                                                                       
8-10节课  :                                                                                                       
  
                                          第8学期  
            周一                周二                周三                周四                周五  
 1-2节课  : 英语                                                                                                  
3-5节课   :                                         英语                                                          
6-7节课   :                                                                                                       
8-10节课  :

以上为排课软件输出的结果