项目说明文档

数据结构课程设计

——排课软件

作 者 姓 名： 李子涵

学 号： 1851892

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](#_Toc9155)

[1.1 背景分析 1](#_Toc14323)

[1.2 功能分析 1](#_Toc26010)

[2 设计 2](#_Toc10161)

[2.1 数据结构设计 2](#_Toc9016)

[2.2 类结构设计 2](#_Toc17007)

[2.3 成员设计 2](#_Toc27094)

[2.4 系统设计 2](#_Toc29996)

[3 实现 3](#_Toc23675)

[3.1 排学期的实现 3](#_Toc8398)

[3.1.1 排学期流程图 3](#_Toc25218)

[3.1.2 排学期核心代码 3](#_Toc14605)

[3.2 星期排课功能的实现 5](#_Toc19306)

[3.2.1 星期排课功能流程图 5](#_Toc22404)

[3.2.2 星期排课核心代码 5](#_Toc6673)

[3.3 2课时排课的实现 7](#_Toc19591)

[3.3.1 2课时排课流程图 7](#_Toc31426)

[3.3.2 2课时排课核心代码 8](#_Toc17261)

[3.4 3节排课的实现 9](#_Toc17090)

[3.4.1 3节排课流程图 9](#_Toc32372)

[3.4.2 3节排课核心代码 10](#_Toc3392)

[3.5 总体系统的实现 13](#_Toc3682)

[3.5.1 总体系统流程图 13](#_Toc26842)

[3.5.2 总体系统核心代码 13](#_Toc19599)

[3.5.3 总体系统截屏示例 16](#_Toc25095)

[4 测试 17](#_Toc21001)

[4.1 功能测试 17](#_Toc23088)

[4.1.1 正常输入输出测试 17](#_Toc15587)

[4.2 出错测试 21](#_Toc22594)

[4.2.1 输入文件名错误 21](#_Toc23793)

[4.2.2 学期课程数与总数不匹配 21](#_Toc9967)

# 1 分析

## 1.1 背景分析

现今大学院系众多、专业细分程度高，每门专业的专业课数量多且差异不小。这种背景下，为每个专业每个学期分配课程并且排出相应的课表是一个不小的工程。若采用人工方式来进行排课，固然可行，但太过费时费力。排课要求的条件并不复杂，人工排课虽然也能达到目的，但大量重复性的工作如果使用计算机完成会更方便快捷。

因此，设计一个大学专业排课软件智能地将每个专业的课程分到各个学期、再排好星期课表，可以省去很多人力和时间，简化高校排课问题。排课软件的设计很具有现实意义。

## 1.2 功能分析

一个排课软件，首先应该有的功能是读取各个学期的课程数目以及所有学科的信息，并进行相应的储存与梳理。排课软件的核心功能应该是进行符合要求的排课，比如要求学期、先修科目以及每星期课表的具体要求。除此以外，还应有对输入错误情况的排查和反馈，以及成功排课后的输出。

综上所述，一个排课软件至少要有的功能是输入课程信息，进行排课运算、错误排查以及输出的功能。

# 2 设计

## 2.1 数据结构设计

根据如上功能分析，核心算法要求的排课需要考虑先决条件，即每个课程分配到某一学期之前必须确保在此之前需要的先修课程都已完成。最符合这一要求的结构即是图，可以用活动网络来存储课程信息。由于所有课程都是一学期完成，不需要考虑完成时间，因此只需要不加权图。

## 2.2 类结构设计

图通常可以使用邻接矩阵或邻接表来表示。由于该排课软件需要的活动网络并不十分复杂，使用邻接矩阵会浪费储存空间，因此选择邻接表来储存图。邻接表采用静态链表储存，方便存取。

## 2.3 成员设计

struct lesson

{

string code;

string name;

int hours=0;//studying hours one week

int sem=0;//semister

string pri\_code[2]{"null","null"};

int prior[2]{ -1,-1 };

};

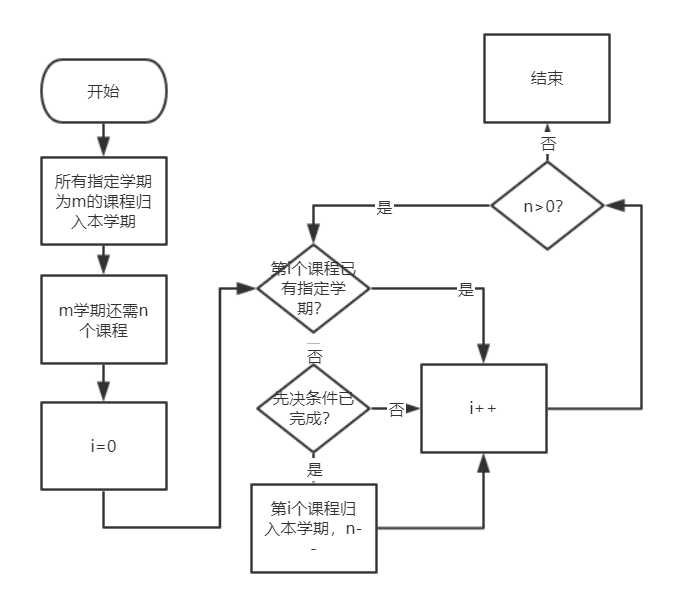
## 2.4 系统设计

首先读取输入文件中的信息，在确认课程数目有效后进行学期分配，然后生成每学期的课表并输出到指定文件。

# 3 实现

## 3.1 排学期的实现

### 3.1.1 排学期流程图



### 3.1.2 排学期核心代码

void arrange\_sem(int s, int n, lesson\* l, int a[])//nunber of semister,quantity of lessons,the array of lessons

{

int count = 0;

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

{

if (l[i].sem == s)

{

a[count] = i;

count++;

}

}

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

{

if (l[i].sem == 0)

{

if ((l[i].prior[0] == -1 || l[l[i].prior[0]].sem != 0) && (l[i].prior[1] == -1 || l[l[i].prior[1]].sem != 0))

{

a[count] = i;

l[i].sem = s;

count++;

}

}

if (count == n)

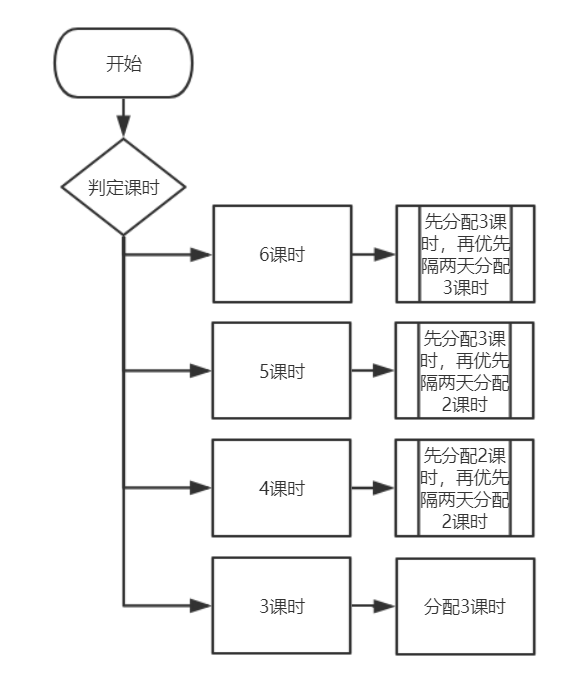
break;

}

}

## 3.2 星期排课功能的实现

### 3.2.1 星期排课功能流程图



### 3.2.2 星期排课核心代码

bool arrange\_week(lesson\* l, int W[10][5], int a[], int n)

{

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

{

switch (l[a[i]].hours)

{

int day, day2;

case 6:

if (arrange\_3(a[i], W, day))

{

if (arrange\_3(a[i], W, day2, (day+2)%5))

break;

}

else return 0;

case 5:

if (arrange\_3(a[i], W, day))

if (arrange\_2(a[i], W, day2, (day + 2) % 5))

break;

else if(arrange\_2(a[i], W, day2))

break;

return 0;

case 4:

if (arrange\_2(a[i], W, day))

{

if (arrange\_2(a[i], W, day2, (day + 2) % 5))

break;

}

else return 0;

case 3:

if (arrange\_3(a[i], W, day))

break;

else return 0;

default: break;

}

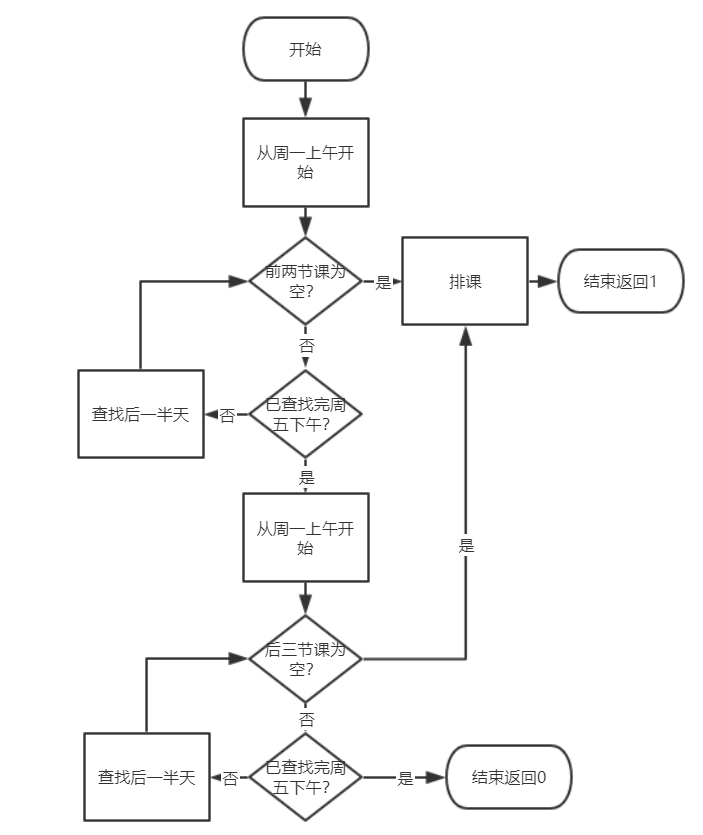
}

return 1;

}

## 3.3 2课时排课的实现

### 3.3.1 2课时排课流程图



### 3.3.2 2课时排课核心代码

bool arrange\_2(int num, int W[10][5], int& d, int day = 0)

{

bool flag = 0;

for (int i=day; i < 5; i++)

{

if (W[0][i] == -1)

{

W[0][i] = num;

W[1][i] = num;

d = i;

flag = 1;

break;

}

else if(W[5][i] == -1)

{

W[5][i] = num;

W[6][i] = num;

d = i;

flag = 1;

break;

}

}

if(flag==0)

for (int i = day; i < 5; i++)

{

if (W[2][i] == -1)

{

W[2][i] = num;

W[3][i] = num;

d = i;

flag = 1;

break;

}

else if (W[7][i] == -1)

{

W[7][i] = num;

W[8][i] = num;

d = i;

flag = 1;

break;

}

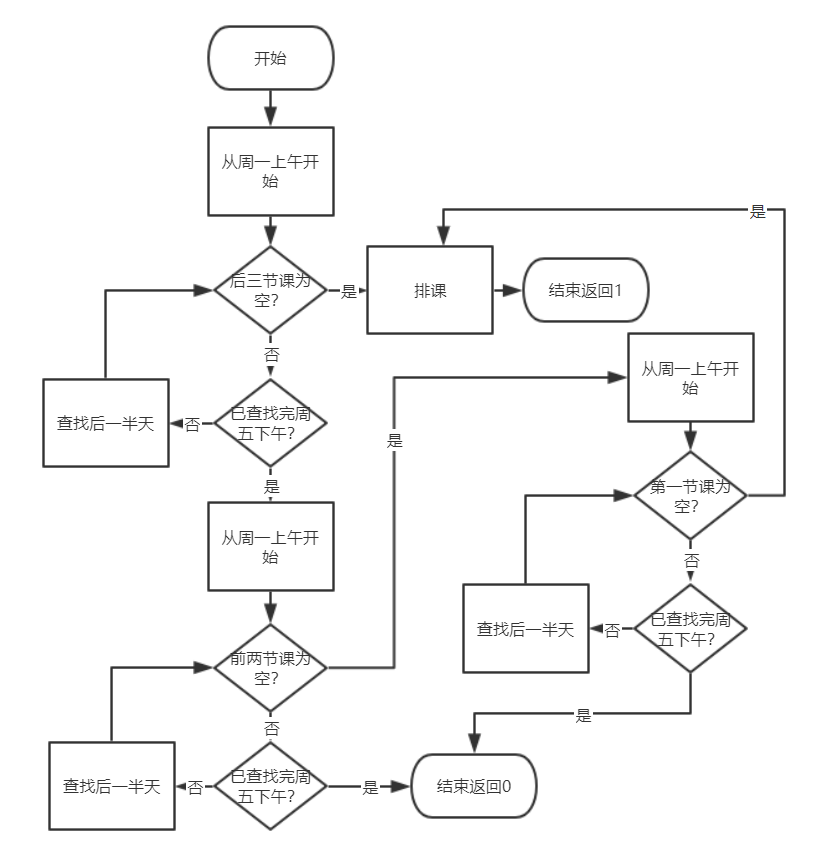
}

return flag;

}

## 3.4 3节排课的实现

### 3.4.1 3节排课流程图



### 3.4.2 3节排课核心代码

bool arrange\_3(int num, int W[10][5], int &d,int day = 0)

{

bool flag = 0;

for (int i = day; i < 5; i++)

{

if (W[2][i] == -1)

{

W[2][i] = num;

W[3][i] = num;

W[4][i] = num;

d = i;

flag = 1;

break;

}

else if (W[7][i] == -1)

{

W[7][i] = num;

W[8][i] = num;

W[9][i] = num;

d = i;

flag = 1;

break;

}

}

if (flag == 0)

{

bool f=0;

for (int i = day; i < 5; i++)

{

if (W[0][i] == -1)

{

W[0][i] = num;

W[1][i] = num;

f = 1;

break;

}

else if (W[5][i] == -1)

{

W[5][i] = num;

W[6][i] = num;

f = 1;

break;

}

}

if(f==1)

for (int i = day; i < 5; i++)

{

if (W[0][i] == -1)

{

W[0][i] = num;

d = i;

flag = 1;

break;

}

else if (W[5][i] == -1)

{

W[5][i] = num;

d = i;

flag = 1;

break;

}

}

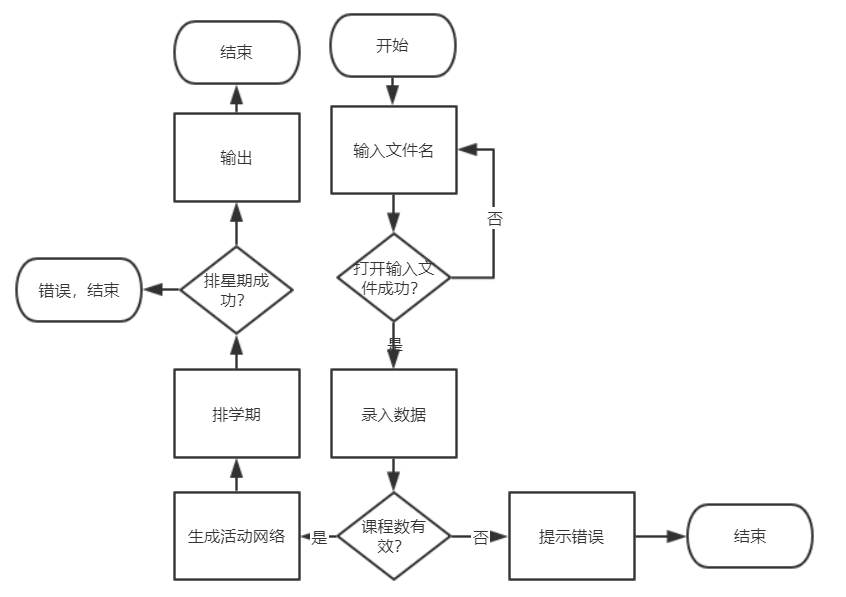
}

return flag;

}

## 3.5 总体系统的实现

### 3.5.1 总体系统流程图



### 3.5.2 总体系统核心代码

int main()

{

ifstream infile;

string inf;

cin >> inf;

infile.open(inf);

while (!infile.is\_open())

{

cout << "wrong file name!input again!";

cin >> inf;

infile.open(inf);

}

int sem[8];

N = 0;

for (int i = 0; i < 8; i++)

{

infile >> sem[i];

N += sem[i];

}

lesson\* L = new lesson[N];

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

{

if (infile.eof())

{

cout << "总课程数与各学期课程和不一致！";

return 0;

}

infile>>L[i].code;

infile >> L[i].name;

infile >> L[i].hours;

infile >> L[i].sem;

char ch;

for(int j=0;infile.get(ch), ch != '\n';j++)

{

infile.putback(ch);

infile>>L[i].pri\_code[j];

}

}

if (!infile.eof())

{

cout << "总课程数与各学期课程和不一致！";

return 0;

}

infile.close();

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

{

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

{

for (int k = 0; k < N; k++)

{

if (L[i].pri\_code[j] == "null")

break;

if (L[k].code == L[i].pri\_code[j])

{

L[i].prior[j] = k;

break;

}

}

}

}

ofstream outfile;

string outf;

cin >> outf;

outfile.open(outf);

for (int i = 1; i < 9; i++)

{

int \*a=new int[sem[i - 1]];

arrange\_sem(i, sem[i - 1], L, a);

int week[10][5];

for (int j = 0; j < 10; j++)

for (int k = 0; k < 5; k++)

week[j][k] = -1;

if (!arrange\_week(L, week, a, sem[i - 1]))

{

cout << "排课失败！";

return 0;

}

outfile << "第" << i << "学期" << endl;

print(outfile, week, L);

outfile << endl;

delete[] a;

}

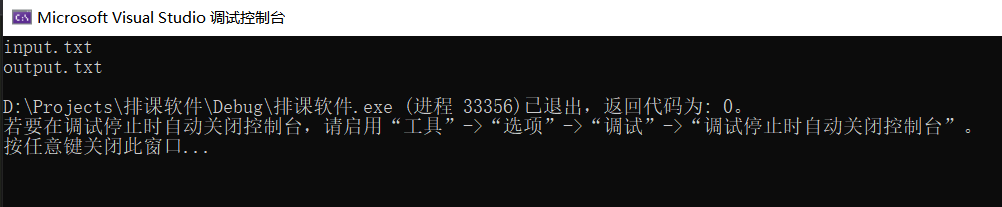
outfile.close();

delete[] L;

return 0;

}

### 3.5.3 总体系统截屏示例



# 4 测试

## 4.1 功能测试

### 4.1.1 正常输入输出测试

**测试用例**：

5 5 5 5 5 5 5 3

c01 程序设计基础 5 0

c02 离散数学 6 0 c01

c03 数据结构算法 4 0 c01 c02

c04 汇编语言 5 0 c01

c05 算法设计 4 0 c03 c04

c06 计算机组成原理 6 0

c07 微机原理 4 0 c03

c08 单片机应用 3 0 c03

c09 编译原理 5 0 c03

c10 操作系统原理 4 0 c03

c11 数据库原理 5 0 c03

c12 高等数学 6 0

c13 线性代数 6 0

c14 数值分析 6 0 c12

c15 普通物理 4 0 c12

c16 计算机文化 3 0

c17 计算机系统结构 6 0 c06

c18 计算机网络 5 0 c03

c19 数据通信 6 0

c20 面向对象程序设计3 0 c01 c03

c21 Java 3 0 c01 c03

c22 C# .net 5 0 c01 c03

c23 PowerBuilder 5 0 c01 c03

c24 VC++ 3 0 c01 c03

c25 ASP程序设计 5 0 c01 c03

c26 JSP程序设计 5 0 c01 c03

c27 VB.net 5 0 c01 c03

c28 Delphi 5 0 c01 c03

c29 C++ Builder 5 0 c01 c03

c30 英语 5 1

c31 英语 5 2

c32 英语 5 3

c33 英语 5 4

c34 英语 5 5

c35 英语 5 6

c36 英语 5 7

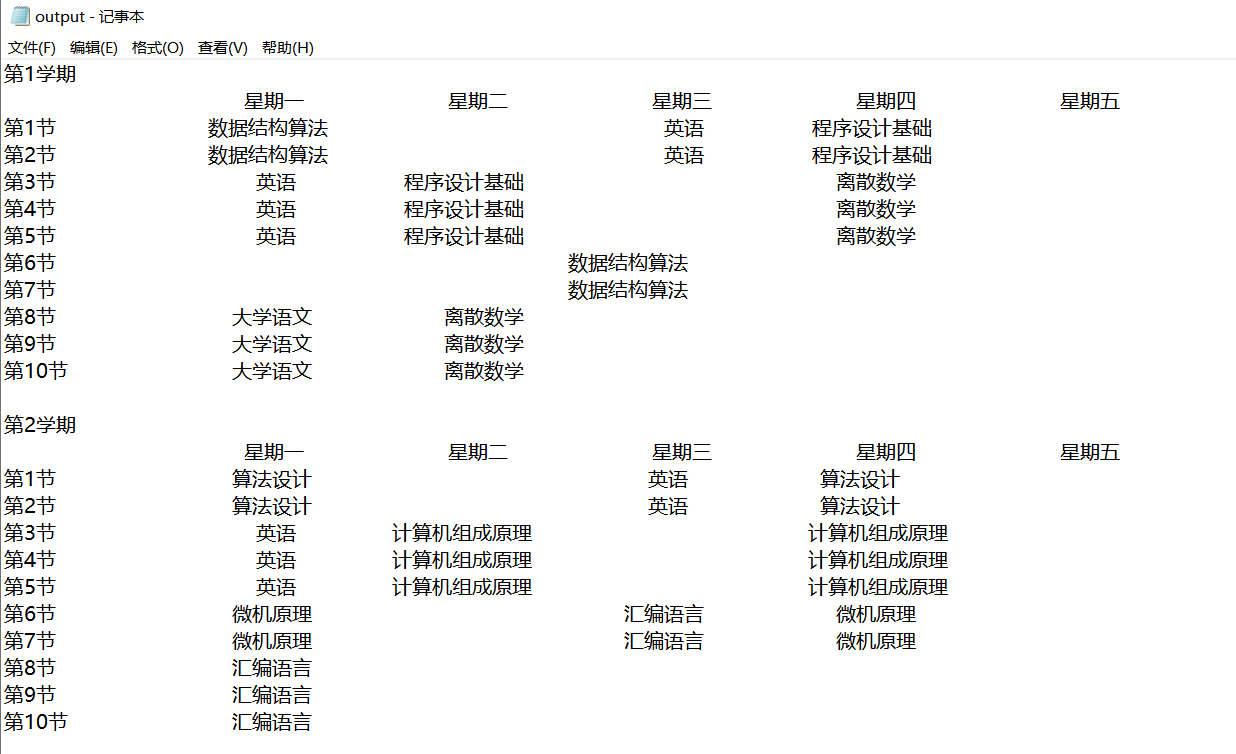
c37 英语 5 8

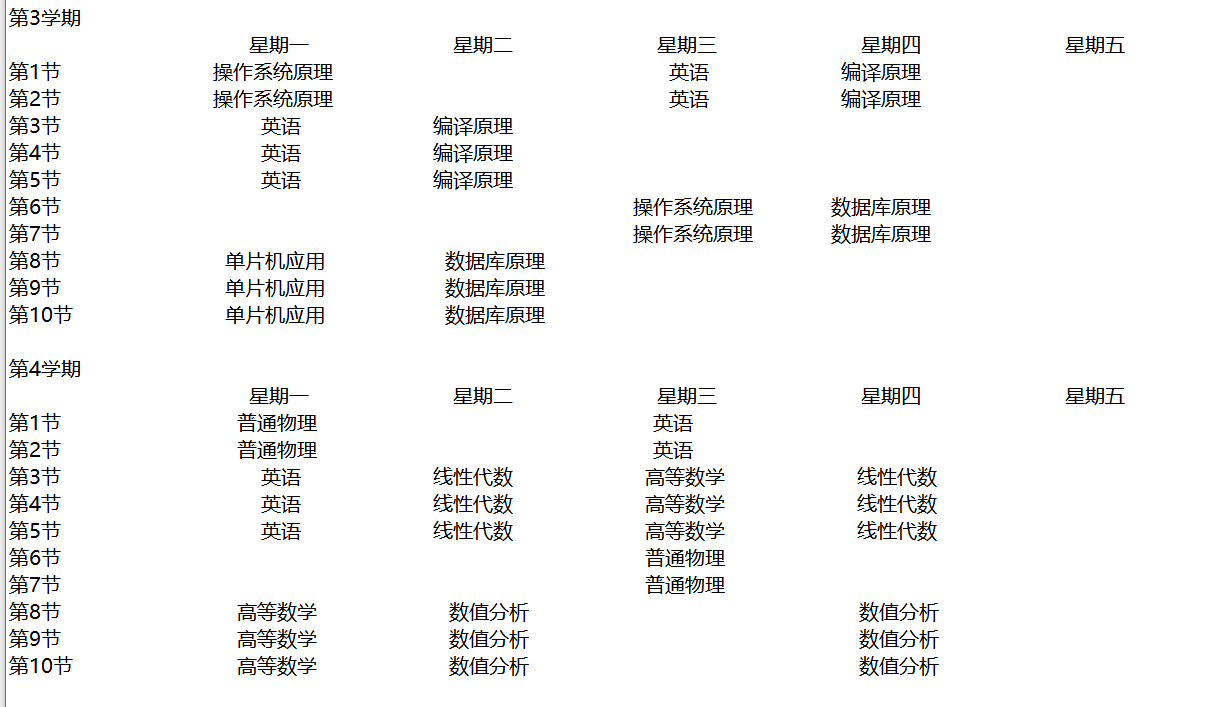
c38 大学语文 3 1

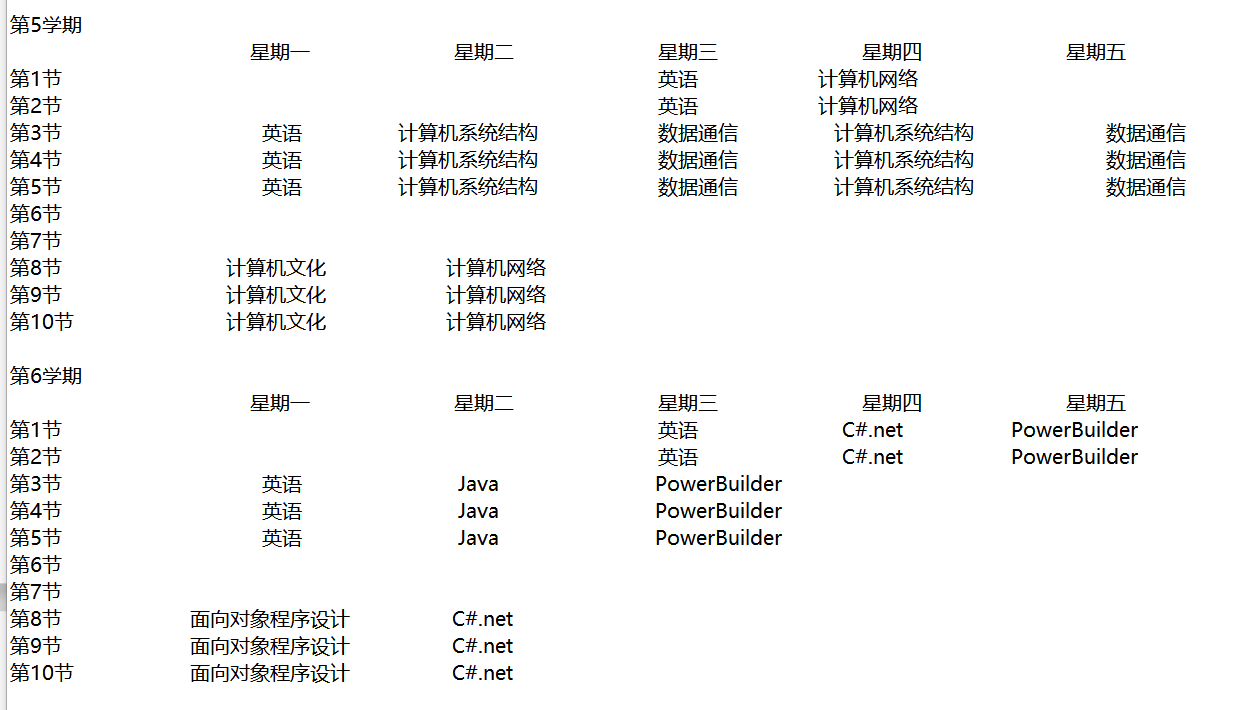
**预期结果**：

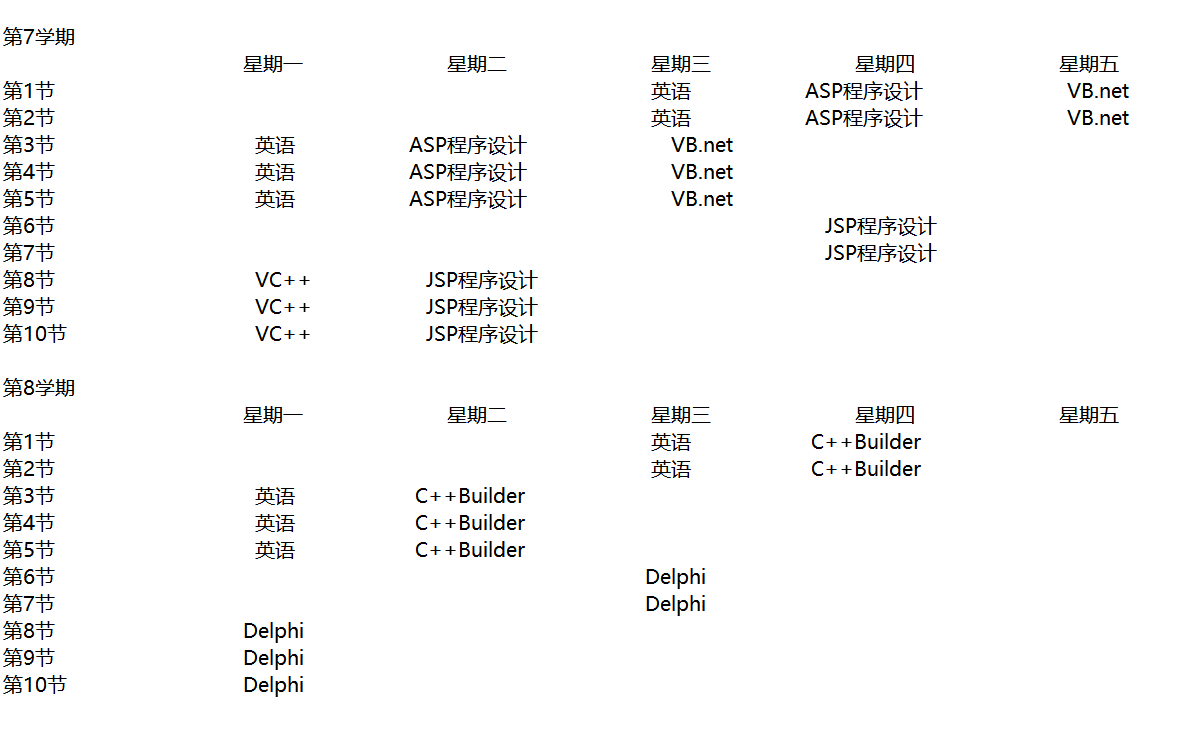
输出课表

**实验结果**









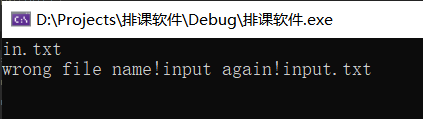
## 4.2 出错测试

### 4.2.1 输入文件名错误

**测试用例：**输入错误文件名

**预期结果：**程序给出提示信息，程序正常运行不崩溃。

**实验结果：**



### 4.2.2 学期课程数与总数不匹配

**测试用例：**

5 5 5 5 5 5 5 3

c01 程序设计基础 5 0

c02 离散数学 6 0 c01

c03 数据结构算法 4 0 c01 c02

c04 汇编语言 5 0 c01

c05 算法设计 4 0 c03 c04

c06 计算机组成原理 6 0

c07 微机原理 4 0 c03

c08 单片机应用 3 0 c03

c09 编译原理 5 0 c03

c10 操作系统原理 4 0 c03

c11 数据库原理 5 0 c03

c12 高等数学 6 0

c13 线性代数 6 0

c14 数值分析 6 0 c12

c15 普通物理 4 0 c12

c16 计算机文化 3 0

c17 计算机系统结构 6 0 c06

c18 计算机网络 5 0 c03

c19 数据通信 6 0

c20 面向对象程序设计 3 0 c01 c03

c21 Java 3 0 c01 c03

c22 C#.net 5 0 c01 c03

c23 PowerBuilder 5 0 c01 c03

c24 VC++ 3 0 c01 c03

c25 ASP程序设计 5 0 c01 c03

c26 JSP程序设计 5 0 c01 c03

c27 VB.net 5 0 c01 c03

c28 Delphi 5 0 c01 c03

c29 C++Builder 5 0 c01 c03

c30 英语 5 1

c31 英语 5 2

c32 英语 5 3

c33 英语 5 4

c34 英语 5 5

c35 英语 5 6

c36 英语 5 7

c37 英语 5 8

**预期结果：**给出错误信息。

**实验结果：**

