项目说明文档

数据结构课程设计

——排课软件

同济大学

Tongji University

目 录

[1 分析 1](#_Toc60329604)

[1.1 背景分析 1](#_Toc60329605)

[1.2 功能分析 1](#_Toc60329606)

[2 设计 3](#_Toc60329607)

[2.1 数据结构设计 3](#_Toc60329608)

[2.2 类结构设计 3](#_Toc60329609)

[2.3 成员与操作设计 3](#_Toc60329610)

[3 实现 5](#_Toc60329611)

[3.1 构造函数的实现 5](#_Toc60329612)

[3.1.1 构造函数流程图 5](#_Toc60329613)

[3.1.2 构造函数核心代码 6](#_Toc60329614)

[3.2 排课表的实现 7](#_Toc60329615)

[3.2.1 排课表流程图 7](#_Toc60329616)

[3.2.2 排课表核心代码 9](#_Toc60329617)

[3.3 单学期排课功能的实现 9](#_Toc60329618)

[3.3.1 单学期排课功能思想 9](#_Toc60329619)

[3.3.2 单学期排课功能核心代码 9](#_Toc60329620)

[3.4 判断排课是否合理功能的实现 10](#_Toc60329621)

[3.4.1 判断选课是否合理思想 10](#_Toc60329622)

[3.4.2 判断选课是否合理核心代码 10](#_Toc60329623)

[4 使用说明以及测试 12](#_Toc60329624)

[4.1 使用说明 12](#_Toc60329625)

[4.2 程序测试 12](#_Toc60329626)

[4.2.1 测试一 12](#_Toc60329627)

[4.2.2 测试二 14](#_Toc60329628)

[4.2.3 测试三 15](#_Toc60329629)

[4.3 出错测试 17](#_Toc60329630)

[4.3.1 输入个数错误 17](#_Toc60329631)

[4.3.2 数据和不为38 18](#_Toc60329632)

# 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

项目功能要求：

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）；

课程信息：

课程编号 课程名称 学时数 指定开课学期 先修课程

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

本题较为复杂，需要考虑较多种情况。在理想的情况下，应该要能够做到：①、如一门课程有3节课，则优先安排3节课连续上；②、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。考虑到实际情况：如果上述课程较为均匀地分散在8个学期，可以求出最优解。所以，本程序的功能是在学期间课程分布较为均匀时求出最优解，而不考虑将3节课拆开，一门课程安排在连续两天的情况，所以会忽略部分可行解。

# 2 设计

## 2.1 数据结构设计

如上功能分析所述，该课表有先修、后修这样一种关系，所以需要采用拓扑排序的思想，所以使用图结构进行存储。

## 2.2 类结构设计

将每一门课程作为一个节点，其中除了存储课程本身的信息外，还需要存储入度和出度。入度的含义是本课程先修课程的个数，出度的含义是以本课程为先修课程的课程的个数。

## 2.3 成员与操作设计

**边类（ArcNode）**

struct ArcNode

{

int adjvex;

// 存储相邻点;

ArcNode\* nextarc;

//下一条边界点

};

**信息类：**

struct info

{

string num, name, pre[2];

//课程号，课程名，先修课程

int time, assign;

//课时数，分配的学期数

}

**点类（VNode）**

typedef struct VNode

{

info data;

//点中存储的信息

int inde, outde;

//入度和出度

ArcNode\* firstarc;

//与该点相连的边链表

}AdjList[MAX\_VERTEX\_NUM + 2];

//点数组

**图类（ALGraph）：**

**私有成员操作：**

AdjList vertices;

//点集合

int vexnum, arcnum;

//点数和边数

**ADT接口：**

ALGraph();

//构造函数

void init();

//初始化全局变量等

void print();

//打印图的信息（供debug使用）

bool judgeOk(int num,ofstream &);

//判断选课方式是否可行

int arrange\_time(int, int, int);

//安排课程时间

void printTable(ofstream&cout);

//打印课程表（打入文件）

int Select(int , int, int,ofstream&);

//递归选择出合适的选课方式（类似stl中的 next\_permutation，从一个数组里挑选出一定数量的组合）

Status TopoSelect();

//用户调用的函数，排出8个学期的课表

# 3 实现

## 3.1 构造函数的实现

### 3.1.1 构造函数流程图

开始

从文件中读取

算出每一个点的入读和出度

结束

### 3.1.2 构造函数核心代码

ALGraph()

{

vexnum = MAX\_VERTEX\_NUM;

arcnum = 0;

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

vertices[i].firstarc = NULL, vertices[i].inde = vertices[i].outde = 0;

ifstream in("course.txt");

if (!in) { printf("找不到课程文件！\n"); exit(-1); }

//错误判断

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

{

vertices[i].data.readIn(in);

string\* pre = vertices[i].data.pre;

for (int j = 0; j < 2 && pre[j] != NULLString; ++j)

{

...//计算出度和入度

}

}

in.close();

}

## 3.2 排课表的实现

### 3.2.1 排课表流程图

开始

选出入度为0的点并且放入第一轮Select中

结束

打开输出文件

在将之前选出的点的入度扣除后，选出入度为0的点并且放入第i+1轮Select中

8学期全部选完？

y

n

### 3.2.2 排课表核心代码

Status TopoSelect()

{

ofstream out("result.txt");

//打开输出文件

init();

//初始化全局变量

...//选出入度为0的点并且放入第一轮Select中

for (int i = 1; i < 8; ++i)

{

...//在将之前选出的点的入度扣除后，选出入度为0的点并且放入第i+1轮Select中

}

out.close();

//关闭文件

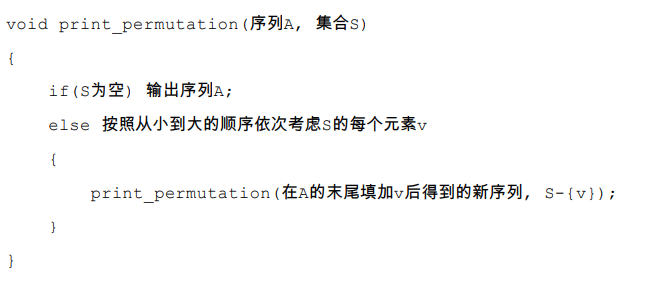
return 1;

}

## 3.3 单学期排课功能的实现

### 3.3.1 单学期排课功能思想

（类似next\_permutation），用于求出全排列



### 3.3.2 单学期排课功能核心代码

int Select(int n, int Num, int t,ofstream&out)

{

int num = semester\_can\_choose[t];

if (n==num)

{

if (judgeOk(semester[t],out)) { return 1;}

//找到一种情况，直接返回

return 0;

}

//递归求全排列

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

{

if (permutation\_used[i])continue;

permutation\_used[i] = 1;

SELECT[n++] = Arr[i];

if (Select(n, Num, t,out))return 1;

--n;

permutation\_used[i] = 0;

}

return 0;

}

## 3.4 判断排课是否合理功能的实现

### 3.4.1 判断选课是否合理思想

判断是否满足两个条件，一， 3＝3，4＝2+2，5＝2+3，6＝2\*3＝3\*2，总共有10个2，10个3，计算3和2的个数是否能够满足3、4、5、6；二，通过模拟判断能否满足时间间隔。将3拆成3，5拆成2+3，6拆成3+3，从星期一往后遍历，如果找到空余位置，则直接填入，由于在之前的空余位置已经填入，后一项的寻找只需要从后数一天开始即可。

### 3.4.2 判断选课是否合理核心代码

bool judgeOk(int num,ofstream &out)

{

int ok\_2 = 10, ok\_3 = 10;

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

{

int t = vertices[SELECT[i].no].data.time;

if (t == 3)--ok\_3;

else if (t == 4)ok\_2 -= 2;

else if (t == 5)--ok\_2, --ok\_3;

else if (t == 6)ok\_3 -= 2;

}

if (ok\_2 < 0 || ok\_3 < 0)return 0;

for (int i = 0; i < 4; ++i)

for (int j = 0; j < 5; ++j)course\_table[i][j] = -1;

bool arrange[MAX\_VERTEX\_NUM] = { 0 };

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

{

int t = vertices[SELECT[i].no].data.time;

if (t == 3)continue;

int n = SELECT[i].no;

arrange[n] = 1;

int day1, day2;

if (t == 4)

{

if ((day1 = arrange\_time(0, n, 2)) == -1)return 0;

if ((day2 = arrange\_time(day1 + 2, n, 2)) == -1)return 0;

}

if (t == 5)

{

if ((day1 = arrange\_time(0, n, 2)) == -1)return 0;

if ((day2 = arrange\_time(day1 + 2, n, 3)) == -1)return 0;

}

if (t == 6)

{

if ((day1 = arrange\_time(0, n, 3)) == -1)return 0;

if ((day2 = arrange\_time(day1 + 2, n, 3)) == -1)return 0;

}

}

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

{

int n = SELECT[i].no;

if (arrange[n])continue;

if (arrange\_time(0, n, 3) == -1)return 0;

}

printTable(out);

return 1;

}

# 4 使用说明以及测试

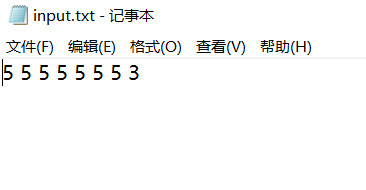
## 使用说明

请在input.txt中输入每学期需要学习的课程数，请保证输入8个数，并且其和为38（否则会给出错误提示）。不需要动course.txt文件，否则程序无法正常运行！

## 4.2 程序测试

### 4.2.1 测试一

**测试用例：**

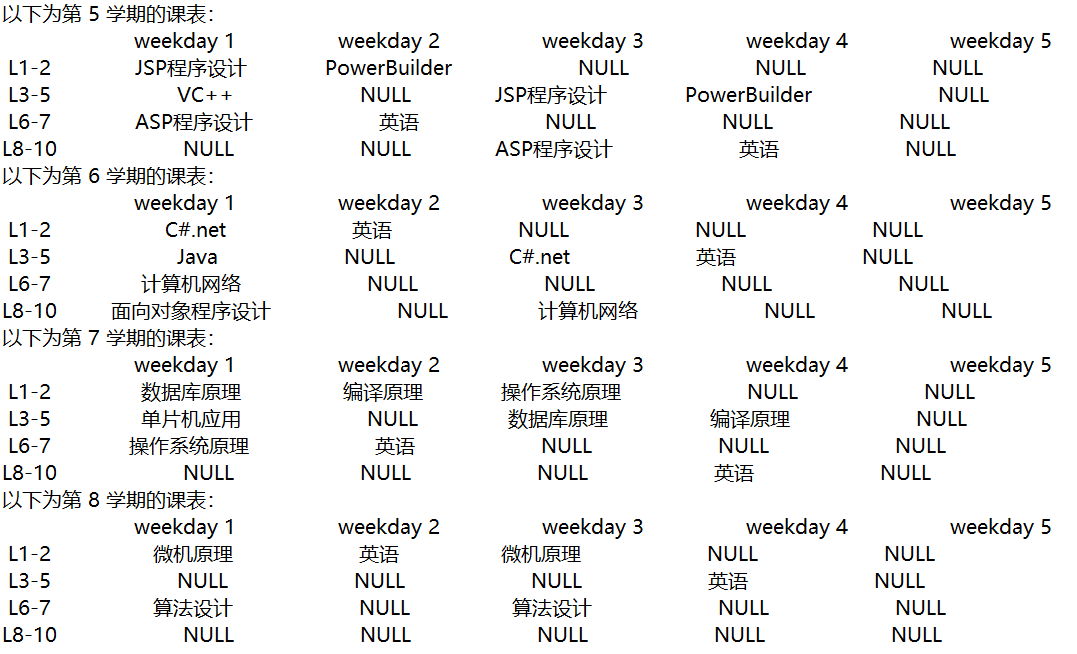


**预期结果：**

给出排课结果

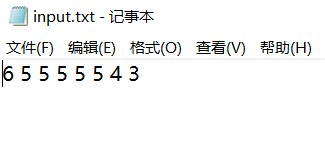
**实验结果：**





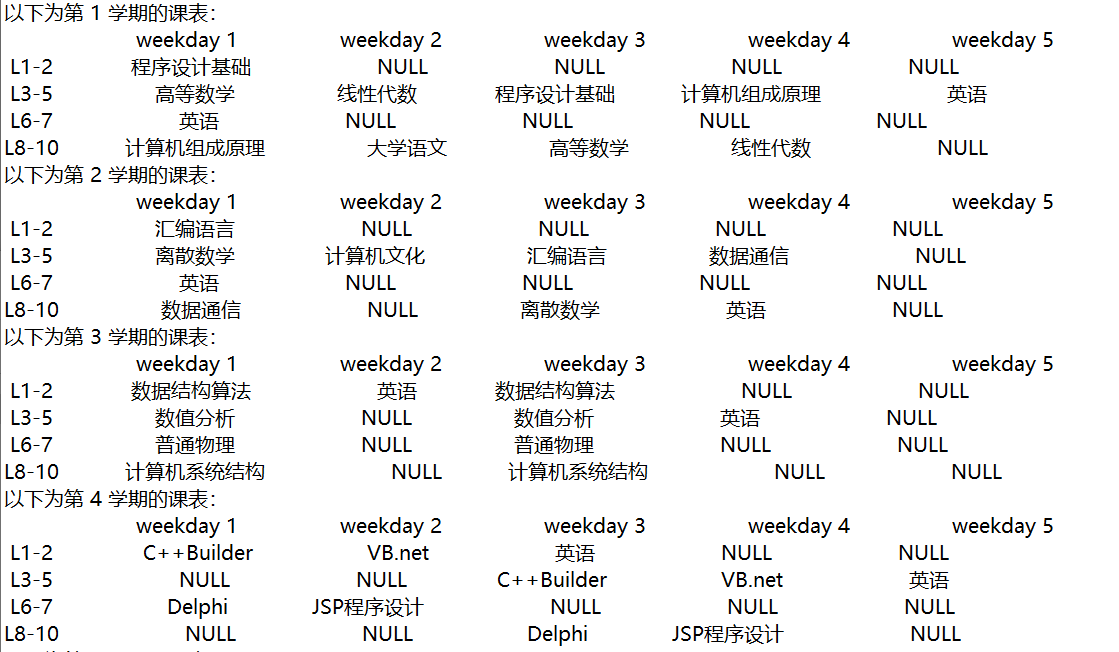
### 4.2.2 测试二

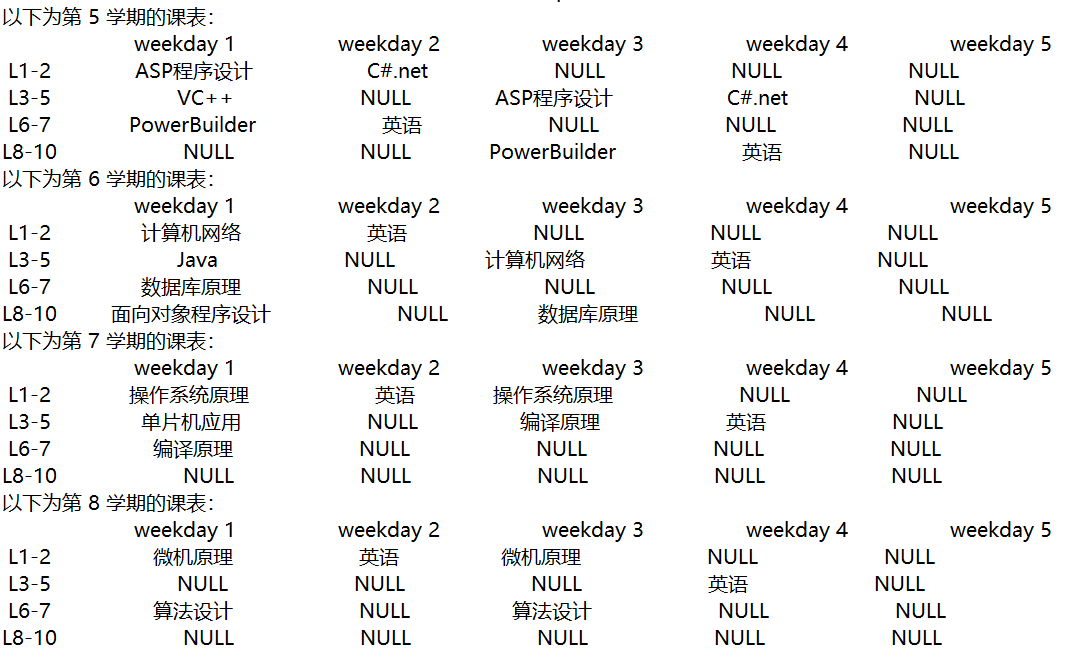
**测试用例：**



**预期结果：**给出排课结果

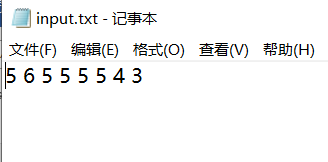
**实验结果：**





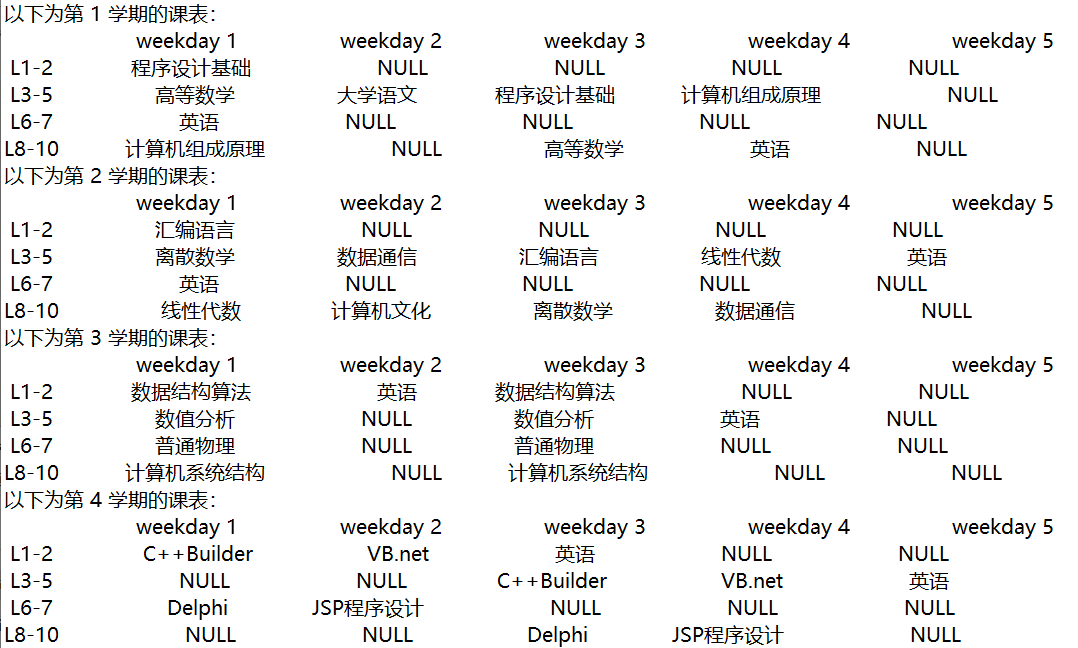
### 4.2.3 测试三

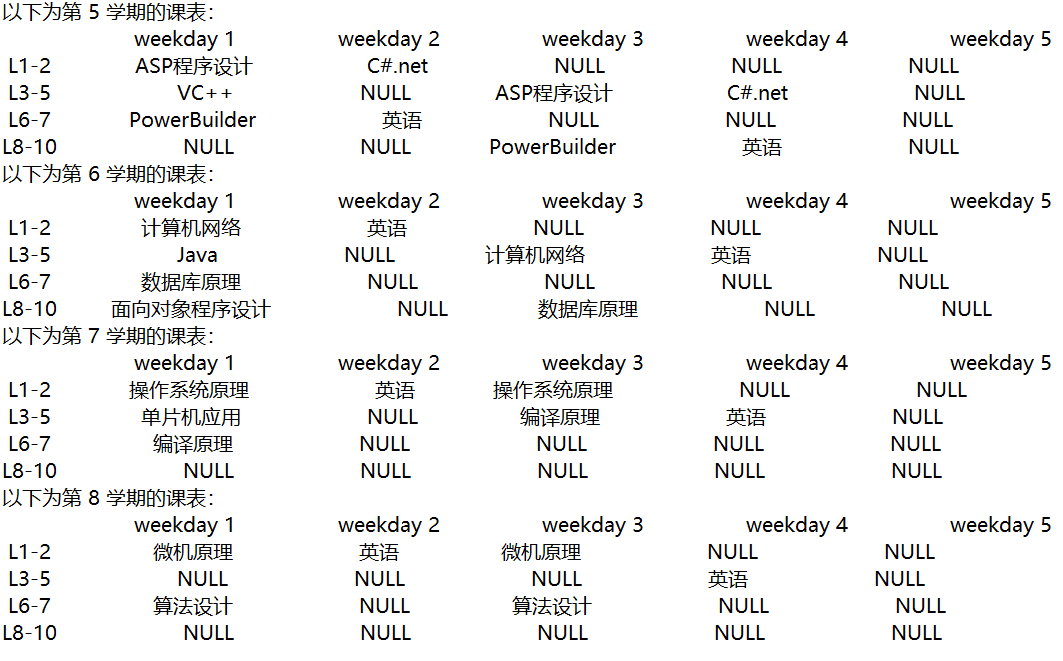
**测试用例：**



**预期结果：**程序正常运行，不崩溃。

**实验结果：**

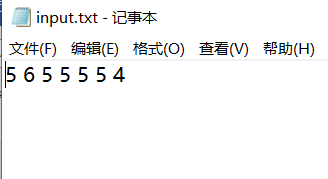




## 4.3 出错测试

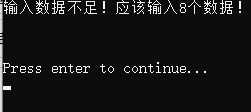
### 4.3.1 输入个数错误

**测试用例：**



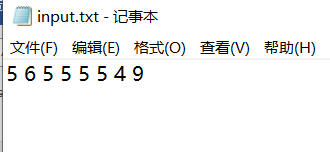
**预期结果：**程序给出提示信息，程序正常运行不崩溃。

**实验结果：**

****

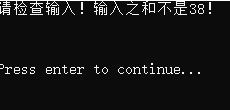
### 4.3.2 数据和不为38

**测试用例：**



**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：**

****