

电子信息与通信学院

实 验 报 告

|  |  |
| --- | --- |
| 实验名称 | 课程综合练习 |
| 课程名称 | 计算机基础  与程序设计(C) |

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | 詹金馨 | 学号 | U202413717 |

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 2025.1.16 | 地点 | 华中科技大学 |

|  |  |  |  |
| --- | --- | --- | --- |
| 成绩 |  | 教师 | 刘威 |

# 实验目的

完成日历系列代码。

# 实验环境

操作系统：Windows 10

编程工具：CodeBlocks 16.01

**写在开始之前：求求老师轻点扣分，我真的尽力写了，但是我们之前的代码任务是分工写的，我选择的是其他课件里的代码，日历系列我真的是只写了大家都要写的前三个。我白天要去电工基地上课，没什么时间写代码，但是晚上我都坐在图书馆认真写实验报告！在十八号快要截止的时候我才看到老师说的范围是后面的内容！但是这之前我只做了前面几个代码的！我最后一天努力写出了最后两个代码和报告，剩下的实在来不及了，不是故意不写的。 （我知道写不完是我自己的问题，我真的很抱歉！！！）**

# 实验一（日历）

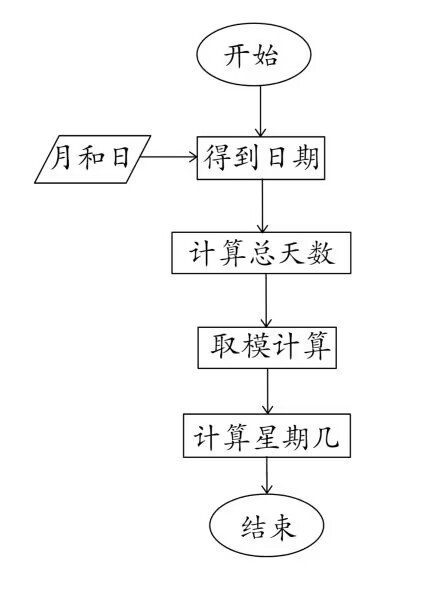
## 实验任务

运用所学到的知识，编写一个计算星期几的程序，计算2024年某月某日是星期几（已知2024年1月1日是星期一）

·设置一个变量计算输入日是全年第几天，通过算术取模运算计算输入日是星期几

·约定一个星期从周一开始，如果是周一则打印1，周二则打印2，……，周日则打印7

## 实验步骤



## 代码测试

### 测试点 （1，1）的测试结果

输入月份为1，日期为1，测试第一天是否为星期一。



输出结果为1，说明在该程序下一月一日确实为星期一，测试成功。

### 测试点（1，8）的测试结果

输入月份为1，日期为8，判断能否正常循环计算星期数。



输出结果为1，说明该程序确实能够正常循环计算星期数，而不是一直无限增加，测试成功。

## 实验结论

代码确实能够正确计算2024年的某一天是星期几。

## 实验总结

问题一：我第一版本的代码中没有给出应有的关于输入的提示，人机交互较差，最终版本加入了输入提示。

问题二：第一版本中将七个日期分了七类来讨论，使得代码冗余复杂，最终版本找到了余数和星期数之间的代数关系，使得代码更加简洁。

# 实验二（月历）

## 实验任务

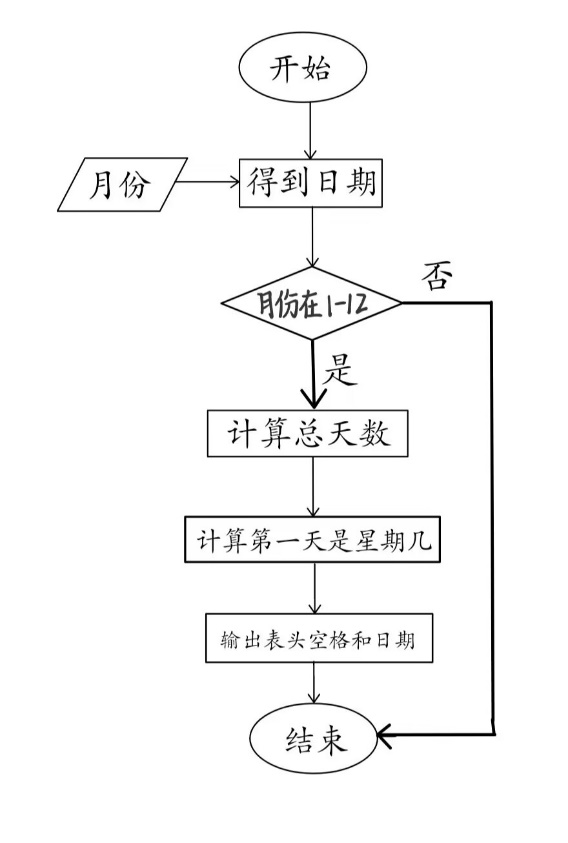
运用所学到的知识，编写打印月历的程序，打印2024年1到12月的某个月的月历（已知2024年1月1日是星期一）

·约定每个星期从周一开始

·约定月历的每列的宽度为10个字符，可以在printf语句中用%10s打印空格、用%10d打印数字来定位

·【提示】设置一个变量存储每个月的总天数；设置一个变量存储第一天是星期几（利用前一个实验计算）；利用循环控制语句打印该月的每一天，如果碰到周日就要换行

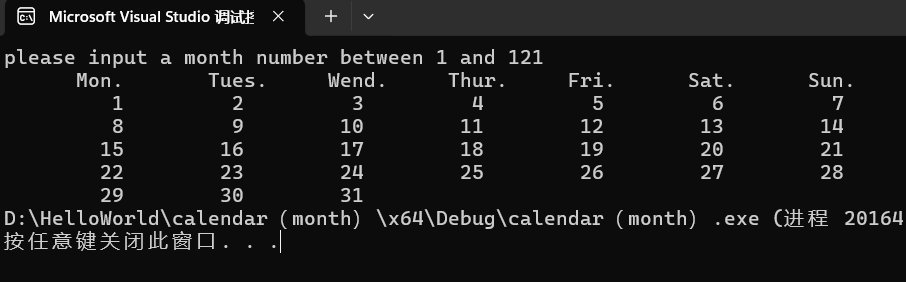
## 实验步骤



## 代码测试

### 样本点1的测试结果

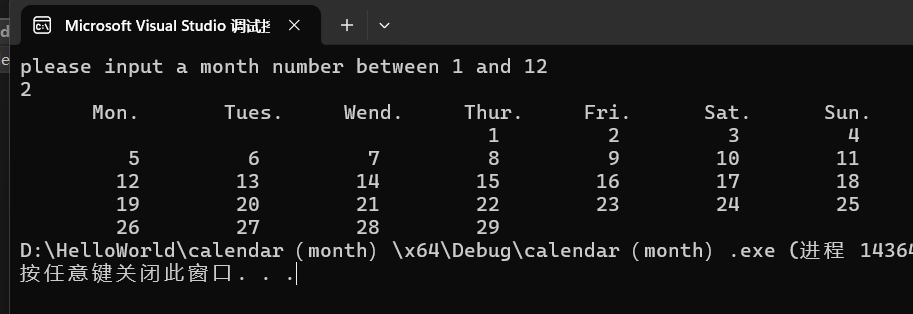
输入月份为1，判断是否能正常输出表头，能否正常换行以及一月一日是否为星期一。



结果表明能完成上述任务，测试通过。

### 样本点2的测试结果

输入月份为1，判断是否能正常输出空格以及二月是否为29天。



结果表明确实能够完成上述任务，测试通过。

## 实验结论

代码确实能够正确打印每个月的月历。

## 实验总结

问题一：我第一次写的版本没有判断月份否在1到12范围内，有数组访问越界的风险，最终的版本加入了判断，防止数组访问越界。

问题二：我第一次写的版本将月份按照第一天的星期数分为了七类进行讨论，实际上可以通过循环来控制输出空格数，不需要分类讨论，最后的版本较为简洁。

# 实验三（周历）

## 实验任务

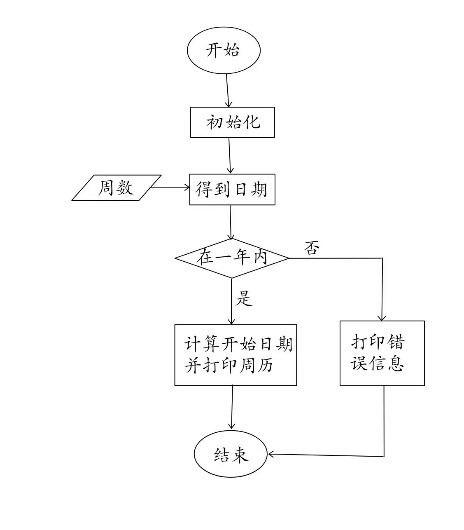
运用所学到的知识，编写一个打印周历的程序，打印2024年某一周的周历（2024年1月1日是星期一）

·约定每个星期从周一开始

·约定周历的每列的宽度为10个字符，可以在printf语句中用%5打印空格、用%2d打印数字来定位

·在第一周和第三十五周中，仅打印2024年的日期。

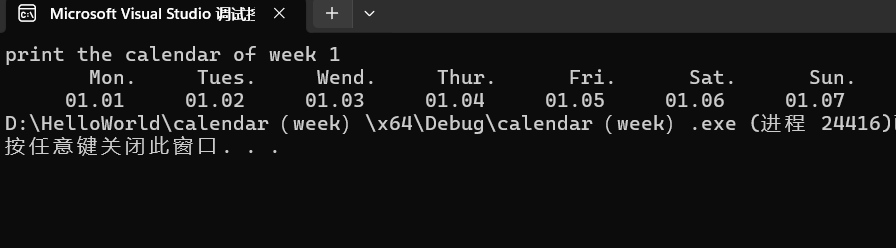
## 实验步骤



## 代码测试

### 样本点1的测试结果

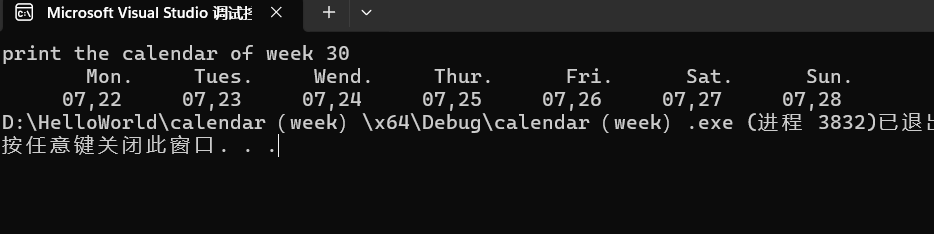
输入周数为1，测试是否能正常打印表头，第一天是否为星期一，是否能够正确定位。



测试结果表明确实能够正常打印表头，且一月一日是星期一，能够正确定位。测试通过。

### 样本点30的测试结果

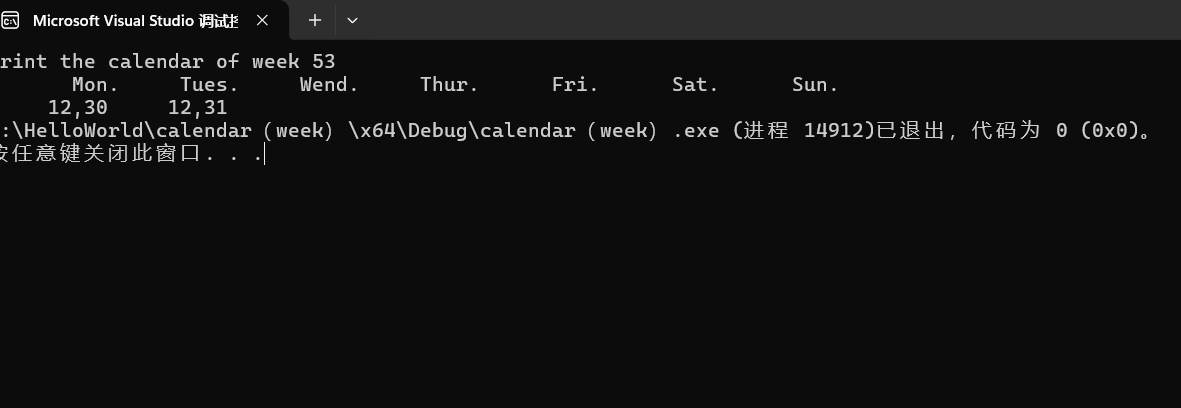
随机输入一个星期数，再次测试能否正确打印表头以及定位。



测试结果表明，确实能够正确完成题目要求，测试通过。

### 样本点53的测试结果

输入最后一周，测试是否只打印2024年的部分。



测试结果表明程序确实只打印了2024年的部分，条件约束没有问题，测试通过。

## 实验结论

该程序能够正确打印2024年各周的周历，且只打印2024年的部分，完成了题目要求。

## 实验总结

该实验代码有简明的注释，有正确的约束条件，且判断了数组查找是否越界，运用for循环代替重复的代码，较成功地完成了实验要求。

# 实验四（春季校历）

## 实验任务

-运用所学到的知识，编写一个打印华中科技大学校历的程序，打印2020年春季学期的校历

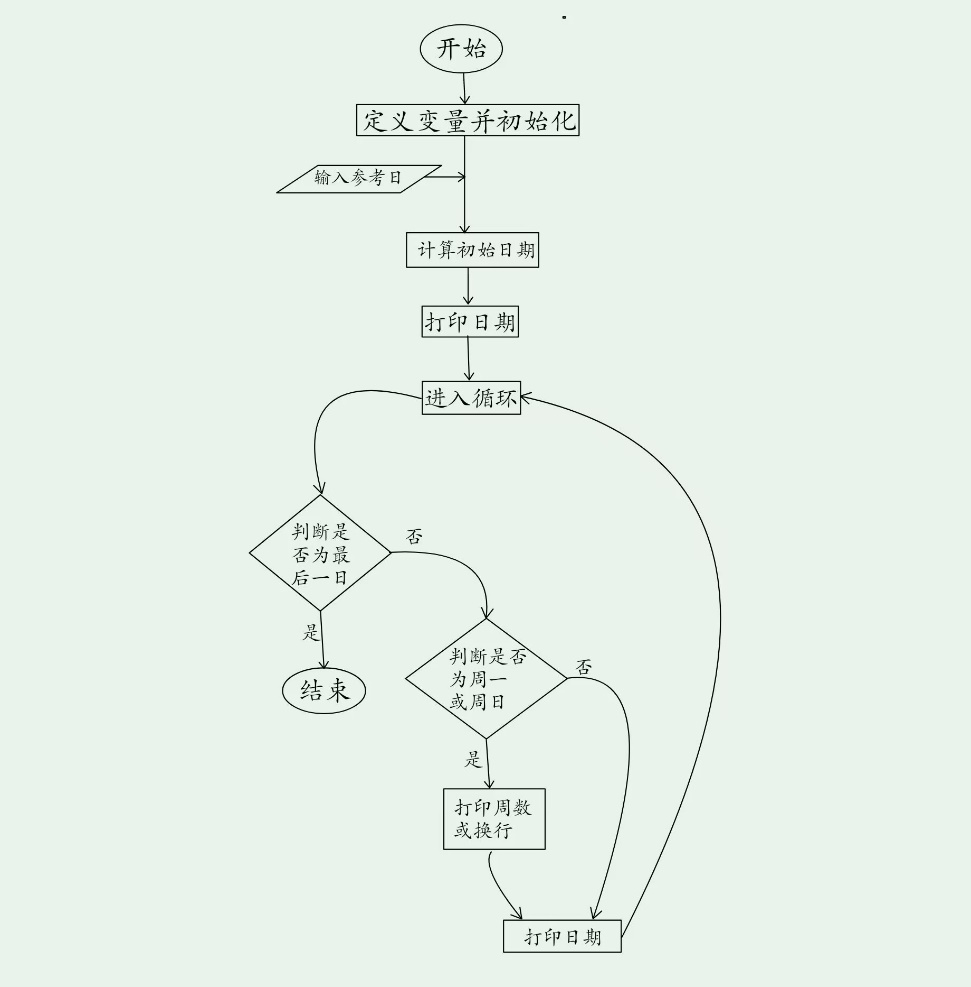
-约定一：春季学期从2月份的某一周开始，用户输入2月份的某一日作为参考日：如果该日就是周一，即从当周开始；如果该日不是周一，即从下周开始。

-约定二：春季学期到第一周周末结束。

-约定校历的每列的宽度为10个字符，可以在printf语句中用%5s打印空格、用%02d打印月或者日的数字

-【提示】编程思路类似周历，以全年第几天为循环变量打印，循环体中判断所打印日期是几月几号；结合周几的信息，打印周一前的周次，打印周日后的换行

## 实验步骤

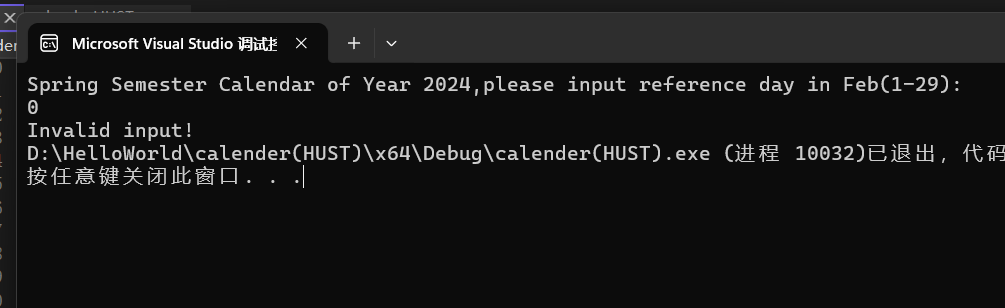


其实在计算初始日期前应该有判断是否越界的步骤，我第一次写的时候没有注意，在修改时我加入了判断的语句，在我的流程图里没有表现出来，特此说明。

## 代码测试

### 样本点0的测试结果

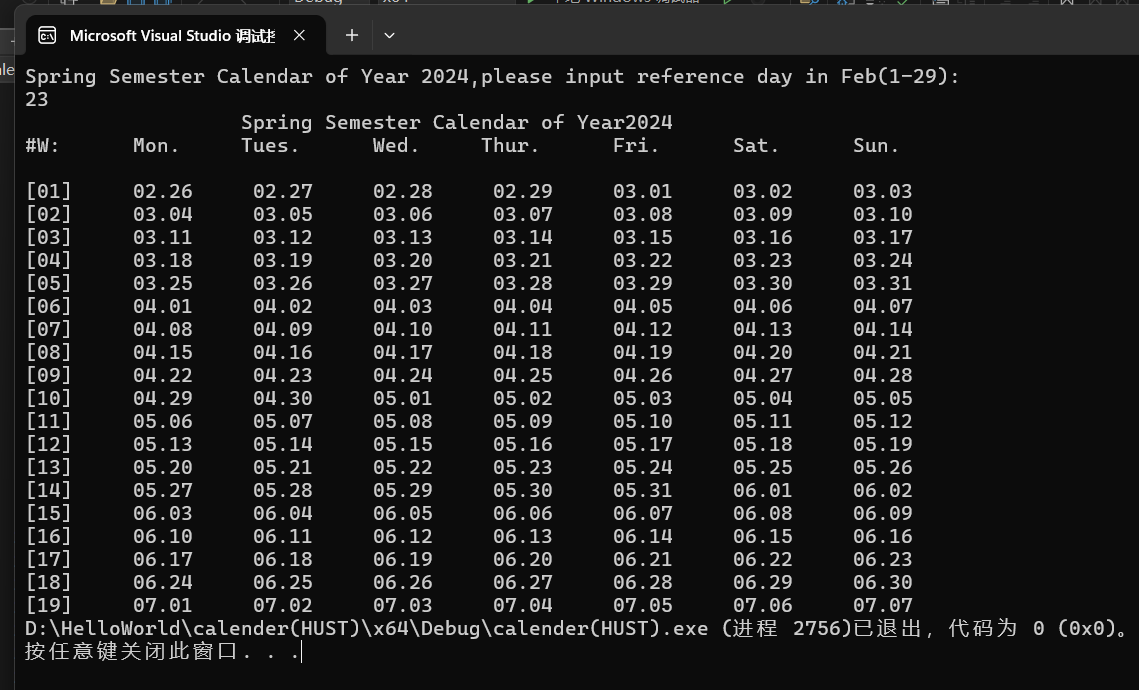
输入参考日为2月0日，测试代码能否正常拦截错误输入并且打印错误信息。



测试结果显示，输入0时确实能正常打印出报错信息并且结束进程，测试通过。

### 样本点23的测试结果

输入参考日为2月23日，测试代码能否正确计算开始日期并在正确的日期结束，测试代码能否正确打印星期数和换行。



实验结果表明程序确实能够完成上述要求，测试通过。

## 实验结论

该程序能够根据所给的参考日计算开始日期并且打印华中科技大学春季校历，测试通过。

## 实验总结

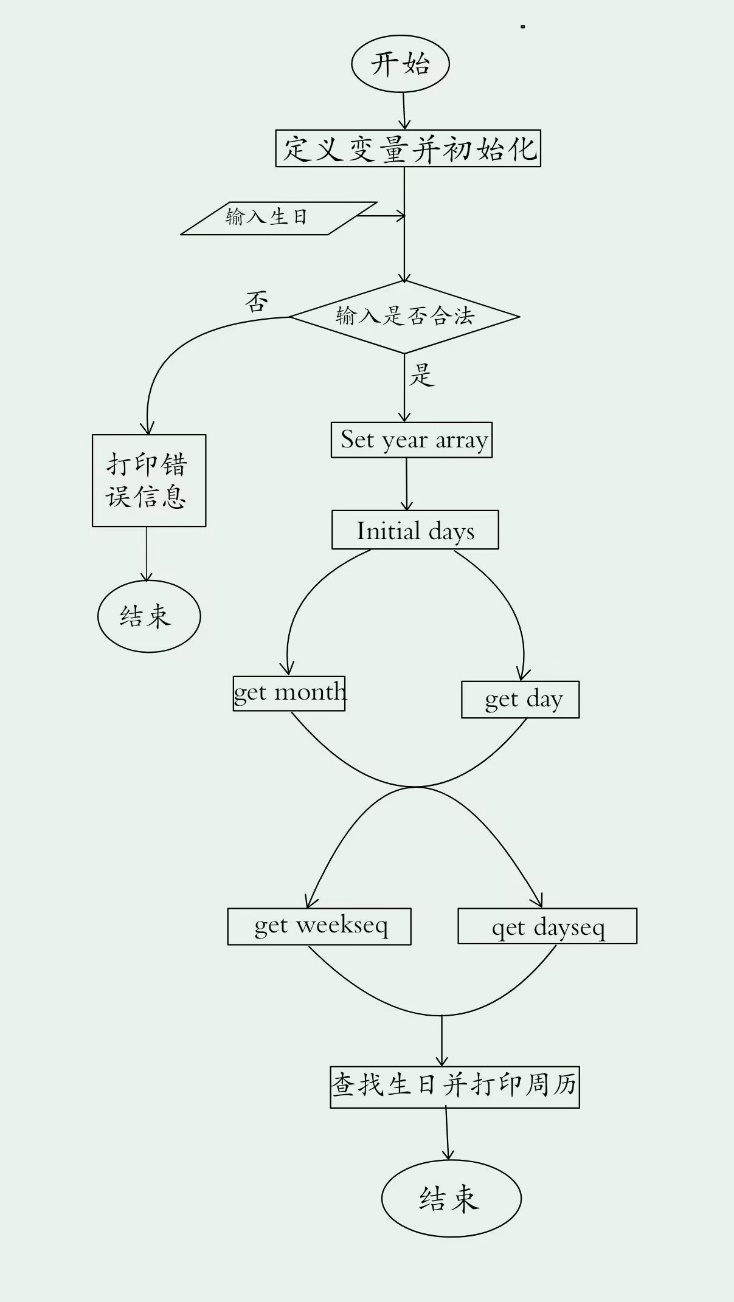
问题：第一次写程序时，我依然忘记了加入约束条件，但是还好我在输入前的提示里备注了输入的范围应该为（1~29），还没有错的太彻底。不过仅仅有提示是不够的，所以我在修改时加入了对输入数据是否在范围内的判断，防止输入超范围的数据导致结果错误或者程序崩溃。

# 实验五（多维数组寻找生日）

## 实验任务

-用三维数组记录多年的日期数据，查找某人的生日，并打印该生日所在周的周历，计算其在周末过生日的次数。

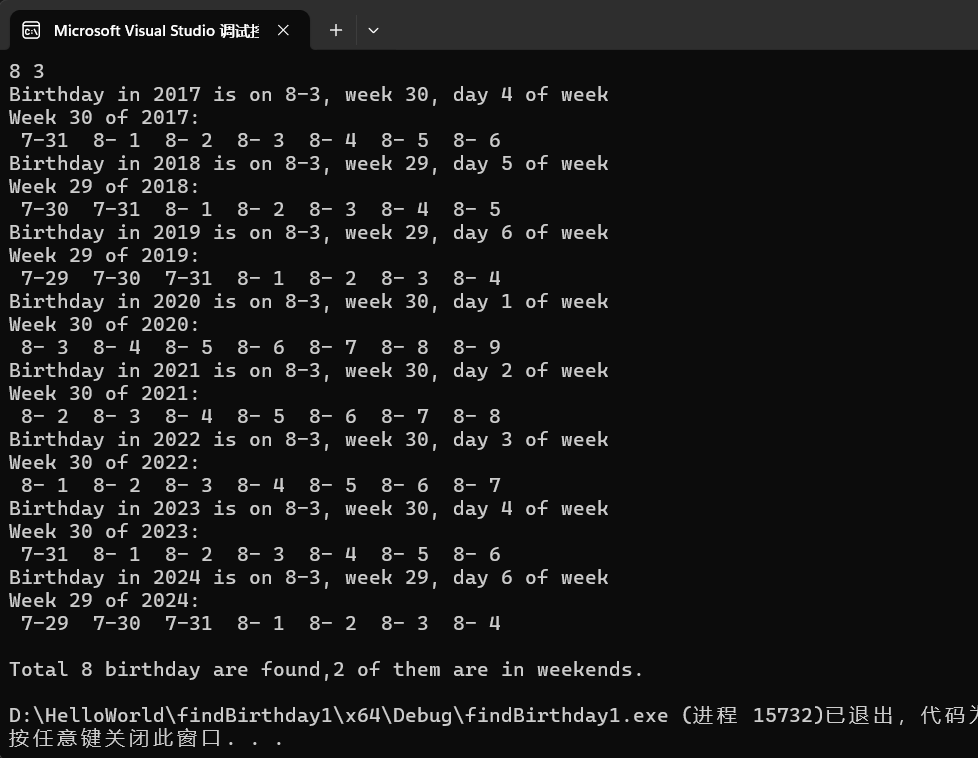
实验步骤



## 代码测试

### 样本点（8，3）的测试结果

输入生日为八月三日，测试程序能否正确查找范围内每一年某人的生日并且正确打印出生日所在的周历。

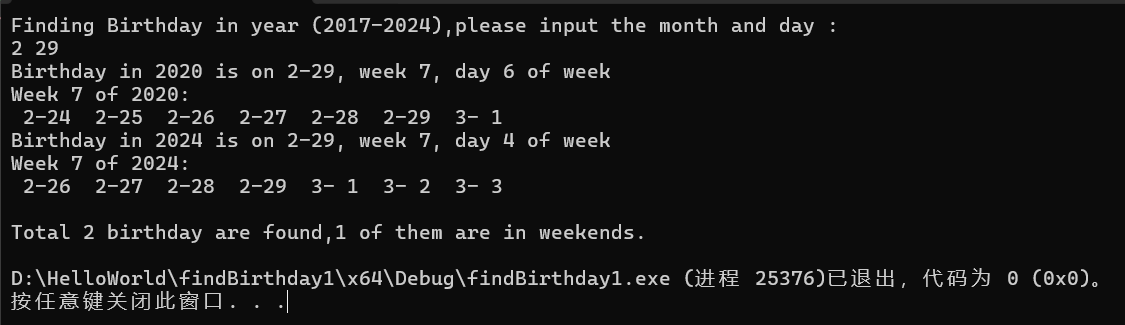




根据系统给出的2017年的日历，可以初步判断程序输出结果正确，测试通过。

### 样本点（2，29）的测试结果

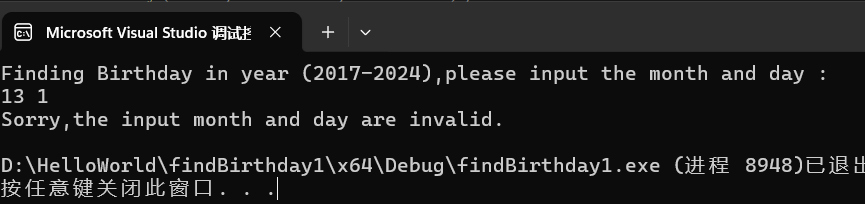
因为只有闰年才有2月29日，所以输入生日为2月29日，测试能否正确判断平闰年并且正确查找生日并输出周历。



测试结果只查找到两次生日，测试通过。

### 样本点（13，1）的测试结果

输入生日为13月1日，测试程序能否判断非法输入并且正确输出报错信息，结束程序。



结果显示报错信息，测试通过。

## 实验结论

该程序能够正确查找每一年的生日，并且用三维数组储存生日信息，然后输出相关信息，测试通过。

## 实验总结

感悟一：多维数组可以作为存储信息的方式

提升一：对多维数组掌握更熟练了

感悟二：要善于运用函数简化代码，如果只有一个main函数将会使得代码及其复杂而且难以观察结构，也难以debug。

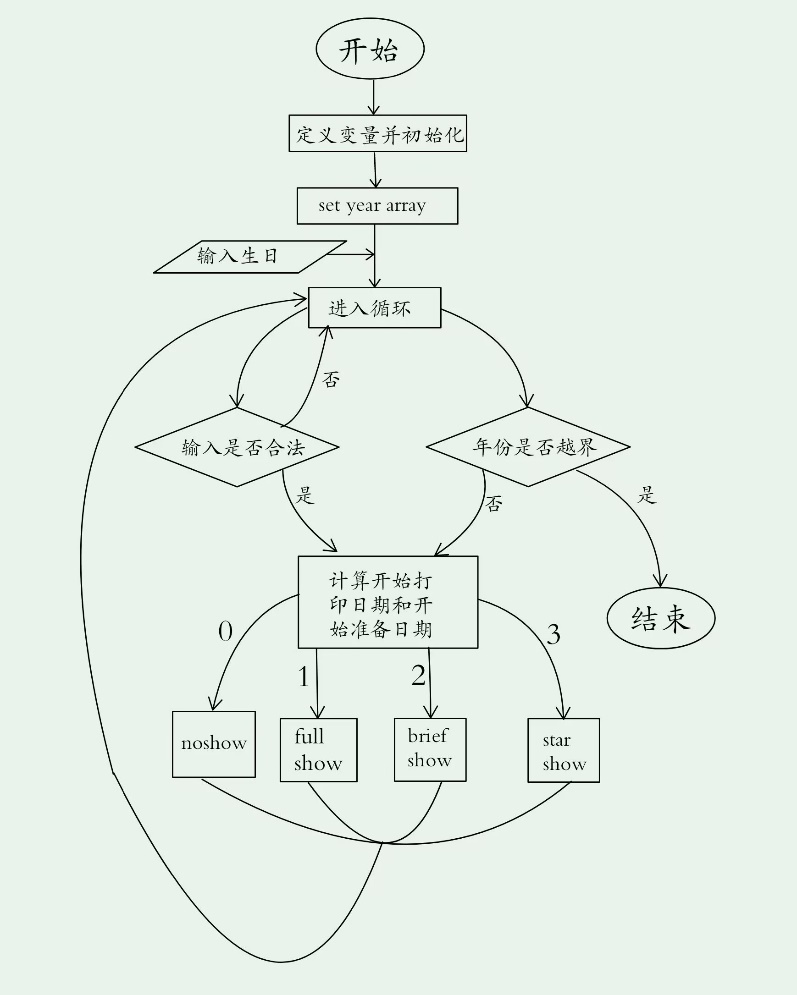
# 实验六（结构体寻找生日）

## 实验任务

-假定某生日趴需要三天时间准备，输入某人的生日，通过日期偏移计算获得前三天并打印相关周历

-用日期结构体记录单一日期的所有数据属性

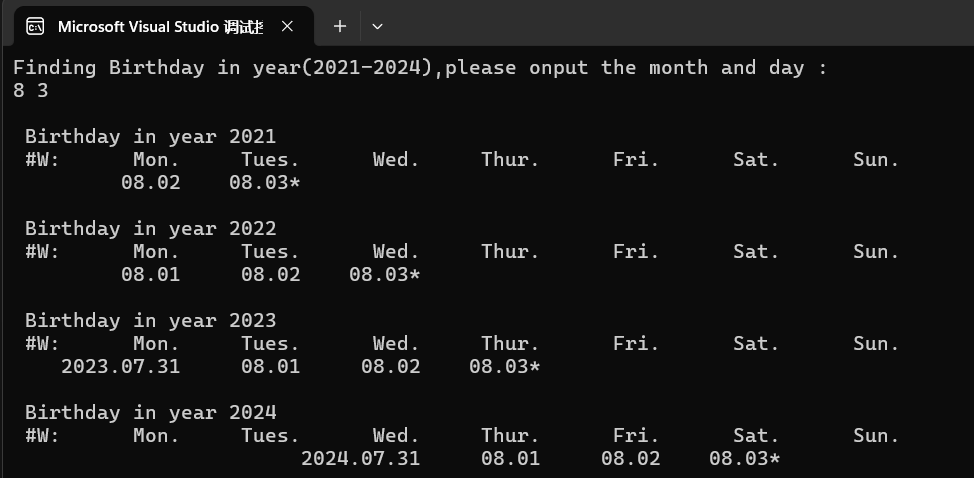
## 实验步骤

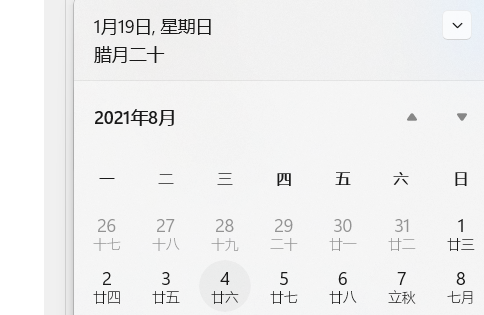


## 代码测试

### 样本点（8，3）的测试结果

输入生日为8月3日，测试程序能否正确获得前三天的信息并且按照格式正确打印出周历史。

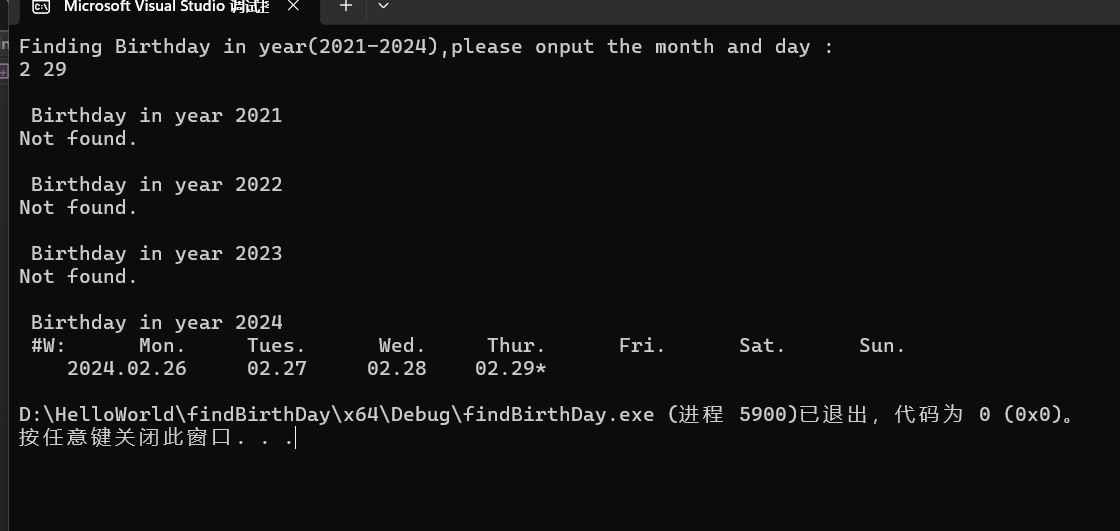




测试结果和实际对照，程序确实能正常输出周历，测试通过。

### 样本点（2，29）的测试结果

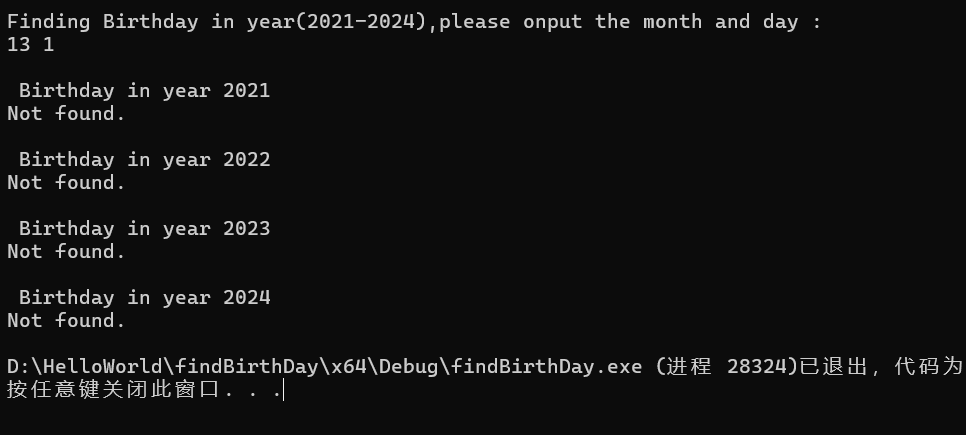
因为只有闰年才有2月29日，所以输入生日为2月29日，测试能否正确判断平闰年并且正确查找生日并输出周历。



测试结果是只有2024年查找到了生日，测试通过。

### 样本点（13，1）的测试结果

输入非法数据，测试程序能否正确打印出报错信息。

、

结果全部显示未找到生日，测试通过。

## 实验结论

该程序能够用结构体储存某一天的相关信息，并且能够通过输入的生日查找前三天和所在周的信息，并且正确打印结果，测试通过。

## 实验总结

通过该实验，我加深了对结构体的理解，对结构体使用掌握更熟练了。

我已经能够较快构思代码并且用c语言实现，缺点是我还是有一点马虎，大于小于号写反了一直没有检查出来，导致浪费很多时间。

# 本课程学习总结

## 学习C语言的历程和总结（心酸血泪史）

其实直接套用老师给的模板，连标题名字都不改，显得有点不走心。但是老师“心酸血泪史”五个字实在给的太贴切了，我想不到更好的标题了。

刚开始学习c语言确实是很心酸的，特别是对于我这样一个小镇做题家。从小我就不被允许使用电子产品，以至于刚开始上课的时候我连打字都不利索，打一个单词要找半天。听着老师滔滔不绝地输出着我听不懂的语法知识，看着同学们胸有成竹埋头敲代码，而我自己却不知所措，那一刻说不心酸都是骗人。

这之后我痛定思痛，决定从打字熟练度到c语言掌握度全方面提升自己的能力。那真的是一段血泪奋斗史。从来没有一点编程基础，加上学时的减少，我在课上很难跟上老师的进度，所以我下课后花费大量的时间去听b站上别人的知识点讲解。和有基础的同学比起来，我要花两倍三倍的时间，还不一定能做的和别人一样好。

第一次的编程作业，我的室友只花了五分钟就写完了，而我在晚上十点钟还望着题目发呆。我挣扎犹豫了很久，决定先把这道题放下，耐心看完c语言的网课。幸好在恶补了两天半的网课之后，我终于一拍脑袋想到了编程方法。按下编译运行键的那一刻我的手是在颤抖的，结果显示正确之后我立马给我妈打了个电话，很激动，但是却说不出来为什么激动。那对于别人只是五分钟的事情，我花了三天才做好，确实没什么好自豪的。

那之后我有了信心，能力却没有质的飞跃。我依然要低头找半天键位，我依然要花两个小时琢磨一段代码（头歌的作业我甚至是拍照之后在自己的软件上做好再进去测试的，因为不想然别人发现我一道题要写两个小时）。但是有了成功的经验便没了放弃的理由。每一次代码报错到崩溃的时候我都会告诉自己，当初那个代码小白都能通过自己的努力做出来，你现在学了更多的知识，怎么可能不会呢。我一定能编出来的信念支撑着我走过了和c语言鏖战的日日夜夜，带领我从无边夜幕行至天光。

每天早起，如果没有早八，就是我的c语言时间，我会先看老师的PPT，找到自己不会的知识点，再上b站找大佬的讲解，找相似的例题——听一遍再自己敲一遍。渐渐的，我敲代码不用再看键盘了，敲一段代码也不再需要花费几个小时，当我看着屏幕上闪过的一行行代码，我体会到了如飞翔的自由感，还有真真切切的自豪感。

进步来得缓慢且静悄悄，我常常发现不了，也因此常常着急得红了眼眶。但现在回头一看，之前的代码幼稚，现在的成熟了不是一星半点，很感谢老师布置的最后一次作业，如果不是这次作业，我恐怕永远不会再看那些有点可笑的代码，那些我含着泪敲完的代码。

风雨之后会见彩虹的，我还是这样相信着。如果我目前做的还是不够好，那一定是因为，我的风暴还没有完全远去，C语言，也不是结课之后就要束之高阁，前路漫长，我会继续努力。

## 程序中出现的问题和改进

问题一：以前我不擅长运用循环语句简化代码，总是写得很复杂。后来我找出了一些自己的代码进行修改，倒逼自己走出安全区，逼自己去使用不熟练的句法，渐渐有所进步。

问题二：我不喜欢写注释，因为一开始写的代码很简单，一目了然。然而代码渐渐从十行到一百行两百行，我迟到了偷懒的苦头，渐渐学会加入一些简明而必要的注释。

问题三：我老是分不清双等号和赋值符号，先在也常常分不清，幸好这类问题容易编译出错，方便我检查……不过最根本的应该是改掉自己马大哈的性格。

问题四：人机交互是一个永恒的命题，从前我不了解自己写的代码的本质，我总是觉得自己知道要输入什么就好了啊，所以在输入之前总是没有提示。是老师告诉我，我的代码是要给别人看的——“万一有一天你要写出流芳百世的代码呢”，从此我开始有了危机意识，努力做好细节，给出充足的提示，而且时刻注意越界等问题。

## 代码规范与调试技巧

代码规范：1.注意合理缩进

2.变量命名的规范化

3.适当的注释

4.代码尽量不要有无谓的重复句段

5.暂时想不到了

调试技巧：1.完成一个功能模块就编译一下，防止全编完之后迷迷糊糊找不到错在哪

2.在进函数的上一行标停（这不是废话，我以前就不知道，小白真的很容易忽略这些细节，没标停debug就不能看过程了，我还以为是我的电脑坏了）

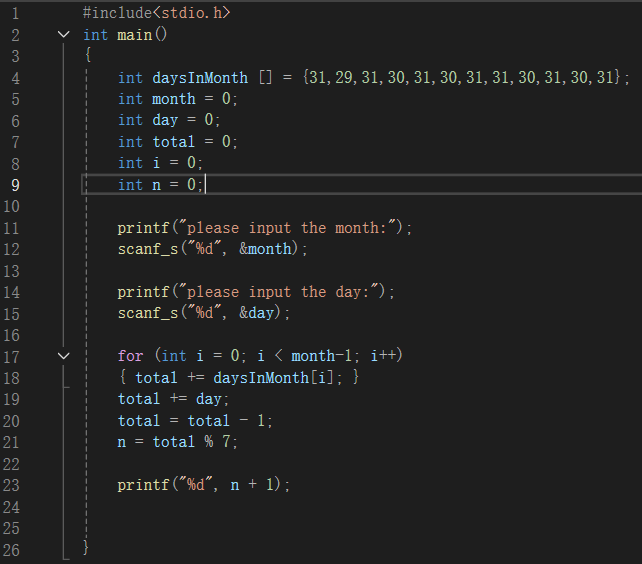
## 学习记录和心得

1. 每天早八黄金时间留给c
2. 每天睡前时间留给翁恺
3. 不惜一掷半天只为完成c语言作业
4. 心得就是努力加仔细加耐心检查能解决百分之九十九的问题
5. 还有千万不要把你的代码交给AI，因为可能会从两个报错变成一整页报错（这不是主要的原因，只有自己敲过才能体会为什么要这样做，AI代写是自己骗自己，最后结果会很悲惨）
6. 最大的心得：千万不要给自己电脑起中文名，尤其是像“忧郁王子”这样的，每次和同学讨论代码时弹出“忧郁王子”我都想钻地缝，而且很多软件不支持中文路径，比如vivado,这是真的血与泪的教训

# 附录

完整实验代码附在此处

1. 实验一（日历）



#include<stdio.h>

int main()

{

int daysInMonth [] = {31,29,31,30,31,30,31,31,30,31,30,31};

int month = 0;

int day = 0;

int total = 0;

int i = 0;

int n = 0;

printf("please input the month:");

scanf\_s("%d", &month);

printf("please input the day:");

scanf\_s("%d", &day);

for (int i = 0; i < month-1; i++)

{ total += daysInMonth[i]; }

total += day;

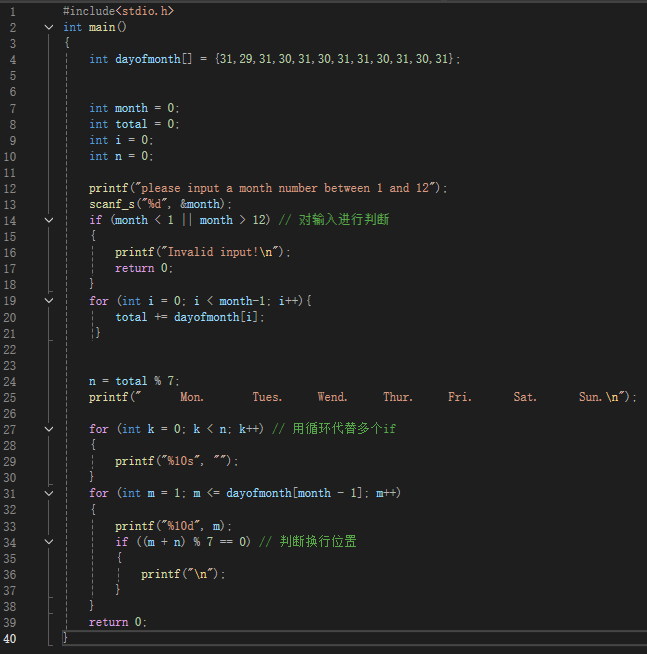
total = total - 1;

n = total % 7;

printf("%d", n + 1);

}

1. 实验二（月历）



#include<stdio.h>

int main()

{

int dayofmonth[] = {31,29,31,30,31,30,31,31,30,31,30,31};

int month = 0;

int total = 0;

int i = 0;

int n = 0;

printf("please input a month number between 1 and 12");

scanf\_s("%d", &month);

if (month < 1 || month > 12) // 对输入进行判断

{

printf("Invalid input!\n");

return 0;

}

for (int i = 0; i < month-1; i++){

total += dayofmonth[i];

}

n = total % 7;

printf(" Mon. Tues. Wend. Thur. Fri. Sat. Sun.\n");

for (int k = 0; k < n; k++) // 用循环代替多个if

{

printf("%10s", "");

}

for (int m = 1; m <= dayofmonth[month - 1]; m++)

{

printf("%10d", m);

if ((m + n) % 7 == 0) // 判断换行位置

{

printf("\n");

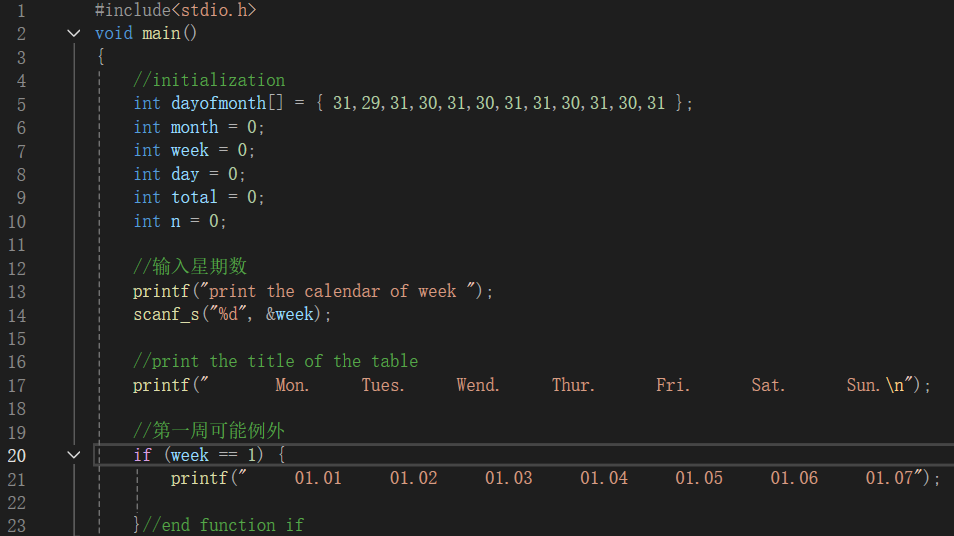
}

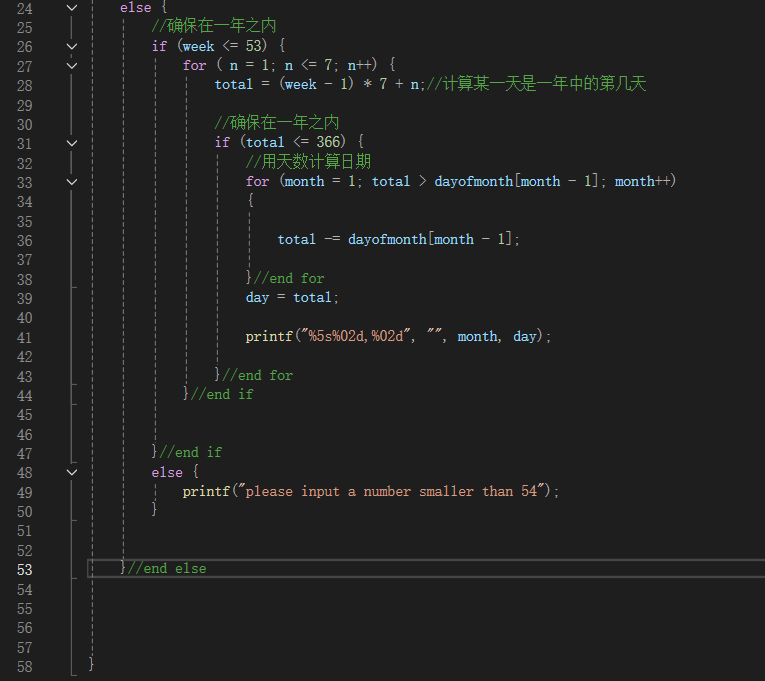
}

return 0;

}

三、实验三（周历）





#include<stdio.h>

void main()

{

//initialization

int dayofmonth[] = { 31,29,31,30,31,30,31,31,30,31,30,31 };

int month = 0;

int week = 0;

int day = 0;

int total = 0;

int n = 0;

//输入星期数

printf("print the calendar of week ");

scanf\_s("%d", &week);

//print the title of the table

printf(" Mon. Tues. Wend. Thur. Fri. Sat. Sun.\n");

//第一周可能例外

if (week == 1) {

printf(" 01.01 01.02 01.03 01.04 01.05 01.06 01.07");

}//end function if

else {

//确保在一年之内

if (week <= 53) {

for ( n = 1; n <= 7; n++) {

total = (week - 1) \* 7 + n;//计算某一天是一年中的第几天

//确保在一年之内

if (total <= 366) {

//用天数计算日期

for (month = 1; total > dayofmonth[month - 1]; month++)

{

total -= dayofmonth[month - 1];

}//end for

day = total;

printf("%5s%02d,%02d", "", month, day);

}//end for

}//end if

}//end if

else {

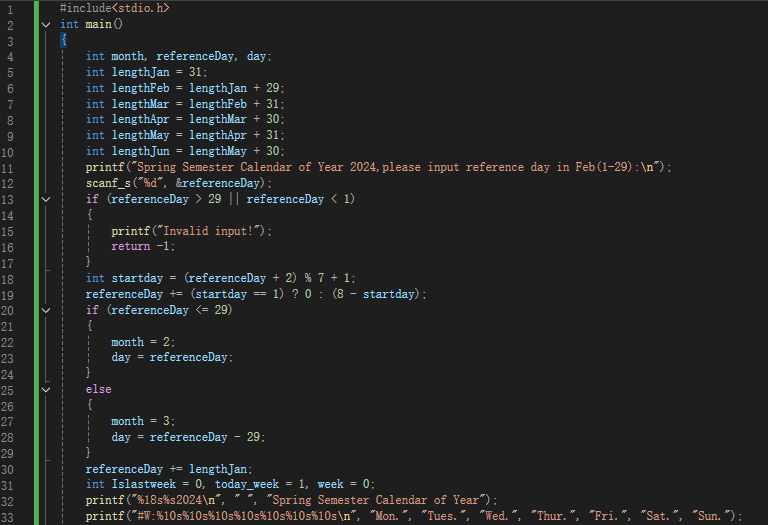
printf("please input a number smaller than 54");

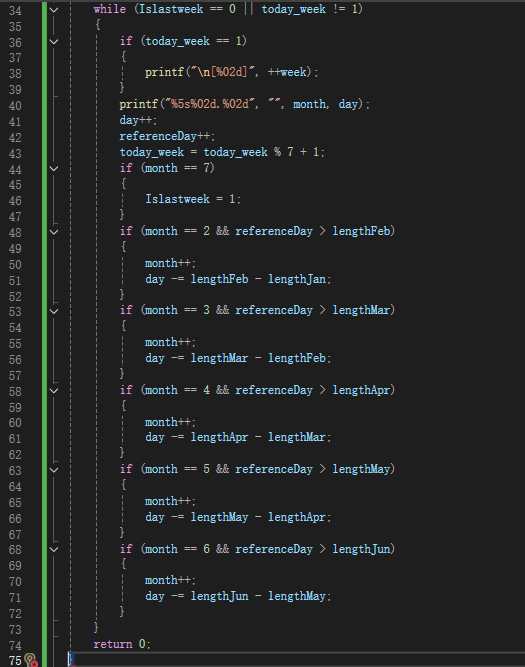
}

}//end else

}

四、实验四（春季校历）





#include<stdio.h>

int main()

{

int month, referenceDay, day;

int lengthJan = 31;

int lengthFeb = lengthJan + 29;

int lengthMar = lengthFeb + 31;

int lengthApr = lengthMar + 30;

int lengthMay = lengthApr + 31;

int lengthJun = lengthMay + 30;

printf("Spring Semester Calendar of Year 2024,please input reference day in Feb(1-29):\n");

scanf\_s("%d", &referenceDay);

if (referenceDay > 29 || referenceDay < 1)

{

printf("Invalid input!");

return -1;

}

int startday = (referenceDay + 2) % 7 + 1;

referenceDay += (startday == 1) ? 0 : (8 - startday);

if (referenceDay <= 29)

{

month = 2;

day = referenceDay;

}

else

{

month = 3;

day = referenceDay - 29;

}

referenceDay += lengthJan;

int Islastweek = 0, today\_week = 1, week = 0;

printf("%18s%s2024\n", " ", "Spring Semester Calendar of Year");

printf("#W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues.", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");

while (Islastweek == 0 || today\_week != 1)

{

if (today\_week == 1)

{

printf("\n[%02d]", ++week);

}

printf("%5s%02d.%02d", "", month, day);

day++;

referenceDay++;

today\_week = today\_week % 7 + 1;

if (month == 7)

{

Islastweek = 1;

}

if (month == 2 && referenceDay > lengthFeb)

{

month++;

day -= lengthFeb - lengthJan;

}

if (month == 3 && referenceDay > lengthMar)

{

month++;

day -= lengthMar - lengthFeb;

}

if (month == 4 && referenceDay > lengthApr)

{

month++;

day -= lengthApr - lengthMar;

}

if (month == 5 && referenceDay > lengthMay)

{

month++;

day -= lengthMay - lengthApr;

}

if (month == 6 && referenceDay > lengthJun)

{

month++;

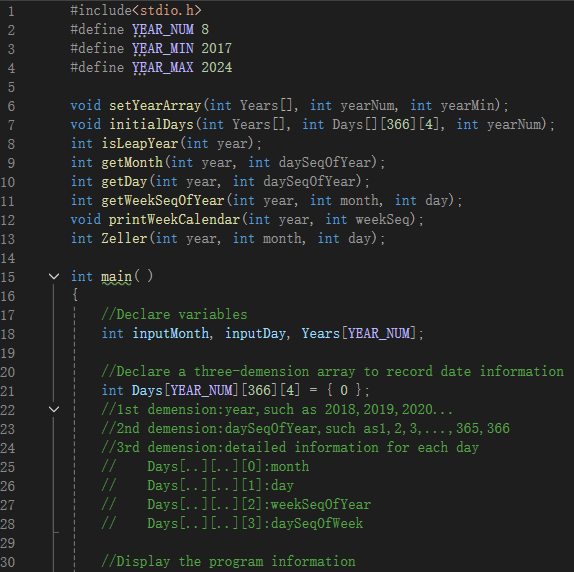
day -= lengthJun - lengthMay;

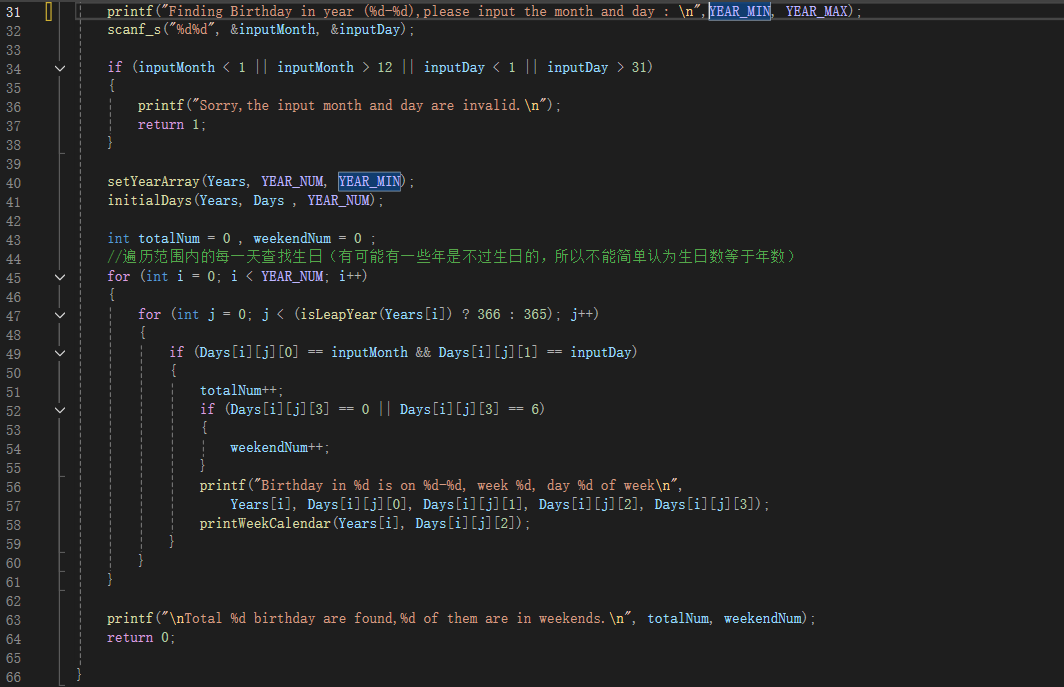
}

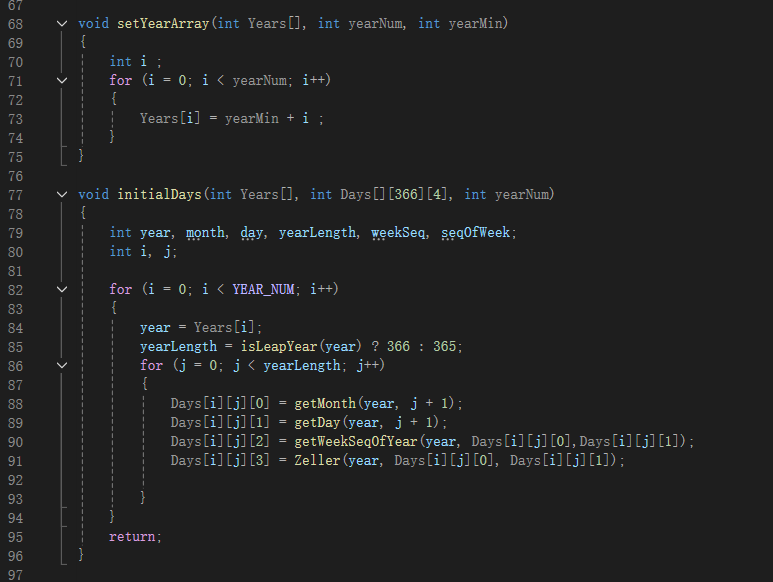
}

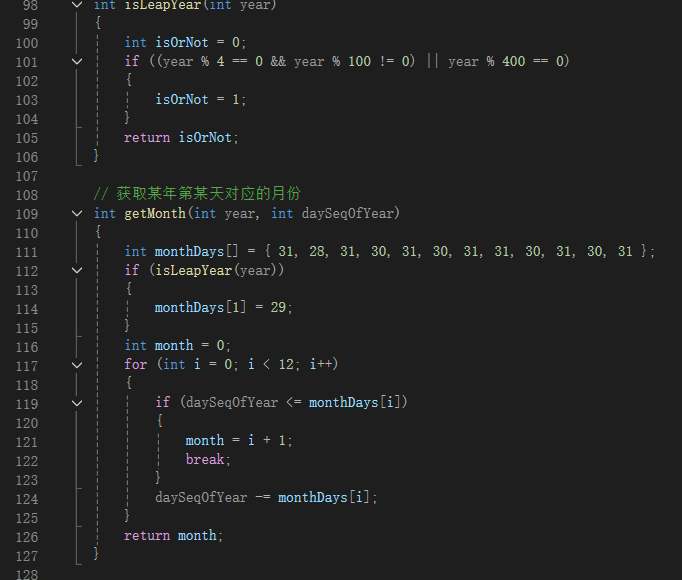
return 0;

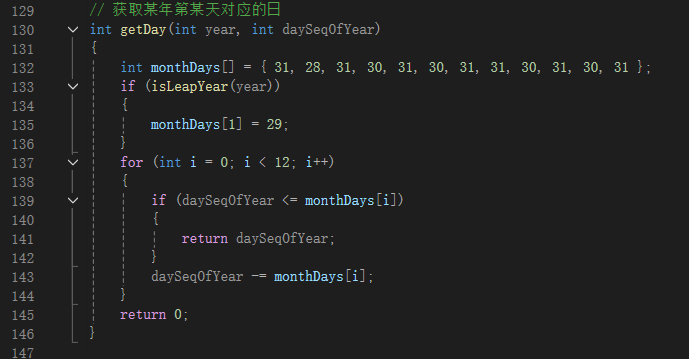
五、实验五（多维数组寻找生日）

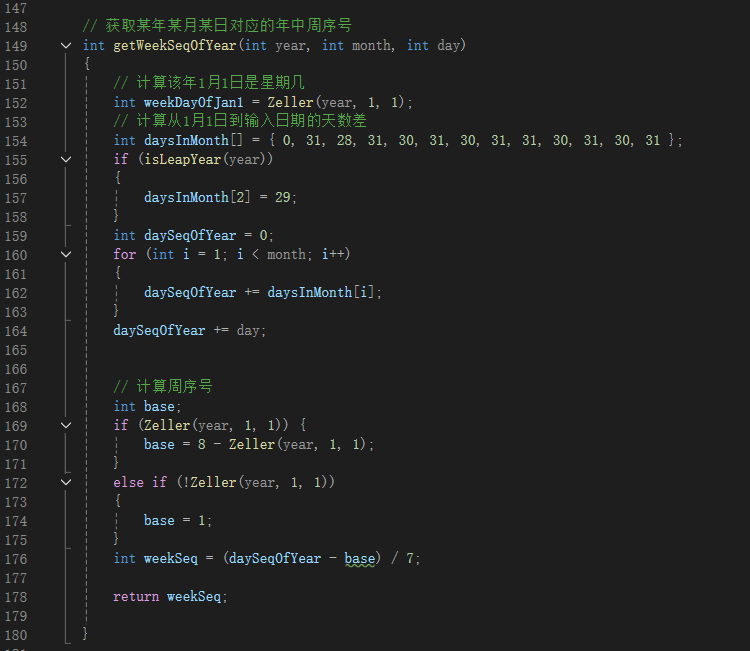




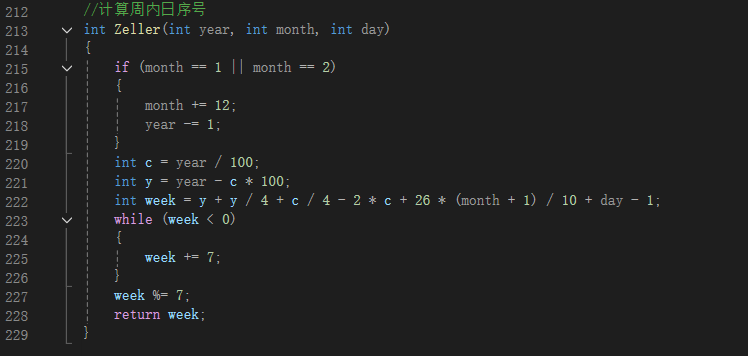












#include<stdio.h>

#define YEAR\_NUM 8

#define YEAR\_MIN 2017

#define YEAR\_MAX 2024

void setYearArray(int Years[], int yearNum, int yearMin);

void initialDays(int Years[], int Days[][366][4], int yearNum);

int isLeapYear(int year);

int getMonth(int year, int daySeqOfYear);

int getDay(int year, int daySeqOfYear);

int getWeekSeqOfYear(int year, int month, int day);

void printWeekCalendar(int year, int weekSeq);

int Zeller(int year, int month, int day);

int main( )

{

//Declare variables

int inputMonth, inputDay, Years[YEAR\_NUM];

//Declare a three-demension array to record date information

int Days[YEAR\_NUM][366][4] = { 0 };

//1st demension:year,such as 2018,2019,2020...

//2nd demension:daySeqOfYear,such as1,2,3,...,365,366

//3rd demension:detailed information for each day

// Days[..][..][0]:month

// Days[..][..][1]:day

// Days[..][..][2]:weekSeqOfYear

// Days[..][..][3]:daySeqOfWeek

//Display the program information

printf("Finding Birthday in year (%d-%d),please input the month and day : \n",YEAR\_MIN, YEAR\_MAX);

scanf\_s("%d%d", &inputMonth, &inputDay);

if (inputMonth < 1 || inputMonth > 12 || inputDay < 1 || inputDay > 31)

{

printf("Sorry,the input month and day are invalid.\n");

return 1;

}

setYearArray(Years, YEAR\_NUM, YEAR\_MIN);

initialDays(Years, Days , YEAR\_NUM);

int totalNum = 0 , weekendNum = 0 ;

//遍历范围内的每一天查找生日（有可能有一些年是不过生日的，所以不能简单认为生日数等于年数）

for (int i = 0; i < YEAR\_NUM; i++)

{

for (int j = 0; j < (isLeapYear(Years[i]) ? 366 : 365); j++)

{

if (Days[i][j][0] == inputMonth && Days[i][j][1] == inputDay)

{

totalNum++;

if (Days[i][j][3] == 0 || Days[i][j][3] == 6)

{

weekendNum++;

}

printf("Birthday in %d is on %d-%d, week %d, day %d of week\n",

Years[i], Days[i][j][0], Days[i][j][1], Days[i][j][2], Days[i][j][3]);

printWeekCalendar(Years[i], Days[i][j][2]);

}

}

}

printf("\nTotal %d birthday are found,%d of them are in weekends.\n", totalNum, weekendNum);

return 0;

}

void setYearArray(int Years[], int yearNum, int yearMin)

{

int i ;

for (i = 0; i < yearNum; i++)

{

Years[i] = yearMin + i ;

}

}

void initialDays(int Years[], int Days[][366][4], int yearNum)

{

int year, month, day, yearLength, weekSeq, seqOfWeek;

int i, j;

for (i = 0; i < YEAR\_NUM; i++)

{

year = Years[i];

yearLength = isLeapYear(year) ? 366 : 365;

for (j = 0; j < yearLength; j++)

{

Days[i][j][0] = getMonth(year, j + 1);

Days[i][j][1] = getDay(year, j + 1);

Days[i][j][2] = getWeekSeqOfYear(year, Days[i][j][0],Days[i][j][1]);

Days[i][j][3] = Zeller(year, Days[i][j][0], Days[i][j][1]);

}

}

return;

}

int isLeapYear(int year)

{

int isOrNot = 0;

if ((year % 4 == 0 && year % 100 != 0) || year % 400 == 0)

{

isOrNot = 1;

}

return isOrNot;

}

// 获取某年第某天对应的月份

int getMonth(int year, int daySeqOfYear)

{

int monthDays[] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

if (isLeapYear(year))

{

monthDays[1] = 29;

}

int month = 0;

for (int i = 0; i < 12; i++)

{

if (daySeqOfYear <= monthDays[i])

{

month = i + 1;

break;

}

daySeqOfYear -= monthDays[i];

}

return month;

}

// 获取某年第某天对应的日

int getDay(int year, int daySeqOfYear)

{

int monthDays[] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

if (isLeapYear(year))

{

monthDays[1] = 29;

}

for (int i = 0; i < 12; i++)

{

if (daySeqOfYear <= monthDays[i])

{

return daySeqOfYear;

}

daySeqOfYear -= monthDays[i];

}

return 0;

}

// 获取某年某月某日对应的年中周序号

int getWeekSeqOfYear(int year, int month, int day)

{

// 计算该年1月1日是星期几

int weekDayOfJan1 = Zeller(year, 1, 1);

// 计算从1月1日到输入日期的天数差

int daysInMonth[] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

if (isLeapYear(year))

{

daysInMonth[2] = 29;

}

int daySeqOfYear = 0;

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

{

daySeqOfYear += daysInMonth[i];

}

daySeqOfYear += day;

// 计算周序号

int base;

if (Zeller(year, 1, 1)) {

base = 8 - Zeller(year, 1, 1);

}

else if (!Zeller(year, 1, 1))

{

base = 1;

}

int weekSeq = (daySeqOfYear - base) / 7;

return weekSeq;

}

// 打印某年某周的周历

void printWeekCalendar(int year, int weekSeq) {

int monthDays[] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

if (isLeapYear(year)) {

monthDays[2] = 29;

}

int base;

if (Zeller(year, 1, 1)) {

base = 8 - Zeller(year, 1, 1);

}

else if (!Zeller(year, 1, 1))

{

base = 1;

}

int daySeqOfYear = weekSeq \* 7 + 1 + base;

int month = getMonth(year, daySeqOfYear);

int day = getDay(year, daySeqOfYear);

printf("Week %d of %d:\n", weekSeq, year);

for (int i = 0; i < 7; i++) {

month = getMonth(year, daySeqOfYear + i);

day = getDay(year, daySeqOfYear + i);

printf("%2d-%2d ", month, day); // Improved formatting

}

printf("\n");

}

//计算周内日序号

int Zeller(int year, int month, int day)

{

if (month == 1 || month == 2)

{

month += 12;

year -= 1;

}

int c = year / 100;

int y = year - c \* 100;

int week = y + y / 4 + c / 4 - 2 \* c + 26 \* (month + 1) / 10 + day - 1;

while (week < 0)

{

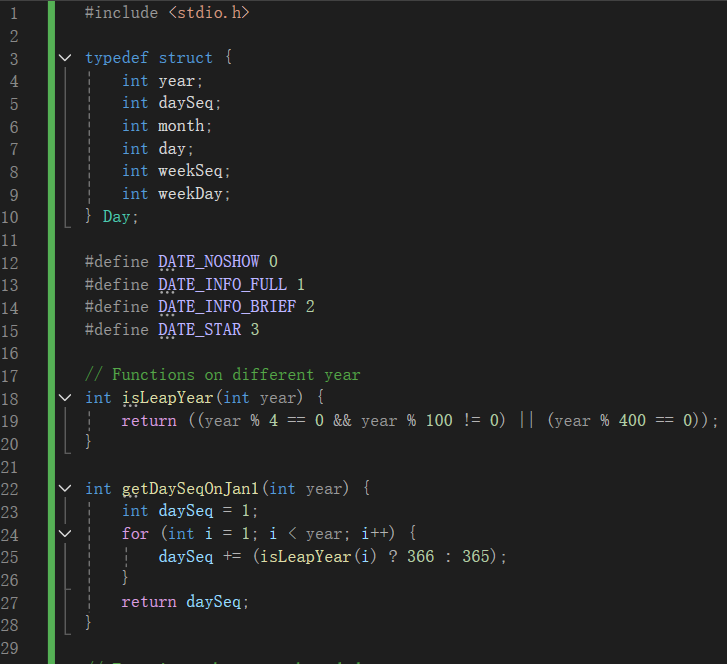
week += 7;

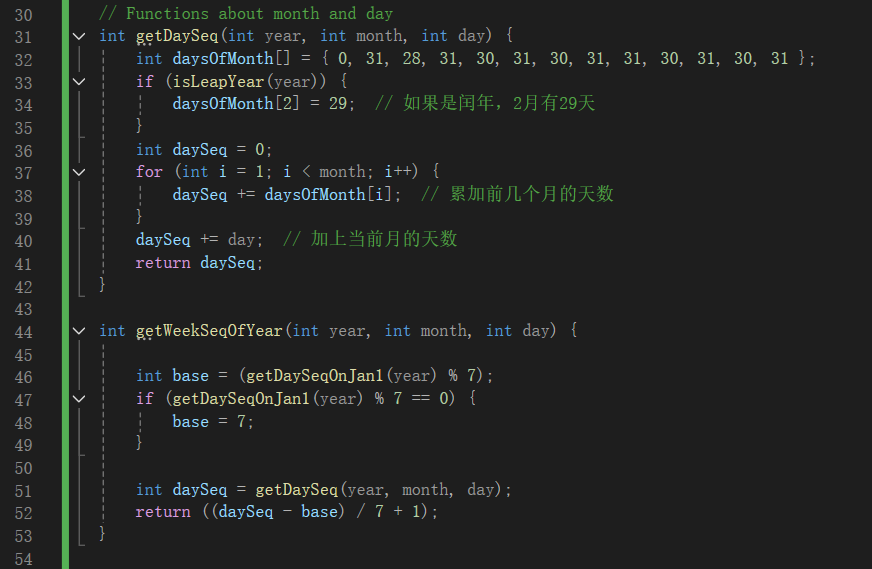
}

week %= 7;

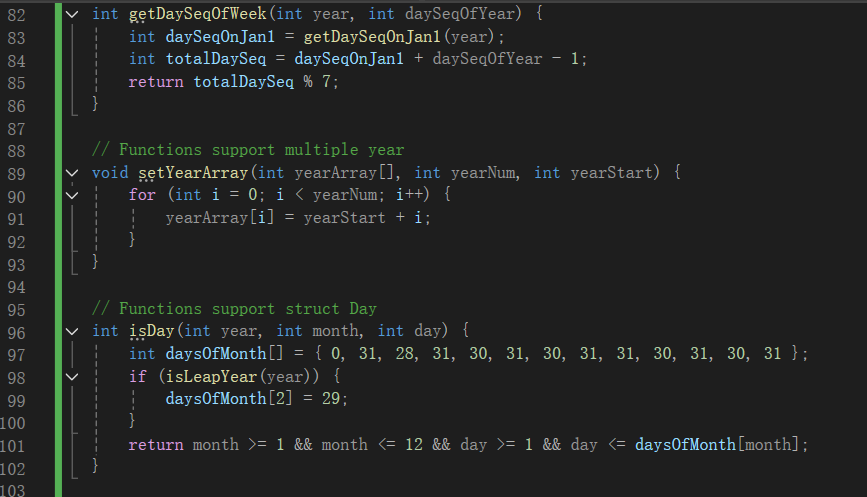
return week;

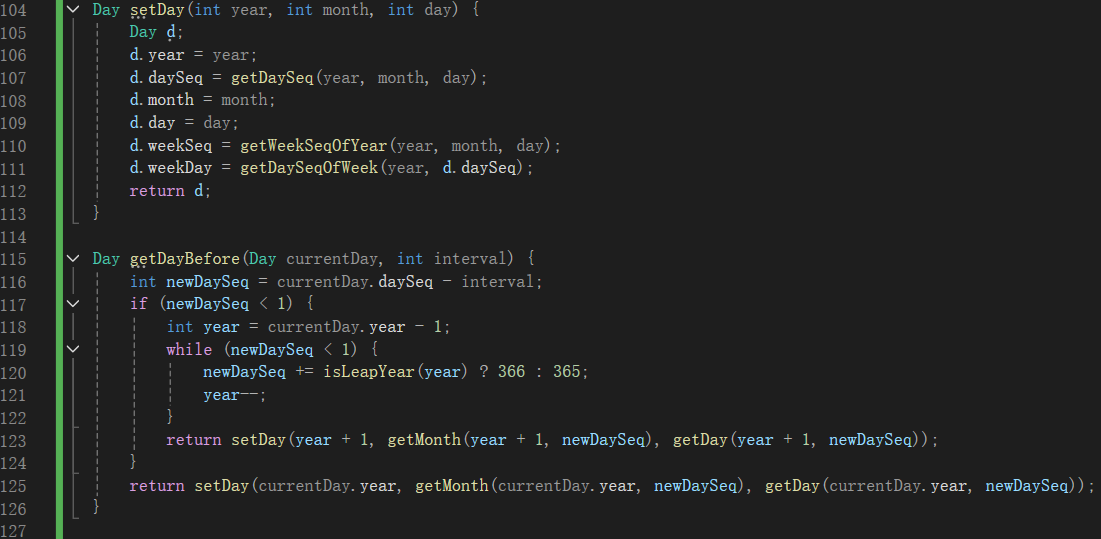
六。、实验六（结构体寻找生日）

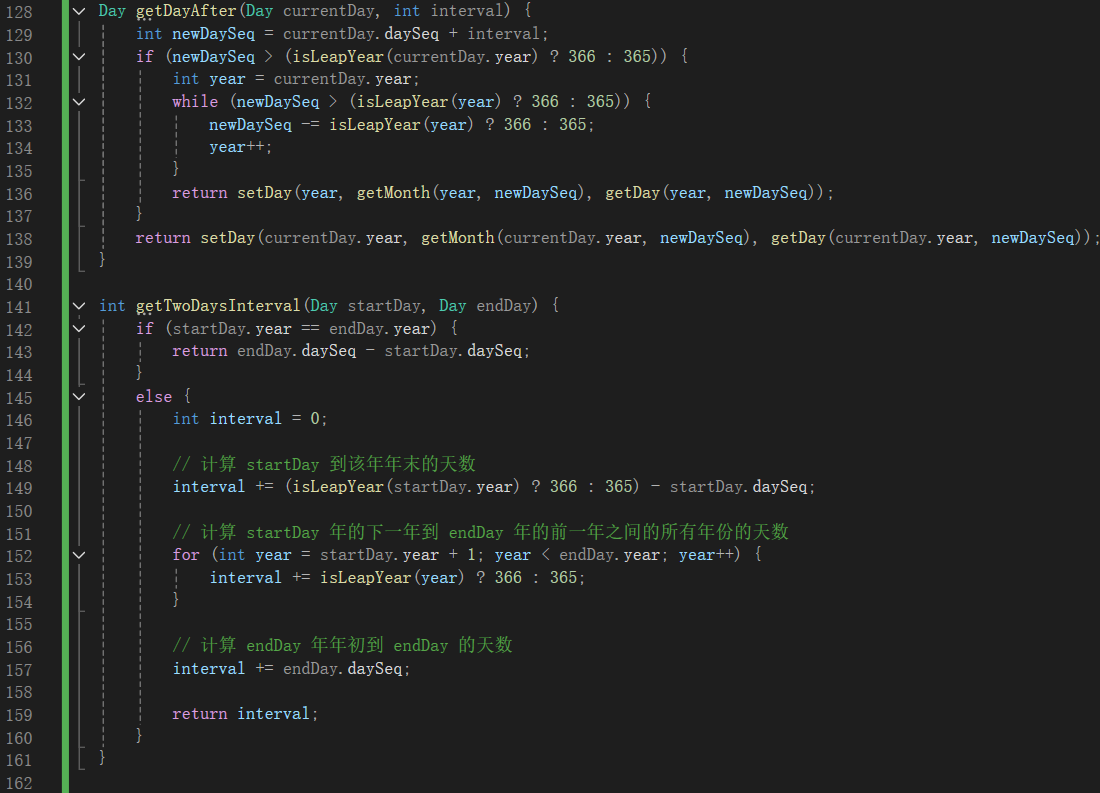


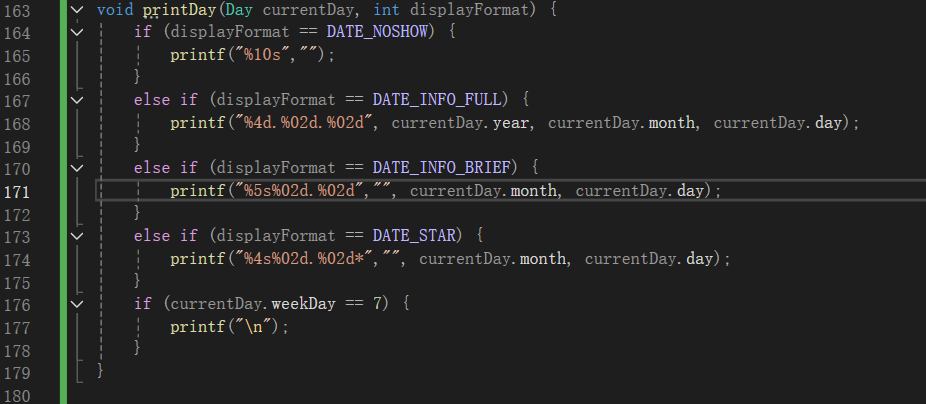


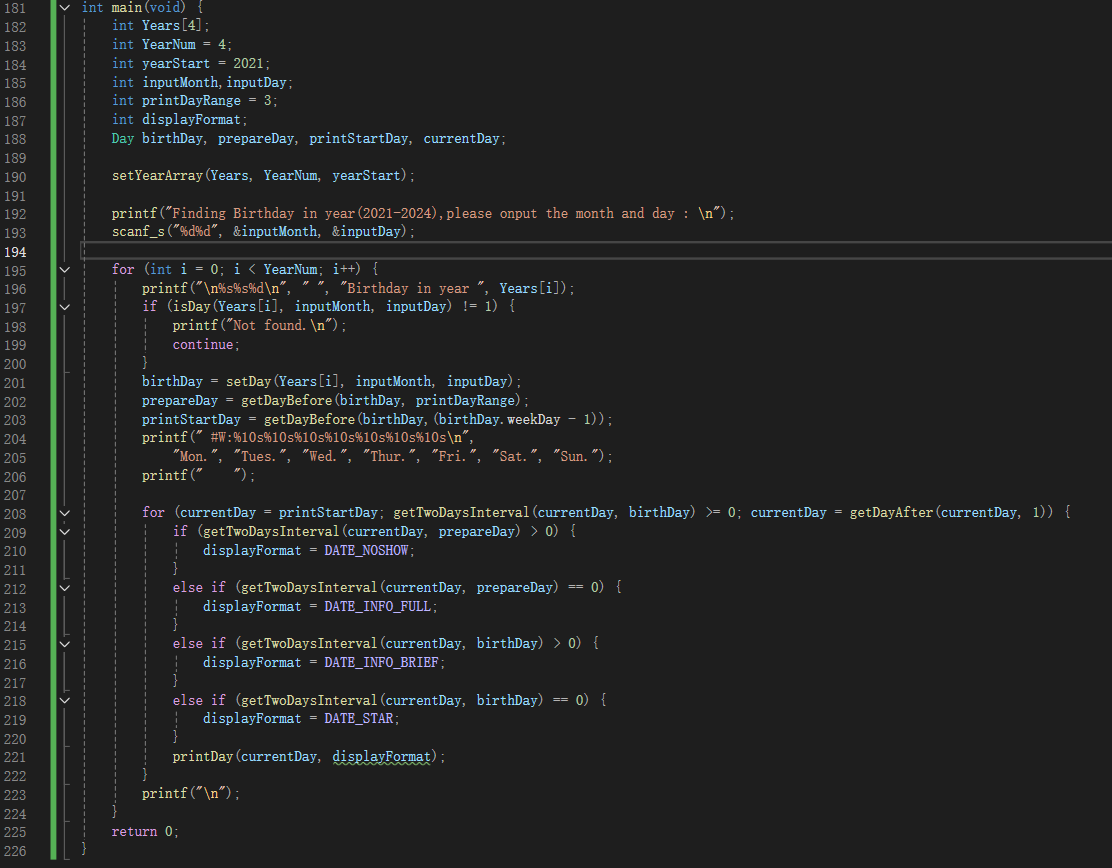




****

****

****

****

**#include <stdio.h>**

**typedef struct {**

**int year;**

**int daySeq;**

**int month;**

**int day;**

**int weekSeq;**

**int weekDay;**

**} Day;**

**#define DATE\_NOSHOW 0**

**#define DATE\_INFO\_FULL 1**

**#define DATE\_INFO\_BRIEF 2**

**#define DATE\_STAR 3**

**// Functions on different year**

**int isLeapYear(int year) {**

**return ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0));**

**}**

**int getDaySeqOnJan1(int year) {**

**int daySeq = 1;**

**for (int i = 1; i < year; i++) {**

**daySeq += (isLeapYear(i) ? 366 : 365);**

**}**

**return daySeq;**

**}**

**// Functions about month and day**

**int getDaySeq(int year, int month, int day) {**

**int daysOfMonth[] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };**

**if (isLeapYear(year)) {**

**daysOfMonth[2] = 29; // 如果是闰年，2月有29天**

**}**

**int daySeq = 0;**

**for (int i = 1; i < month; i++) {**

**daySeq += daysOfMonth[i]; // 累加前几个月的天数**

**}**

**daySeq += day; // 加上当前月的天数**

**return daySeq;**

**}**

**int getWeekSeqOfYear(int year, int month, int day) {**

**int base = (getDaySeqOnJan1(year) % 7);**

**if (getDaySeqOnJan1(year) % 7 == 0) {**

**base = 7;**

**}**

**int daySeq = getDaySeq(year, month, day);**

**return ((daySeq - base) / 7 + 1);**

**}**

**// Functions about properties of one day**

**int getMonth(int year, int daySeqOfYear) {**

**int daysOfMonth[] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };**

**if (isLeapYear(year)) {**

**daysOfMonth[2] = 29;**

**}**

**int month = 1;**

**while (daySeqOfYear > daysOfMonth[month]) {**

**daySeqOfYear -= daysOfMonth[month];**

**month++;**

**}**

**return month;**

**}**

**int getDay(int year, int daySeqOfYear) {**

**int daysOfMonth[] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };**

**if (isLeapYear(year)) {**

**daysOfMonth[2] = 29;**

**}**

**int month = 1;**

**while (daySeqOfYear > daysOfMonth[month]) {**

**daySeqOfYear -= daysOfMonth[month];**

**month++;**

**}**

**return daySeqOfYear;**

**}**

**int getDaySeqOfWeek(int year, int daySeqOfYear) {**

**int daySeqOnJan1 = getDaySeqOnJan1(year);**

**int totalDaySeq = daySeqOnJan1 + daySeqOfYear - 1;**

**return totalDaySeq % 7;**

**}**

**// Functions support multiple year**

**void setYearArray(int yearArray[], int yearNum, int yearStart) {**

**for (int i = 0; i < yearNum; i++) {**

**yearArray[i] = yearStart + i;**

**}**

**}**

**// Functions support struct Day**

**int isDay(int year, int month, int day) {**

**int daysOfMonth[] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };**

**if (isLeapYear(year)) {**

**daysOfMonth[2] = 29;**

**}**

**return month >= 1 && month <= 12 && day >= 1 && day <= daysOfMonth[month];**

**}**

**Day setDay(int year, int month, int day) {**

**Day d;**

**d.year = year;**

**d.daySeq = getDaySeq(year, month, day);**

**d.month = month;**

**d.day = day;**

**d.weekSeq = getWeekSeqOfYear(year, month, day);**

**d.weekDay = getDaySeqOfWeek(year, d.daySeq);**

**return d;**

**}**

**Day getDayBefore(Day currentDay, int interval) {**

**int newDaySeq = currentDay.daySeq - interval;**

**if (newDaySeq < 1) {**

**int year = currentDay.year - 1;**

**while (newDaySeq < 1) {**

**newDaySeq += isLeapYear(year) ? 366 : 365;**

**year--;**

**}**

**return setDay(year + 1, getMonth(year + 1, newDaySeq), getDay(year + 1, newDaySeq));**

**}**

**return setDay(currentDay.year, getMonth(currentDay.year, newDaySeq), getDay(currentDay.year, newDaySeq));**

**}**

**Day getDayAfter(Day currentDay, int interval) {**

**int newDaySeq = currentDay.daySeq + interval;**

**if (newDaySeq > (isLeapYear(currentDay.year) ? 366 : 365)) {**

**int year = currentDay.year;**

**while (newDaySeq > (isLeapYear(year) ? 366 : 365)) {**

**newDaySeq -= isLeapYear(year) ? 366 : 365;**

**year++;**

**}**

**return setDay(year, getMonth(year, newDaySeq), getDay(year, newDaySeq));**

**}**

**return setDay(currentDay.year, getMonth(currentDay.year, newDaySeq), getDay(currentDay.year, newDaySeq));**

**}**

**int getTwoDaysInterval(Day startDay, Day endDay) {**

**if (startDay.year == endDay.year) {**

**return endDay.daySeq - startDay.daySeq;**

**}**

**else {**

**int interval = 0;**

**// 计算 startDay 到该年年末的天数**

**interval += (isLeapYear(startDay.year) ? 366 : 365) - startDay.daySeq;**

**// 计算 startDay 年的下一年到 endDay 年的前一年之间的所有年份的天数**

**for (int year = startDay.year + 1; year < endDay.year; year++) {**

**interval += isLeapYear(year) ? 366 : 365;**

**}**

**// 计算 endDay 年年初到 endDay 的天数**

**interval += endDay.daySeq;**

**return interval;**

**}**

**}**

**void printDay(Day currentDay, int displayFormat) {**

**if (displayFormat == DATE\_NOSHOW) {**

**printf("%10s","");**

**}**

**else if (displayFormat == DATE\_INFO\_FULL) {**

**printf("%4d.%02d.%02d", currentDay.year, currentDay.month, currentDay.day);**

**}**

**else if (displayFormat == DATE\_INFO\_BRIEF) {**

**printf("%5s%02d.%02d","", currentDay.month, currentDay.day);**

**}**

**else if (displayFormat == DATE\_STAR) {**

**printf("%4s%02d.%02d\*","", currentDay.month, currentDay.day);**

**}**

**if (currentDay.weekDay == 7) {**

**printf("\n");**

**}**

**}**

**int main(void) {**

**int Years[4];**

**int YearNum = 4;**

**int yearStart = 2021;**

**int inputMonth,inputDay;**

**int printDayRange = 3;**

**int displayFormat;**

**Day birthDay, prepareDay, printStartDay, currentDay;**

**setYearArray(Years, YearNum, yearStart);**

**printf("Finding Birthday in year(2021-2024),please onput the month and day : \n");**

**scanf\_s("%d%d", &inputMonth, &inputDay);**

**for (int i = 0; i < YearNum; i++) {**

**printf("\n%s%s%d\n", " ", "Birthday in year ", Years[i]);**

**if (isDay(Years[i], inputMonth, inputDay) != 1) {**

**printf("Not found.\n");**

**continue;**

**}**

**birthDay = setDay(Years[i], inputMonth, inputDay);**

**prepareDay = getDayBefore(birthDay, printDayRange);**

**printStartDay = getDayBefore(birthDay,(birthDay.weekDay - 1));**

**printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n",**

**"Mon.", "Tues.", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");**

**printf(" ");**

**for (currentDay = printStartDay; getTwoDaysInterval(currentDay, birthDay) >= 0; currentDay = getDayAfter(currentDay, 1)) {**

**if (getTwoDaysInterval(currentDay, prepareDay) > 0) {**

**displayFormat = DATE\_NOSHOW;**

**}**

**else if (getTwoDaysInterval(currentDay, prepareDay) == 0) {**

**displayFormat = DATE\_INFO\_FULL;**

**}**

**else if (getTwoDaysInterval(currentDay, birthDay) > 0) {**

**displayFormat = DATE\_INFO\_BRIEF;**

**}**

**else if (getTwoDaysInterval(currentDay, birthDay) == 0) {**

**displayFormat = DATE\_STAR;**

**}**

**printDay(currentDay, displayFormat);**

**}**

**printf("\n");**

**}**

**return 0;**

**}**