

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 | 黄彦哲 | 学号 | U202413657 |

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 2024.12 | 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 11

编程工具：CodeBlocks 16.01

# 实验一

## 实验任务

Calendar1.计算星期几

[实验目标]

学习使用算术表达式，完成简单的功能。

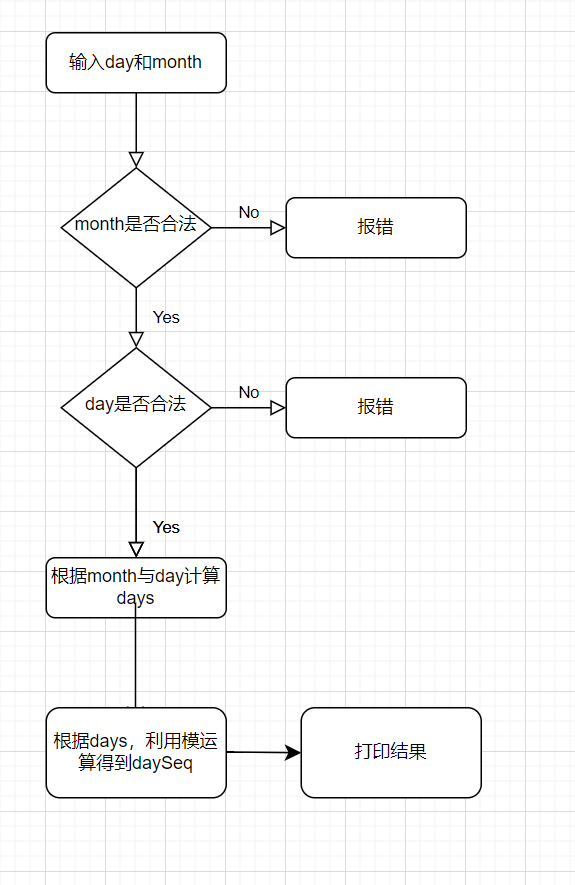
[实验任务]

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。

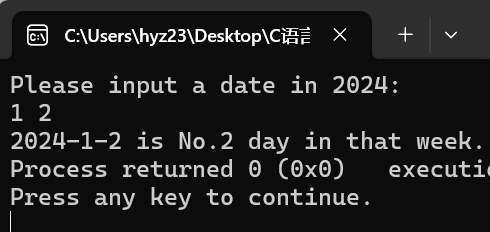
约定每个期从周一开始，如果是周一则打印1，周二则打印2，…，周日打印7。

## 实验步骤

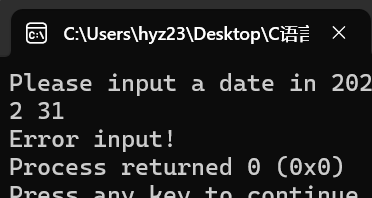
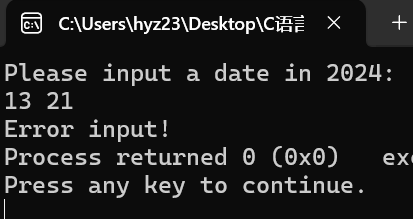


## 代码测试

### 代码能否正确输出结果



### 代码能否正确报错



## 实验结论

代码成功实现了预期的目标！

## 实验总结

该实验代码是在后期补写的，我认为主要问题在于如何用后期的知识以尽量简化代码。我主要从以下几个角度入手：

①利用常数组MonthlengthIn2024数组储存每个月的长度，尽量避免了较多的if语句。

②利用常字符串储存”Monday”等单词，代替No.的输出方式。

①在本代码中得到了体现，②会在后续的代码中得以体现。

# 实验二

## 实验任务

Calendar2.打印月历

[实验目标]

-学习使用循环、条件等控制语句、算术表达式，

完成简单的功能

[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2021年1月1日周五)

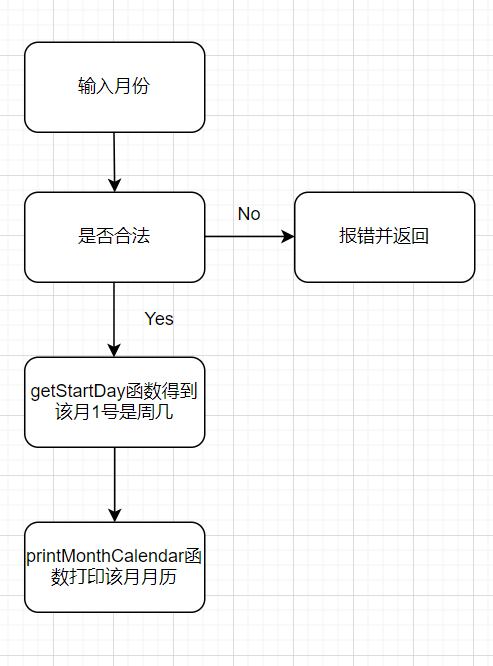
约定每个星期从周一开始

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

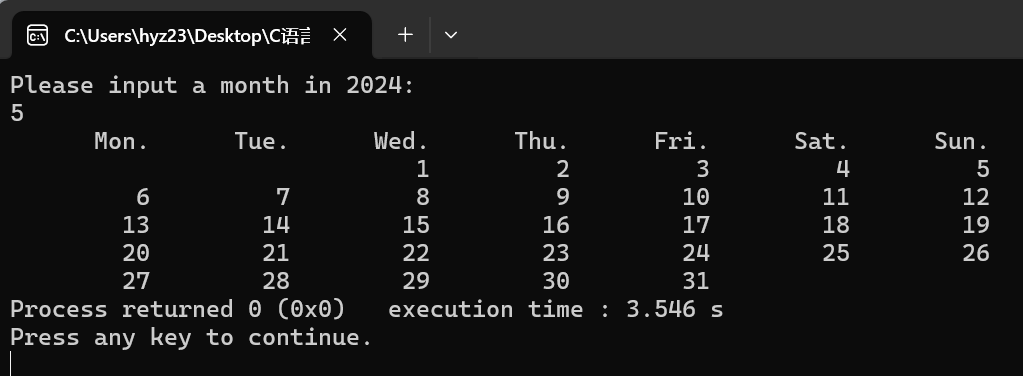
前一个实验计算);利用循环控制语句打印该月的每一天，如果碰到周日要换行

## 实验步骤

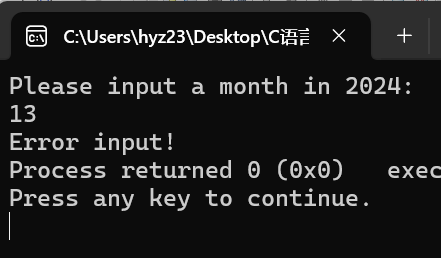
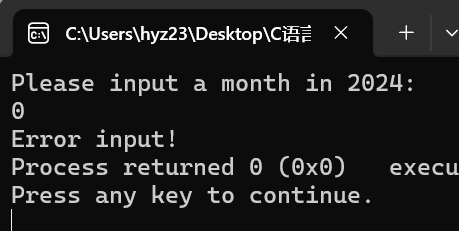


## 代码测试

### 测试代码能否正常打印月历



### 测试代码能否正常报错



## **实验结论**

代码能准确完成预期的所有功能！

## **实验总结**

本代码沿用了上个任务中的常数组MonthLengthIn2024，利用函数的思想简化了代码的结构，使得main函数更加简洁。

# 实验三

## **实验任务**

Calendar03.打印周历

[实验目标]

学习使用循环、条件等控制语句、算术表达式，完成简单的功能

[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日周四)

约定每个星期从周一开始

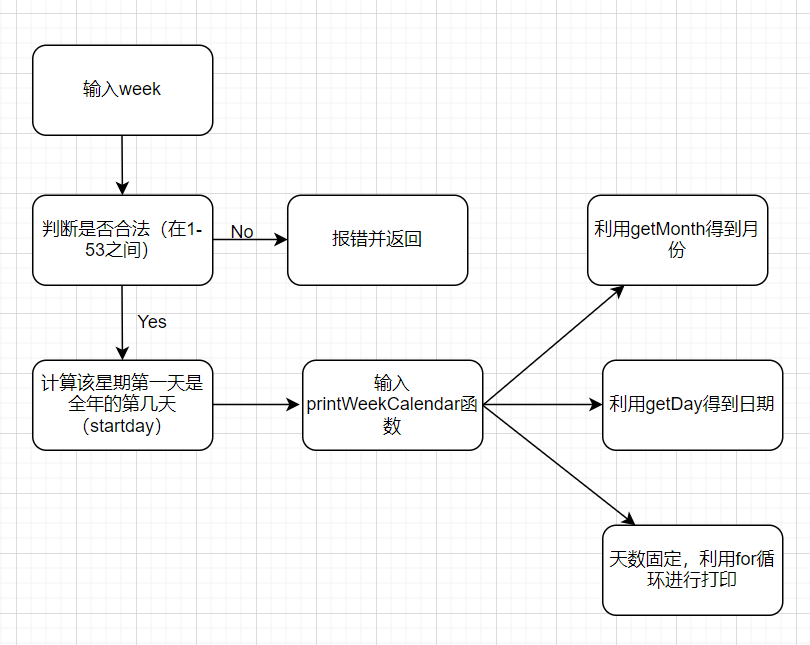
约定周历的每列的宽度为10个字符，可以在printf语句中用%5s打印空格，用%2d打印数字来定位

第1周和第53周中，仅打印2021年的日期

【提示】先求出该周的起始日、结束日是全年的第几天，以全年第几天为循环

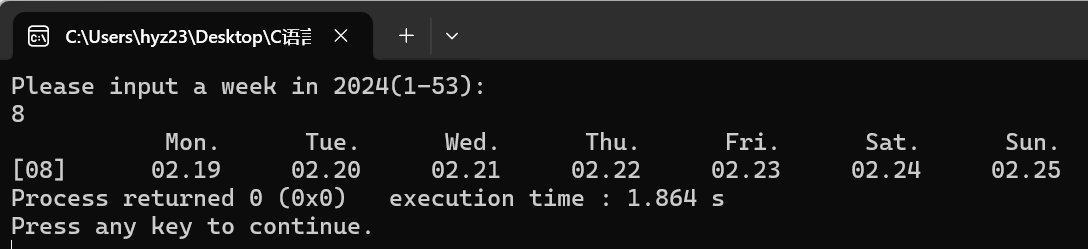
变量打印周历，循环体中判断所打印日期是几月几号

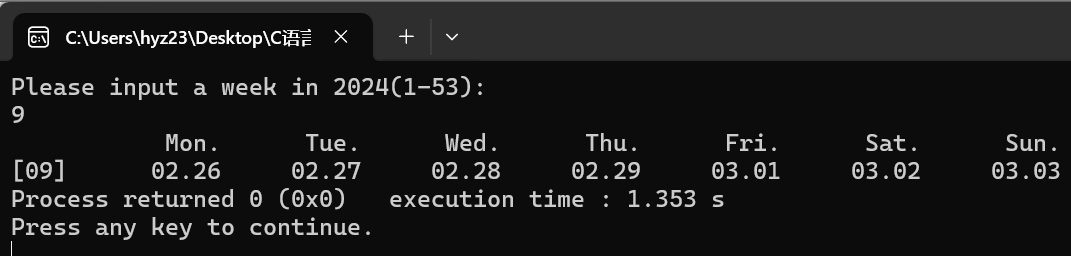
## **实验步骤**



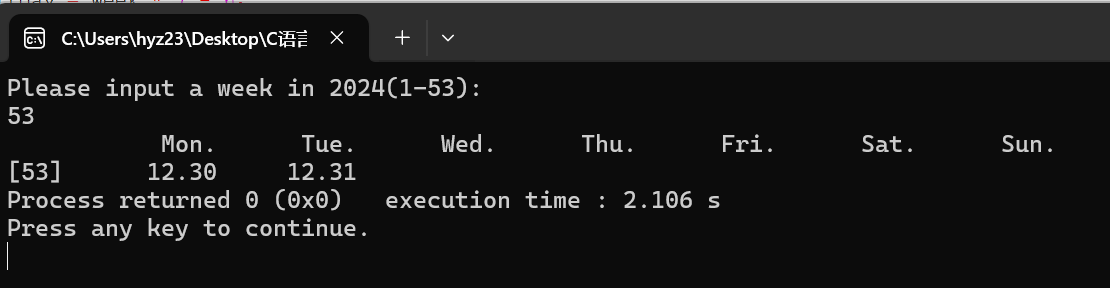
## **代码测试**

### 测试代码能否打印年中某一周的周历





### 测试代码能否打印年末的周历，并省去下一年的部分



### 测试代码能否正常报错

### 

## **实验结论**

代码准确的实现了预期的功能！

## **实验反思**

这个题带给了我极大的启发，并在后续的日历系列任务中得以应用。本题没有采用常规的方法，根据第一天的日期推断出第二天的日期，从而避免了复杂的跨月份讨论，而是利用一个简单的数据——一年的第几天，进行递推，再利用getMonth与getDay这两个函数进行转化，极大的简化了代码。

# 实验四

## **实验任务**

Calendar05.春季校历函数版（2024春）

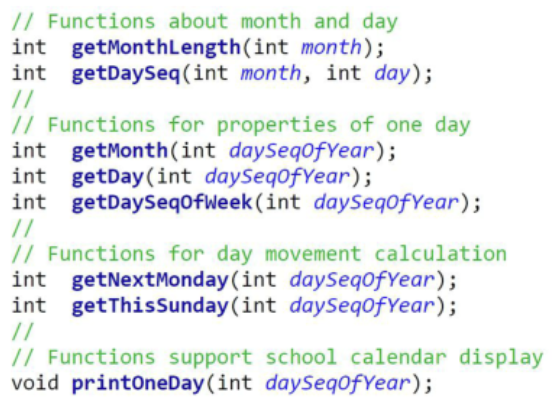
[实验目标]

学习使用函数改造已有程序，体会“基于语句的编程”与“基于函数的编程”的差异，感受模块化的优势。

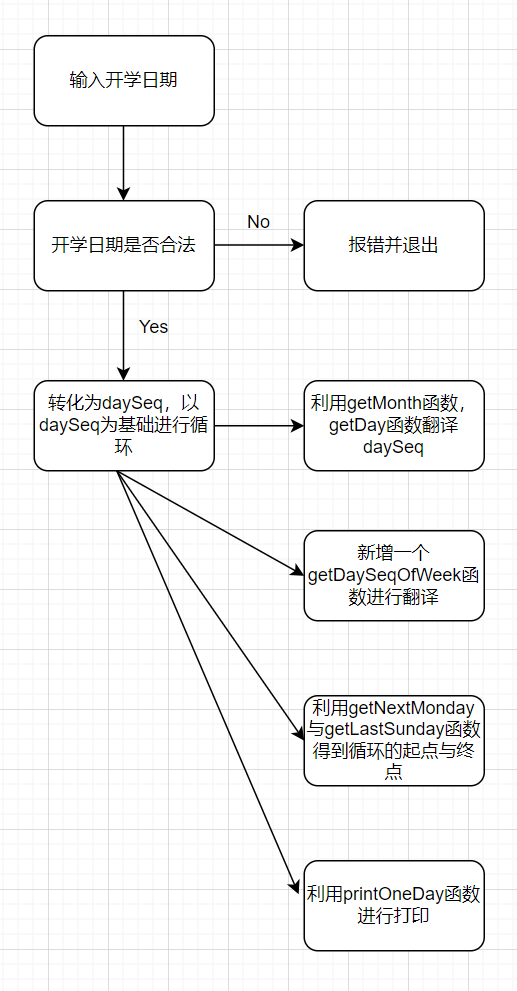
[实验任务]

用函数改写打印华中科技大学校历的程序，打印2021年春季学期的校历

要求开发下列函数:

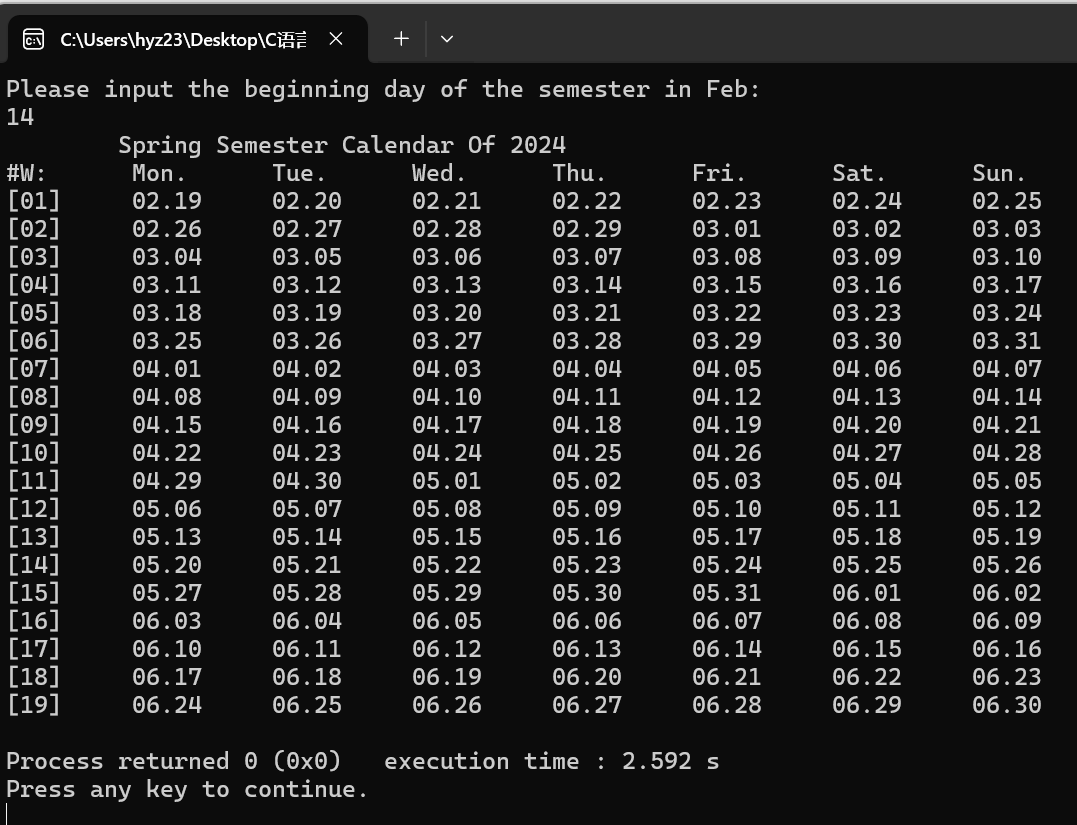


## **实验思路**

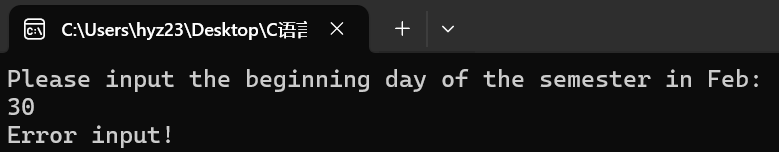


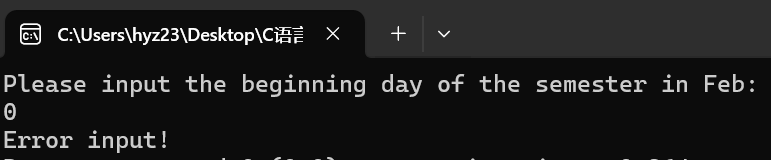
## **代码测试**

### 测试代码能否打印2024年春季学期的校历



### 测试代码能否正常报错





## **实验结论**

代码准确的实现了预期的功能！

## **实验总结**

该任务继续沿用了上一实验的“转化—翻译”思想，进一步体现了模块化的优势。主要是引入了printOneDay函数，该函数可以单独完成一定的功能，简洁且好用。在后续的实验中还可以对这个函数进行升级修改，实现更复杂的功能。

# 实验五

## **实验任务**

Calendar06.秋季校历函数版(2024秋)

[实验目标]

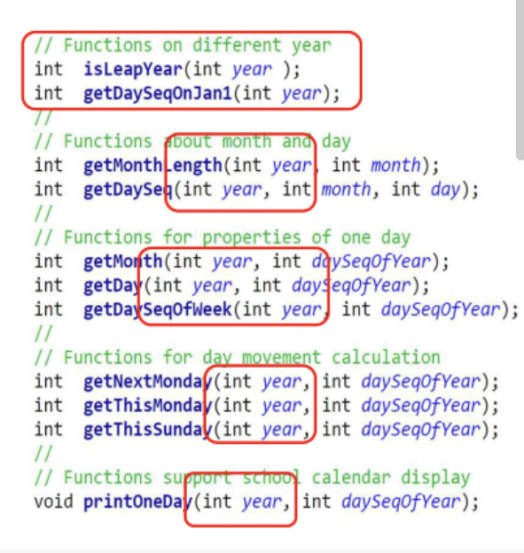
巩固函数的使用，对已有函数进行功能升级，支持跨年的打印功能，体会代码模块化给代码维护带来的优势。

[实验任务]

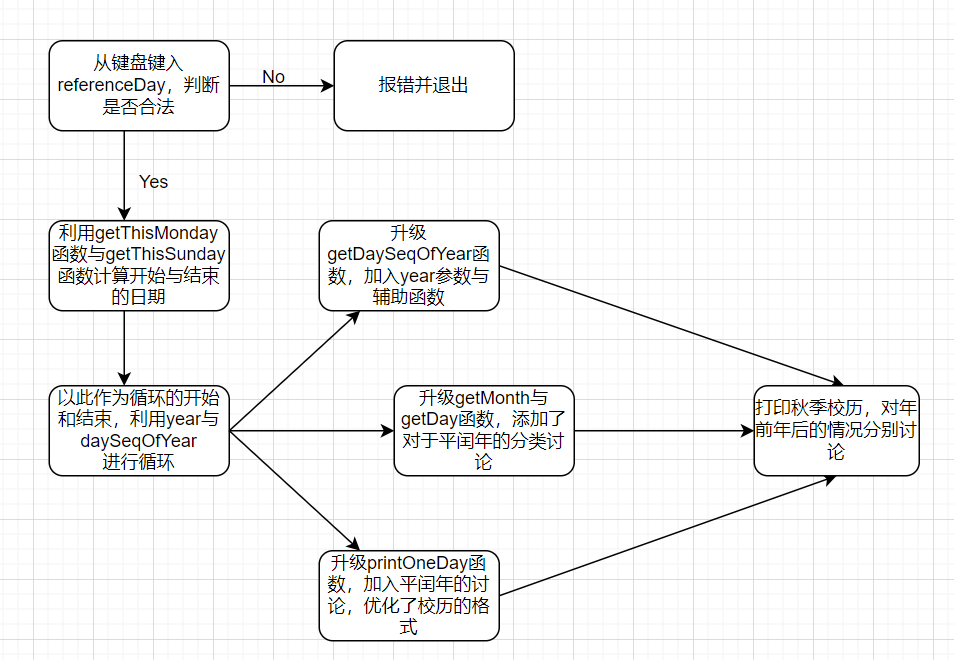
用函数改写打印华中科技大学校历的程序，打印2021年秋季学期的校历

秋季学期从9月1日所在周的周一开始，到第二年1月的某日结束:用户输入1月份的某日作为参考日，取该日所在周周日作为学期的结束。

更新相关函数，支持不同的年份的日期计算。相关函数需要增加一个参数int year。

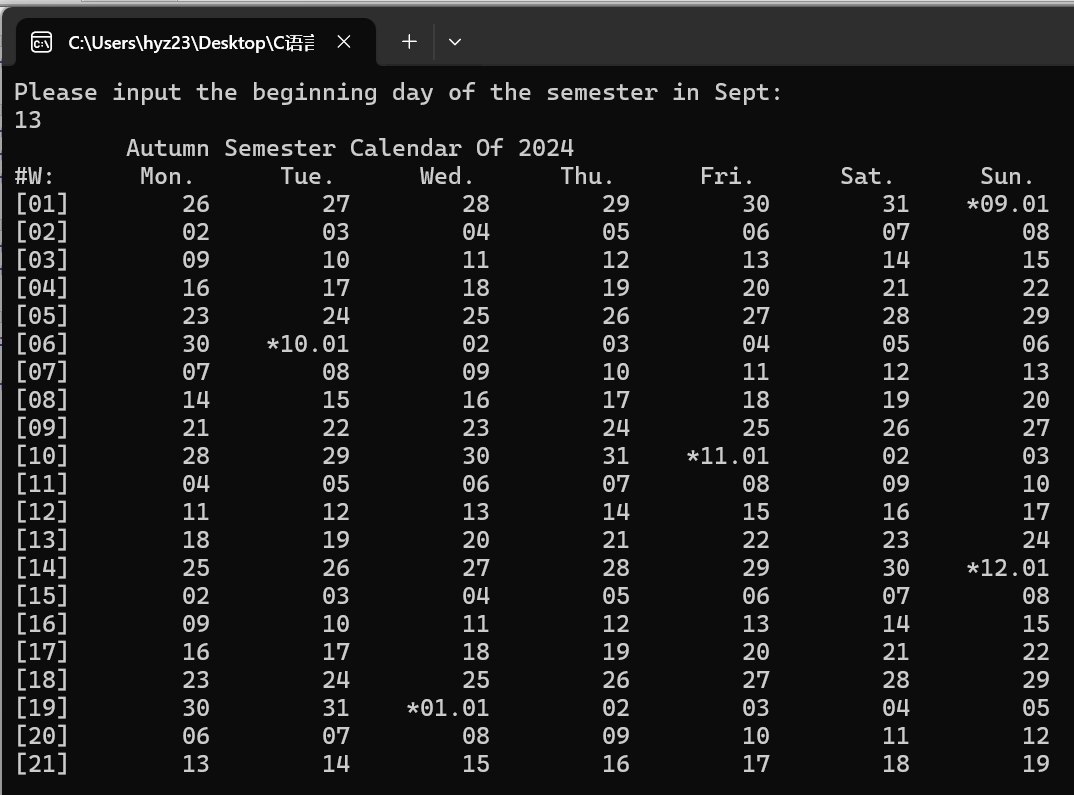


## **实验思路**

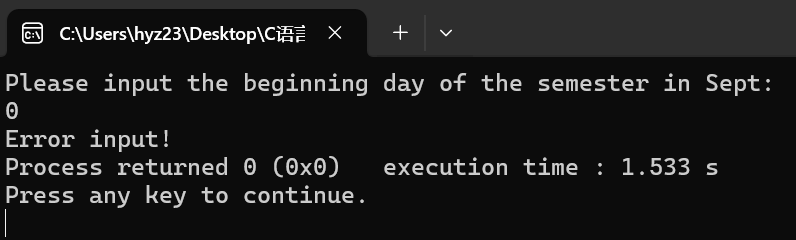


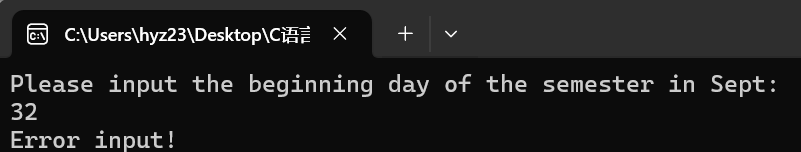
## **代码测试**

### 测试代码能否正常打印2024年的秋季校历



### 测试代码能否正常报错





## **实验结论**

代码完整准确地完成了预期的功能！

## **实验总结**

本实验的突破之处在于引入了year参数，因此所有的函数都迎来了相应的升级，常数组也变为了两个：MonthLengthInLeapYear 和MonthLengthInNormalYear。本实验既承接了上面实验中基本函数的运用，又拓展了日历的研究范围，实现了年份的跨越，为后续实验打下了基础。

# 实验六

## **实验任务**

Calendar07.打印全学年校历(函数版)

[实验目标]

巩固函数的使用，对现有函数进行功能升级(支持全学年校历打印)，实现通用功能函数printOneWeek()，体会“面向过程编程”的设计方法

[实验任务]

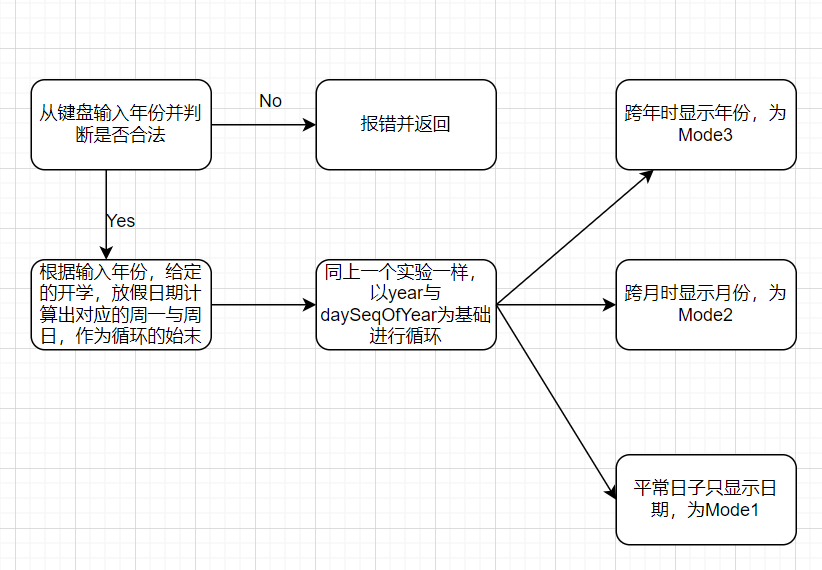
用函数改写打印华中科技大学校历的程序，打印指定学年度的第一学期和第二学期的校历

约定秋季学期从当年9月4日所在周的周一开始，到第二年1月20日之前一周结束;约定春季学期从次年2月15日之后一周开始，7月第一周结束。

要求设计一个打印某周周历的函数，支持不同学期校历的打印需求;改进

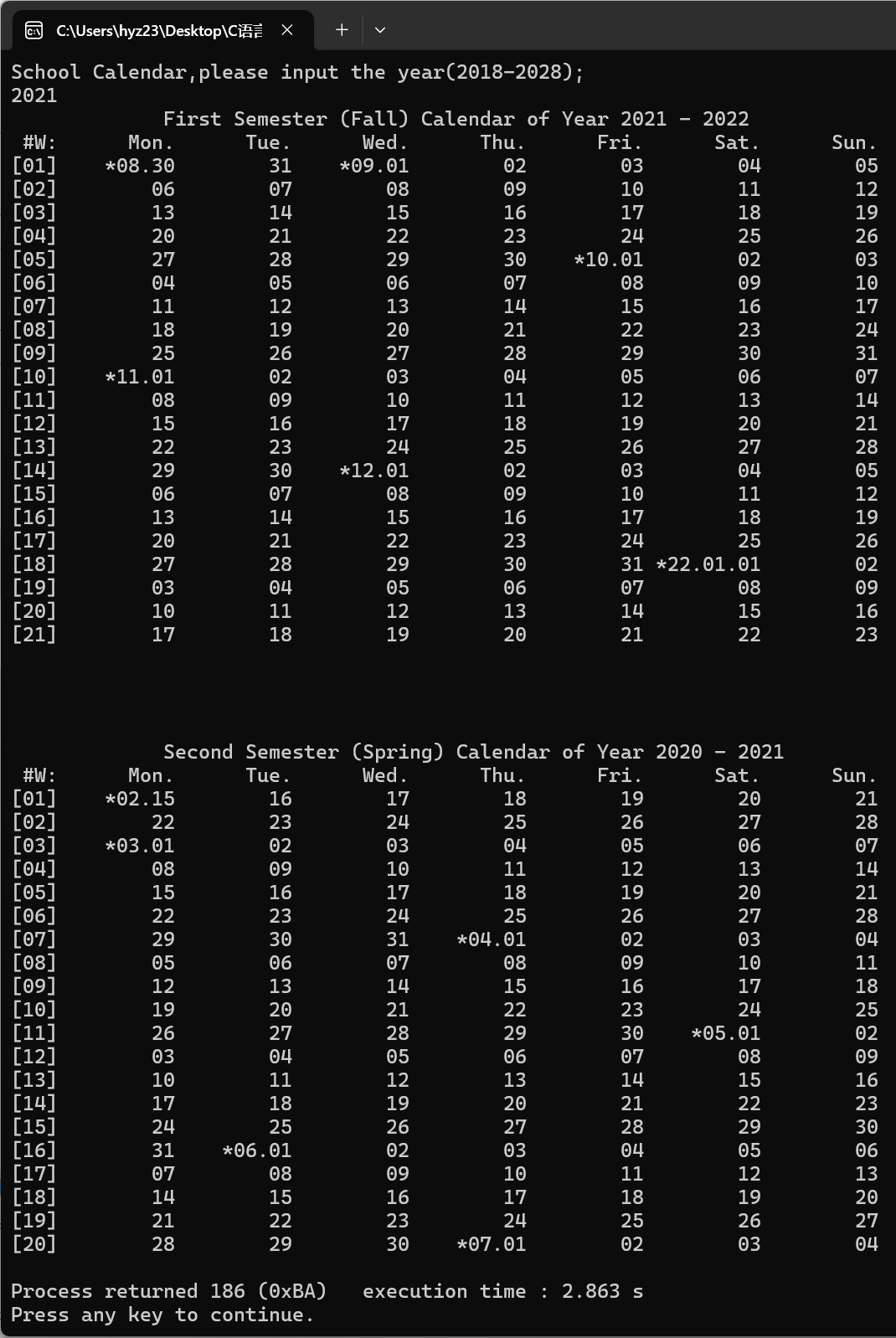
打印某日的函数，支持在校历首日和跨年日显示年月日信息。

## **实验思路**

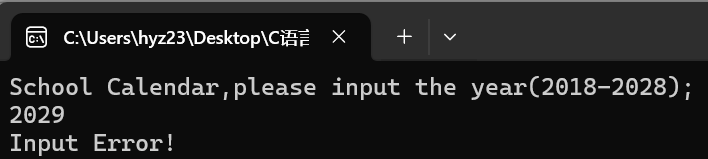


## **代码测试**

### 测试代码能否正常打印某一年的校历



### 测试代码能否正常报错



## **实验结论**

代码成功且准确地实现了待定的功能！

## **实验反思**

本实验主要是将上面两个实验融合，同时改进了printOneDay函数，使之能够实现多种模式的打印。在后续的实验中将得到进一步体现。

# 实验七

## **实验任务**

Calendar08.不同年份秋季校历首周对比

[实验目标]

-学习数组的使用，通过对现有的多变量程序进行升级，体会数组这种数据结构的优势

[实验任务]

用数组改写打印华中科技大学校历的程序，具备打印校历的能力，比较多个年份的校历秋校

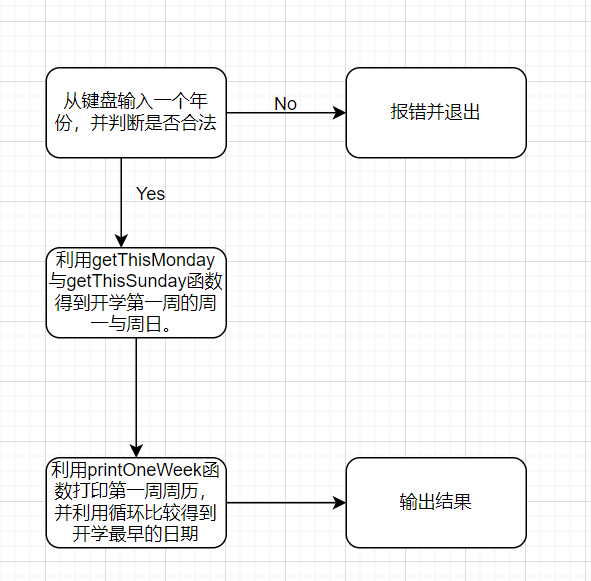
历的首日，寻找开学最早的那一年

约定秋季学期从当年9月4日所在周的周一开始，到第二年1月20日之前一周结束;约定学期从次年2月15日之后一周开始，7月第一周结束。

用两个只读的全局的一维数组记录不同月份的天数长度

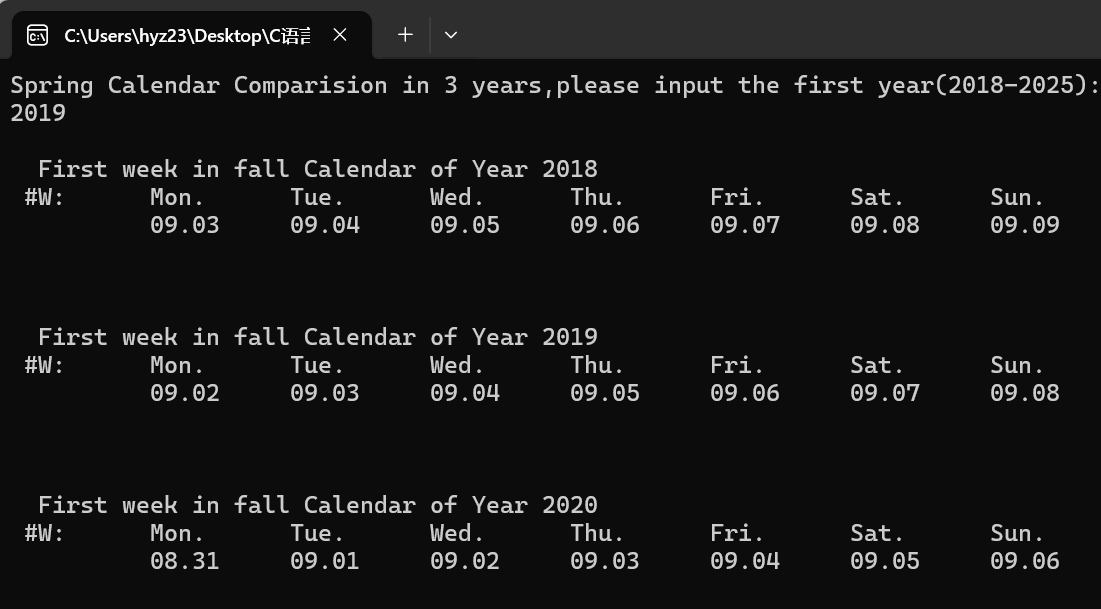
基于数组改进现有的日历程序，改进计算日期的函数

## **实验思路**

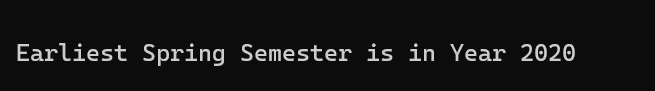


## **代码测试**

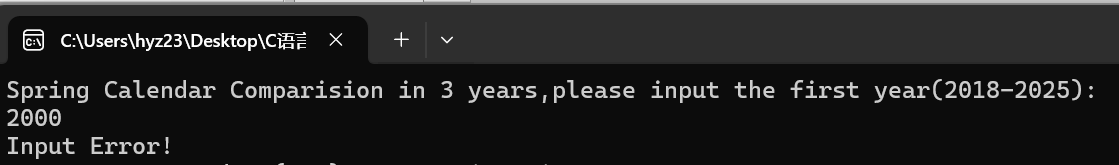
### 测试代码能否正常打印三年秋季学期的开学周



### 测试代码能否正常比较得出开学最早的一年



### 测试代码能否正常报错



## **实验结论**

代码正常且准确地实现了要求的全部功能！

## **实验总结**

该实验的主要思想是引入了月份数组，这也是我在前面的实验中一直应用的思想。此外，该实验对printOneDay函数进行了升级，在getThisMonday，getThisSunday等函数的辅助下得到了printOneWeek函数。

# 实验八

## **实验任务**

Calendar9.寻找生日(多维数组)

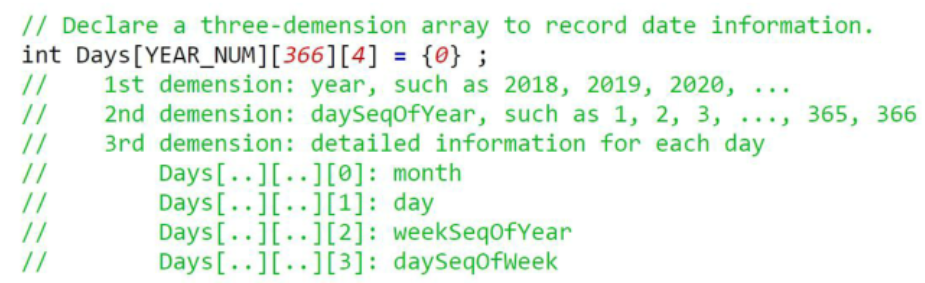
[实验目标]

学习多维数组的使用。利用多维数组事先存储所需处理的数据，体会“过程中生成数据”与“生成数据再操作”两种编程思路的差异。

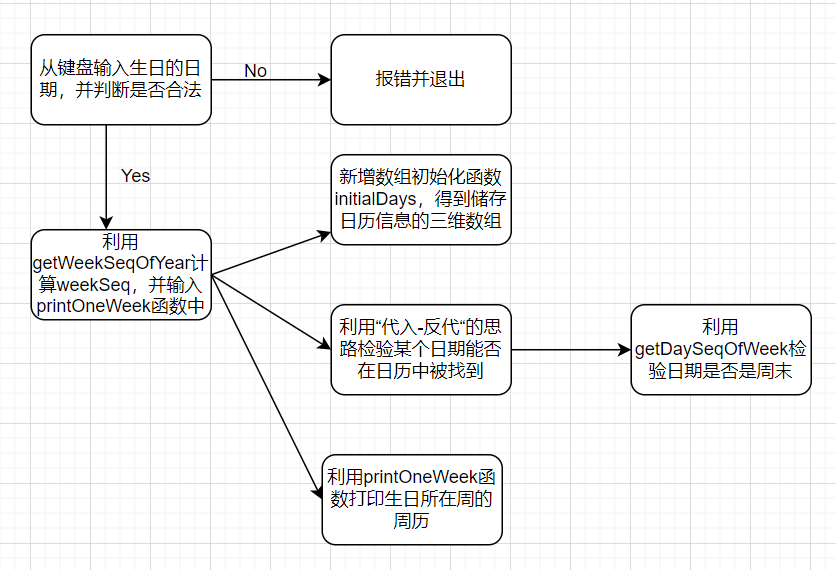
[实验任务]

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。

三维数组的定义如下：

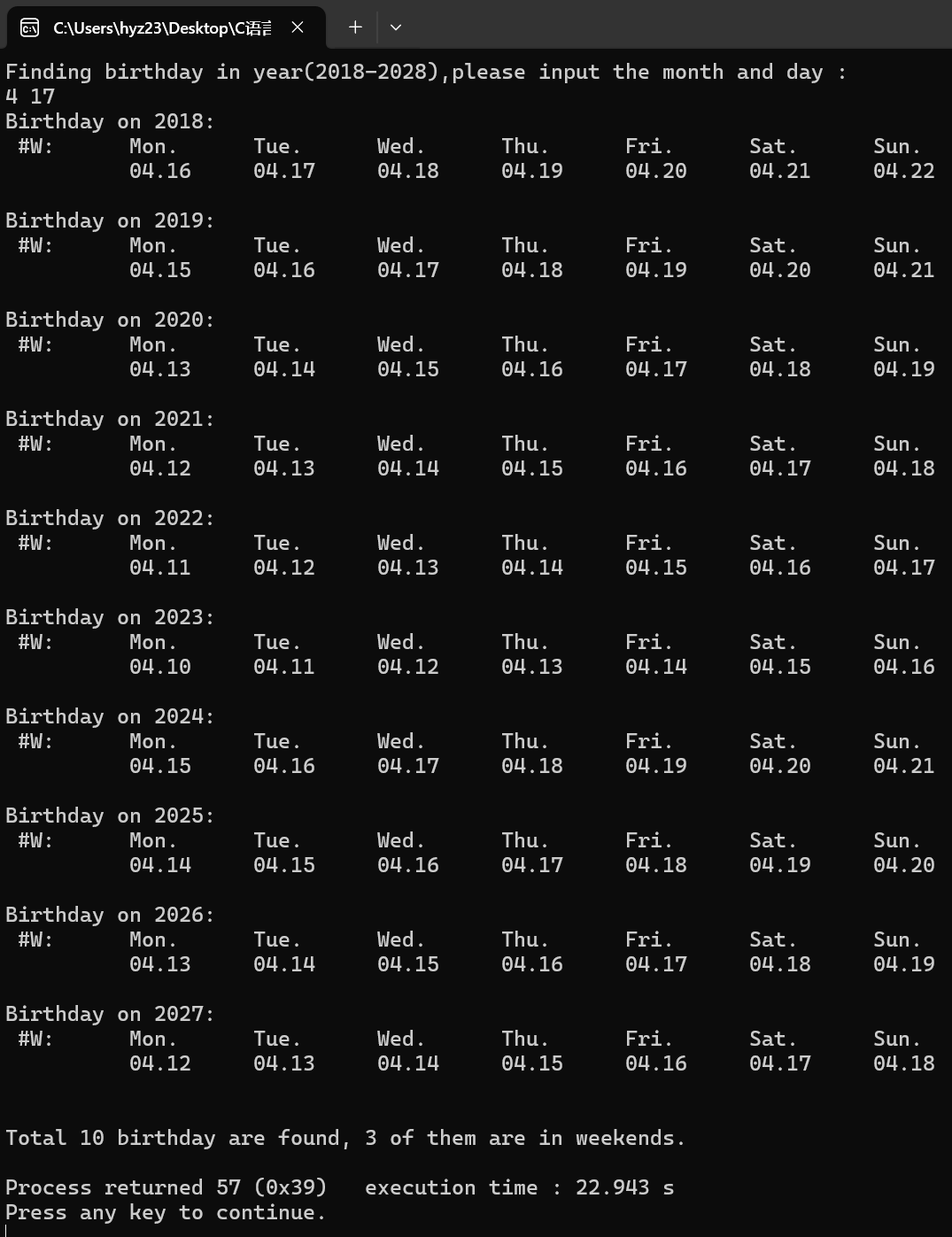


## **实验思路**

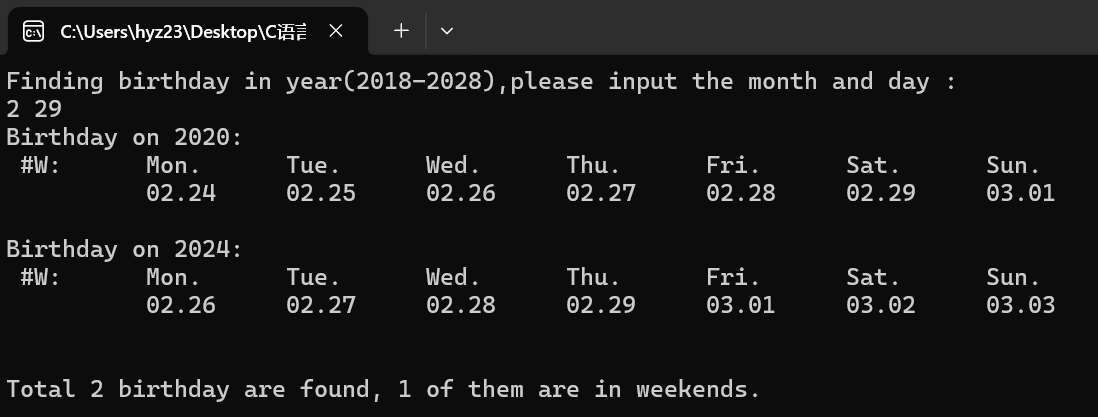


## **代码测试**

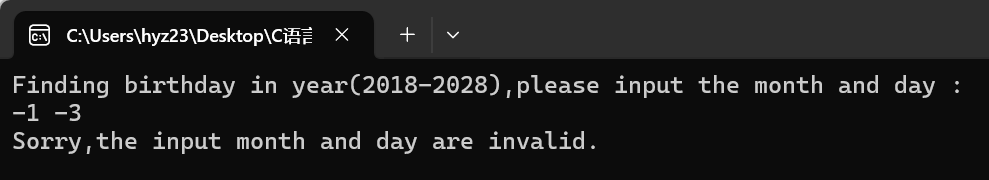
### 测试代码能否打印生日所在周周历



### 测试代码能否应对2.29这样的特殊日期



### 测试代码能否正常报错



## **实验结论**

代码完整且准确地实现了预期的功能！

## **实验总结**

该代码是对跨年份日历的进一步运用，同时引入了日期查找的功能。除此之外，本实验运用多维数组作为数据储存的手段，虽然这种方法不如结构体便捷，但是也展现出了一种全新的思路：不再是在main函数里调用getDay，getMonth等函数，将其返回值作为参数，而是在运算开始前先对数组进行初始化，再引用数组中的元素，使代码结构更加清晰。

# 实验九

## **实验任务**

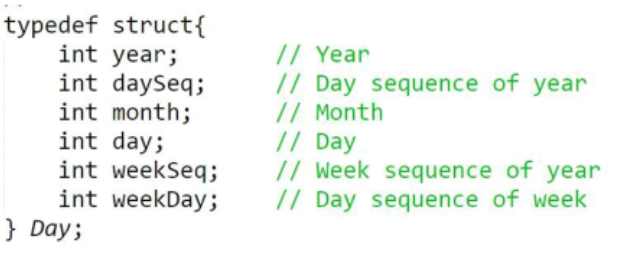
Calendar10.寻找生日(结构体)

[实验目标]

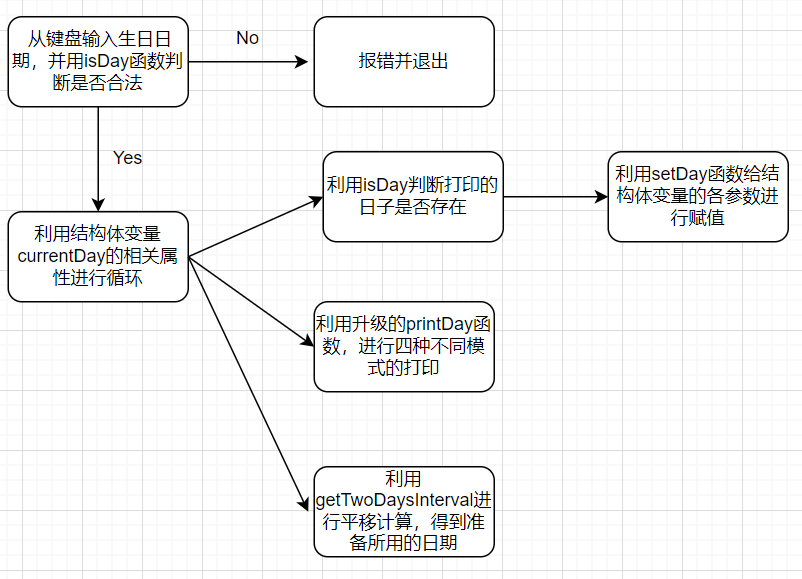
学习结构体的使用。将日期作为一种新的数据类型，对其进行“赋值”“偏移天数”等特殊操作，体会“面向过程的编程”与“面向数据类型的编程”在设计思路上的差异

[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:

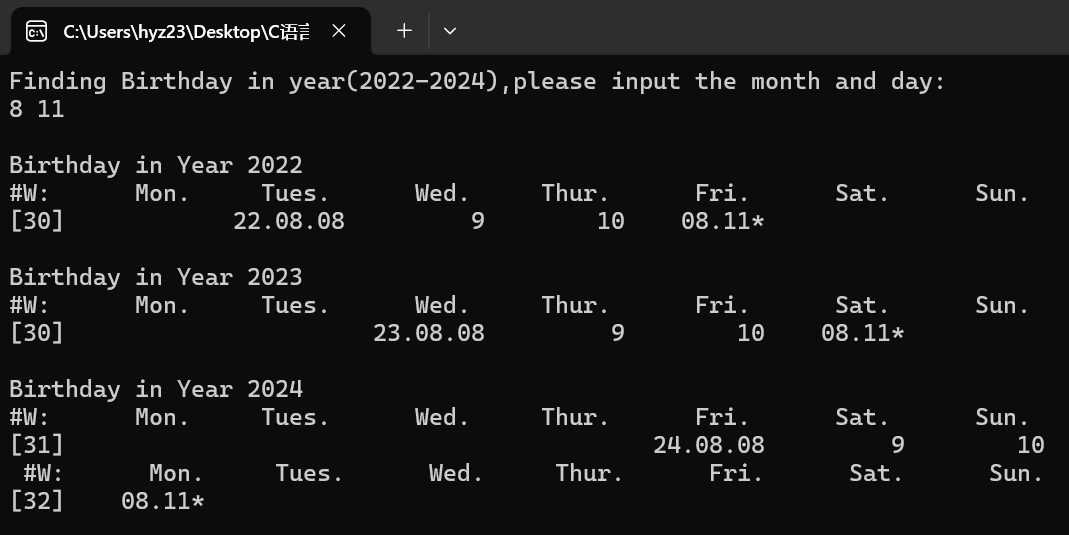


## **实验思路**

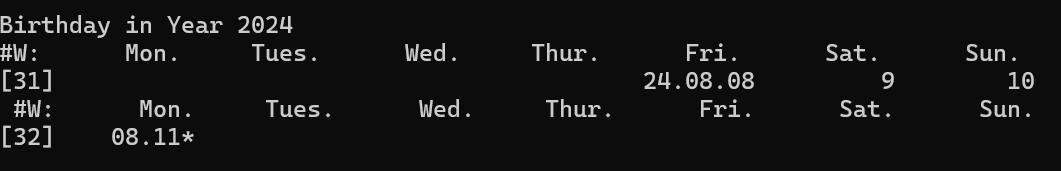


## **代码测试**

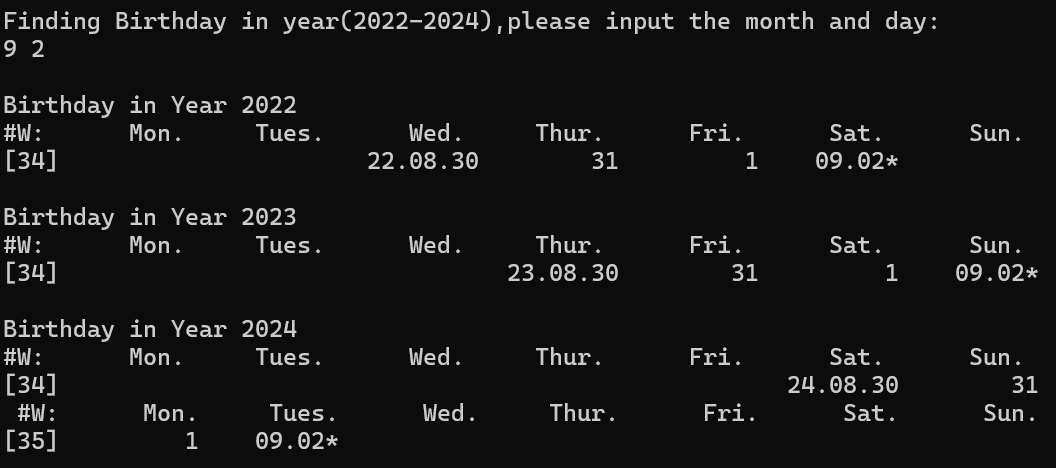
### 测试代码能否正常寻找生日



### 测试代码能否实现跨周打印



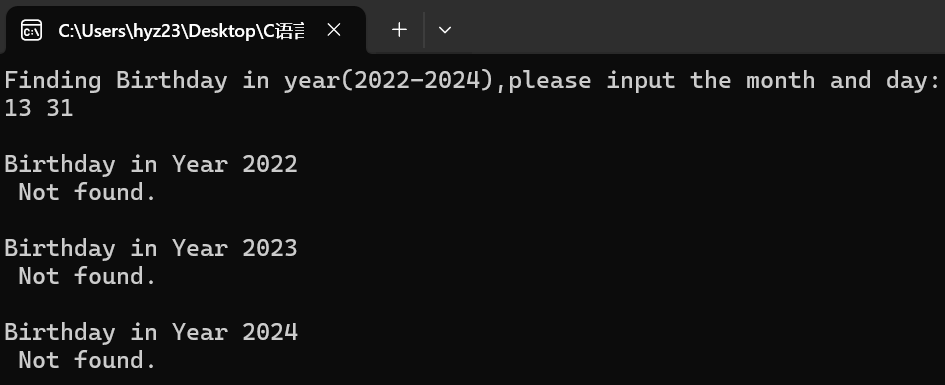
### 测试代码能否实现跨月打印

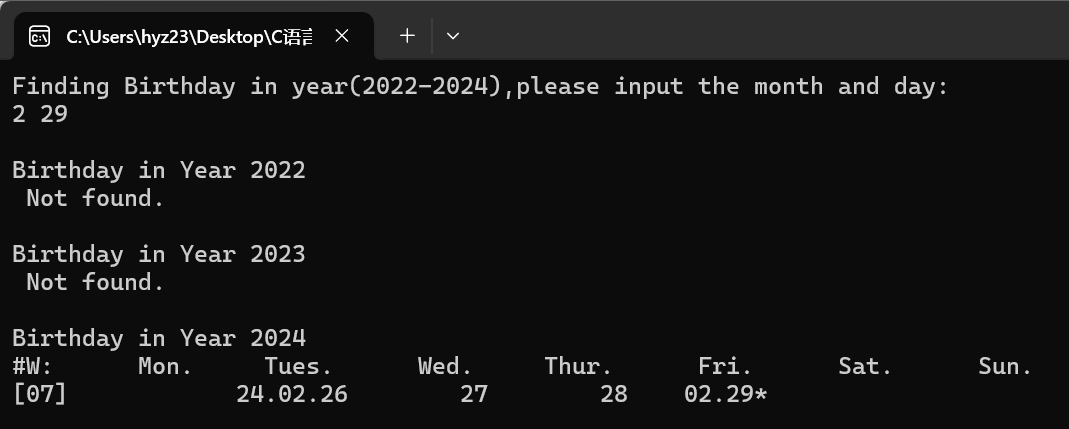


### 测试代码能否实现跨年打印



### 测试代码能否正常报错,以及应对2.29这样的特殊日期





## **实验结论**

该代码完整且准确地实现了预期地所有功能！

## **实验反思**

该代码是对结构体和前面实验中日历系列的有关函数的结合应用，体现了结构体作为一种信息储存手段的强大与便捷，同时该代码中再次升级了printOneDay函数，使之具有四种不同的打印模式。在这之外，代码还引入了getTwoDaysInterval函数，用以进行日期的平移，引入了isDay函数，用来判断日期是否在日历中被找到。

# 本课程学习总结

第一次翻开C语言的课本，映入眼帘的就是几行程序：

#include <stdio.h>

int main ( void )

{

printf(“Welcome to C!\n”);

}

作为一个初学者，我几乎是看不懂里面的任何一个符号。“好好的print，为什么要写成printf呢？”我一边心里泛着嘀咕，一边翻到了书的下一面，里面详细地进行了介绍：“#include表示预处理命令... \\是注释...main是一个函数...”诸如此类。C语言这门课程为我开启了一扇我从未向内窥视过的大门，如此多的新鲜概念向我涌来，令我一时无法接受。

不过好在，在接下来的几天内，我反复地阅读书上的内容。虽然我看不懂上面晦涩的定义，但是好在这本教材有丰富的例子，能让我从完全不懂到稍微懂一些。看完例子，再回头看看书上的定义，我似乎就能get到知识点了。

紧接着来的是第一次上课。老师讲课的速度很快，在那个我连scanf都不会用的时候，老师便布置了第一个任务——计算一个日期是星期几（也就是本报告中的实验一）。那是一次令我非常难忘的编程经历。我硬着头皮，在国庆假期期间自学了if，while等语句。回到学校，我开始思考这个任务：如果我能计算出一个日期是一年中的第几天，我应该就能计算出它是星期几了。于是我将各式各样的while与if拼接起来，试图凑出来结果。但我并没有一个明确的方向，只是想到哪儿写到哪儿。最终，我在复杂的变量命名：counter1，counter2...中迷失了自我，直到上课时我也没能编出能运行的程序。来到教室，我打算看看同学们的作品，看看他们是怎么样写的。结果答案出乎了我的意料——

if( month == 1) days += 31;

if( month == 2) days += (31 + 29);

...

if( month == 12) days += (366 - 31);

没有复杂的分类讨论——只有一排干净的if，清晰地向代码的阅读者展示着他的逻辑。我震惊于这样暴力的方法，但转念一想——这不正是计算机的魅力所在吗？在此后的学习中，老师所写的代码带给我最大的感受就是“清晰”。有命名简洁易懂的函数，有清晰的注释和整洁的排版，还有恰到好处巧妙的逻辑，这教给我了最重要的一课——编程最重要的，是先捋清楚自己的逻辑，再是将自己的逻辑用代码表达出来。

但C语言带给我的挑战远不止于此。C语言中不仅有全新的概念，还有全新的思想。数组倒比较好理解——就是数列嘛。新的思想主要体现在指针——地址与值的对应关系。初学指针，我总是不能理解地址和值的关系，更不用说还有指向函数的指针的数组，套娃又套娃的复杂概念。现在想想，指针之所以存在，是因为计算机储存数据的特点决定了数据有两种传递方式——按值传递与按引用传递，因而决定了一个值的存在必须依托于一个地址，就有了这后面的种种。

于我而言，C语言的学习是艰难的。还记得因为看不懂符号而在网上到处查，发现越查越混乱的日子；还记得一整个下午泡在教室里敲大数相加，令人身心俱疲的感觉；还记得晚上在宿舍里写头歌，发现不通过的测试集全部是被锁起来的那个时的破防。但正因如此，取得成果的感觉才是喜悦的。还记得用一个星期写的日历结构体成功跑起来，打印出和电脑日历上一模一样的结果时的兴奋，还记得经历无数遍debug后又一次按下F9，看到长实数相加的程序打印出正确的计算结果时的喜悦。我并非什么编程高手，但是看到自己从完全看不懂最基础的代码，到自己写出了小几百行的程序并成功运行的过程，怎能让人不欢喜雀跃呢？

回顾这半年学习C语言的经历，有痛苦，有喜悦，这二者共同指向了成长。我学到的不仅仅是如何编写C程序，更是对编程有了全新的认识，为其他编程语言的学习打下了基础。可以说是C语言的学习将我带进了信息技术的门槛，让我能在电子信息这条路上走向更远的远方。

# 附录

一，Calendar1.计算星期几

#include <stdio.h>

#include <stdlib.h>

const int MonthlengthIn2024[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

int main( void )

{

int day;

int month;

printf("Please input a date in 2024:\n");

scanf("%d %d",&month,&day);

if( month < 1 || month > 12 )

{

printf("Error input!");

return 0;

}

else if( day < 1 || day > MonthlengthIn2024[month])

{

printf("Error input!");

return 0;

}

else

{

int days = day;

int i;

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

{

days += MonthlengthIn2024[i];

}

int daySeq = ( days - 1 ) % 7 + 1;

printf("2024-%d-%d is No.%d day in that week.",month,day,daySeq);

return 0;

}

}

二，Calendar2-打印月历

#include <stdio.h>

#include <stdlib.h>

const int MonthLengthIn2024[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

int getStartDay( int month );

void printMonthCalendar( int month , int startday );

int main( void )

{

int month;

printf("Please input a month in 2024:\n");

scanf("%d",&month);

if( month < 1 || month > 12 )

{

printf("Error input!");

return 0;

}

else

{

int startday = getStartDay( month );

printMonthCalendar( month, startday );

}

return 0;

}

int getStartDay( int month )

{

int days = 1;

int i;

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

{

days += MonthLengthIn2024[i];

}

return ( days - 1 ) % 7 + 1;

}

void printMonthCalendar( int month, int startday )

{

int i;

printf("%10s%10s%10s%10s%10s%10s%10s\n","Mon.","Tue.","Wed.","Thu.","Fri.","Sat.","Sun.");

for( i = 1 ; i < startday + MonthLengthIn2024[month] ; i ++ )

{

if( i < startday )

{

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

}

else

{

printf("%10d",i - startday + 1 );

if( i % 7 == 0 )

{

printf("\n");

}

}

}

}

三，Calendar3-打印周历

#include <stdio.h>

#include <stdlib.h>

const int MonthLengthIn2024[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

int getMonth( int days );

int getDay( int days );

void printWeekCalendar( int startday, int week );

int main()

{

int week;

printf("Please input a week in 2024(1-53):\n");

scanf("%d",&week);

if( week < 1 || week > 53 )

{

printf("Error input!");

return 0;

}

else

{

int startday = week \* 7 - 6;

printWeekCalendar( startday, week );

return 0;

}

}

int getMonth( int days )

{

int month;

for( month = 0 ; days > MonthLengthIn2024[month] ; month ++)

{

days -= MonthLengthIn2024[month];

}

return month;

}

int getDay( int days )

{

int month;

for( month = 0 ; days > MonthLengthIn2024[month] ; month ++)

{

days -= MonthLengthIn2024[month];

}

return days;

}

void printWeekCalendar( int startday, int week )

{

int i;

int currentDay = getDay( startday );

int currentMonth = getMonth( startday );

printf("%5s%10s%10s%10s%10s%10s%10s%10s\n","","Mon.","Tue.","Wed.","Thu.","Fri.","Sat.","Sun.");

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

for( i = 1 ; i <= 7 ; i ++ )

{

if( currentDay <= MonthLengthIn2024[currentMonth] && currentMonth <= 12 )

{

printf("%5s","");

printf("%02d.%02d",currentMonth,currentDay);

currentDay = getDay( startday + i );

currentMonth = getMonth( startday + i );

}

else

{

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

}

}

return;

}

四，Calendar5-春季校历函数版

#include <stdio.h>

#include <stdlib.h>

const int MonthLengthIn2024[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

//Functions about month and day

int getDaySeq( int month, int day );

//Functions for properties of one day

int getMonth( int daySeqOfYear );

int getDay( int daySeqOfYear );

int getDaySeqOfWeek( int daySeqOfYear );

//Functions for day movement calculation

int getNextMonday( int daySeqOfYear );

int getThisSunday( int daySeqOfYear );

//Functions support school calendar display

void printOneDay( int daySeqOfYear );

int main()

{

int referenceDay;

printf("Please input the beginning day of the semester in Feb:\n");

scanf("%d",&referenceDay);

if( referenceDay > 29 || referenceDay < 1 )

{

printf("Error input!");

return 0;

}

int sStartSeqOfYear = getNextMonday( getDaySeq( 2,referenceDay ) );

int sEndSeqOfYear = getThisSunday( getDaySeq( 7,1 ) );

int daySeqOfYear,daySeqOfWeek,weekSeqOfSemester;

printf(" Spring Semester Calendar Of 2024 \n");

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

for( daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0 , weekSeqOfSemester = 1;

daySeqOfYear <= sEndSeqOfYear;

daySeqOfYear ++, daySeqOfWeek ++ ,daySeqOfWeek %= 7 )

{

if( daySeqOfWeek == 0 )

{

printf("[%02d]",weekSeqOfSemester );

}

printOneDay( daySeqOfYear );

if( daySeqOfWeek == 6 )

{

printf("\n");

weekSeqOfSemester ++;

}

}

return 0;

}

int getDaySeq( int month, int day )

{

int i;

int daySeq = day;

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

{

daySeq += MonthLengthIn2024[month];

}

return daySeq;

}

int getMonth( int daySeqOfYear )

{

int month = 0;

for( ; daySeqOfYear > MonthLengthIn2024[month] ; month ++ )

{

daySeqOfYear -= MonthLengthIn2024[month];

}

return month;

}

int getDay( int daySeqOfYear )

{

int month = 0;

for( ; daySeqOfYear > MonthLengthIn2024[month] ; month ++ )

{

daySeqOfYear -= MonthLengthIn2024[month];

}

return daySeqOfYear;

}

int getDaySeqOfWeek( int daySeqOfYear )

{

return ( daySeqOfYear - 1 ) % 7 + 1;

}

int getNextMonday( int daySeqOfYear )

{

return daySeqOfYear + ( 8 - getDaySeqOfWeek( daySeqOfYear ));

}

int getThisSunday( int daySeqOfYear )

{

return daySeqOfYear - ( getDaySeqOfWeek( daySeqOfYear ));

}

void printOneDay( int daySeqOfYear )

{

int day = getDay( daySeqOfYear );

int month = getMonth( daySeqOfYear );

printf("%5s","");

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

return;

}

五，Calendar6-秋季校历函数版

#include <stdio.h>

#include <stdlib.h>

#define PRINT\_MODE\_FULL 2

#define PRINT\_MODE\_BRIEF 1

const int MonthLengthInNormalYear[13] = {0,31,28,31,30,31,30,31,31,30,31,30,31};

const int MonthLengthInLeapYear[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

const int currentYear = 2024;

//Functions on different year

int isLeapYear( int year );

int getDaySeqOnJan1( int year );

//

//Functions about month and day

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

//

//Functions for properties of one day

int getMonth( int year, int daySeqOfYear );

int getDay( int year, int daySeqOfYear );

int getDaySeqOfWeek( int year, int daySeqOfYear );

//

//Functions for day movement calculation

int getThisMonday( int year, int daySeqOfYear );

int getThisSunday( int year, int daySeqOfYear );

//

//Functions support school calendar display

void printOneDay( int year, int daySeqOfYear, int printMode );

int main( void )

{

int referenceDay;

printf("Please input the beginning day of the semester in Sept:\n");

scanf("%d",&referenceDay);

if( referenceDay > 29 || referenceDay < 1 )

{

printf("Error input!");

return 0;

}

int sStartSeqOfYear = getThisMonday( currentYear,getDaySeq( currentYear,9,1 ) );

int sEndSeqOfYear = getThisSunday( currentYear + 1,getDaySeq( currentYear + 1, 1,referenceDay ) );

int currentYearLength = 365 + isLeapYear( currentYear );

int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;

int printMode = 0;

printf(" Autumn Semester Calendar Of 2024 \n");

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

for( daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0 , weekSeqOfSemester = 1;

daySeqOfYear <= sEndSeqOfYear + currentYearLength;

daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7 )

{

printMode = 1;

if( daySeqOfWeek == 0 )

{

printf("[%02d]", weekSeqOfSemester );

}

if( daySeqOfYear <= currentYearLength )

{

if( getDay( currentYear, daySeqOfYear) == 1 )

{

printMode ++;

}

printOneDay( currentYear, daySeqOfYear, printMode );

}

else

{

if( getDay( currentYear, daySeqOfYear - currentYearLength) == 1 )

{

printMode ++;

}

printOneDay( currentYear + 1, daySeqOfYear - currentYearLength , printMode );

}

if( daySeqOfWeek == 6 )

{

printf("\n");

weekSeqOfSemester ++;

}

}

return 0;

}

int isLeapYear( int year )

{

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

{

return 1;

}

else

{

return 0;

}

}

int getDaySeqOnJan1( int year )

{

int daySeqOnJan1 = 1;

int currentYear = 2018;

for( ; currentYear < year ; currentYear ++ )

{

daySeqOnJan1 += ( 1 + isLeapYear( currentYear ));

}

return (daySeqOnJan1 - 1) % 7 + 1;

}

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

{

int i;

int daySeq = day;

if( isLeapYear( year ) == 1 )

{

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

{

daySeq += MonthLengthInLeapYear[i];

}

}

else

{

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

{

daySeq += MonthLengthInNormalYear[i];

}

}

return daySeq;

}

int getMonth( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > MonthLengthInLeapYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > MonthLengthInNormalYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInNormalYear[month];

}

}

return month;

}

int getDay( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > MonthLengthInLeapYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > MonthLengthInNormalYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInNormalYear[month];

}

}

return daySeqOfYear;

}

int getDaySeqOfWeek( int year, int daySeqOfYear )

{

int daySeqOnJan1 = getDaySeqOnJan1( year );

return ( daySeqOfYear + daySeqOnJan1 - 2 ) % 7 + 1;

}

int getThisSunday( int year, int daySeqOfYear )

{

int daySeqOfWeek = getDaySeqOfWeek( year, daySeqOfYear );

return daySeqOfYear + ( 7 - daySeqOfWeek );

}

int getThisMonday( int year, int daySeqOfYear )

{

int daySeqOfWeek = getDaySeqOfWeek( year, daySeqOfYear );

return daySeqOfYear - daySeqOfWeek + 1;

}

void printOneDay( int year, int daySeqOfYear, int printMode )

{

int day = getDay( year, daySeqOfYear );

int month = getMonth( year, daySeqOfYear );

if( printMode == PRINT\_MODE\_FULL )

{

printf("%4s","");

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

}

else if( printMode == PRINT\_MODE\_BRIEF )

{

printf("%8s","");

printf("%02d",day);

}

return;

}

六，Calendar7-某年校历

#include <stdio.h>

#include <stdlib.h>

#define PRINT\_MODE\_YEAR 3

#define PRINT\_MODE\_MONTH 2

#define PRINT\_MODE\_BRIEF 1

#define YEAR\_MIN 2018

#define YEAR\_MAX 2028

const int MonthLengthInNormalYear[13] = {0,31,28,31,30,31,30,31,31,30,31,30,31};

const int MonthLengthInLeapYear[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

//Functions on different year

int isLeapYear( int year );

int getDaySeqOnJan1( int year );

//

//Functions about month and day

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

//

//Functions for properties of one day

int getMonth( int year, int daySeqOfYear );

int getDay( int year, int daySeqOfYear );

int getDaySeqOfWeek( int year, int daySeqOfYear );

//

//Functions for day movement calculation

int getThisMonday( int year, int daySeqOfYear );

int getThisSunday( int year, int daySeqOfYear );

//

//Functions support school calendar display

void printOneDay( int year, int daySeqOfYear, int printMode );

void printOneWeek( int year, int weekSeqOfYear, int weekSeqShow );

int main( void )

{

int inputYear;

printf("School Calendar,please input the year(%d-%d);\n",YEAR\_MIN,YEAR\_MAX);

scanf("%d", &inputYear );

if( inputYear < YEAR\_MIN || inputYear > YEAR\_MAX )

{

printf("Input Error!");

return 0;

}

int currentYear = inputYear;

printf("%13s%s%d%s%d\n"," ","First Semester (Fall) Calendar of Year ", inputYear," - ",inputYear + 1 );

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

int sStartSeqOfYear = getThisMonday( currentYear,getDaySeq( currentYear,9,4 ) );

int sEndSeqOfYear = getThisSunday( currentYear + 1,getDaySeq( currentYear + 1, 1,20 ) );

int currentYearLength = 365 + isLeapYear( currentYear );

int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;

int printMode = 0;

for( daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0 , weekSeqOfSemester = 1;

daySeqOfYear <= sEndSeqOfYear + currentYearLength;

daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7 )

{

printMode = 1;

if( daySeqOfYear == sStartSeqOfYear )

{

printMode = 2;

}

if( daySeqOfWeek == 0 )

{

printf("[%02d]", weekSeqOfSemester );

}

if( daySeqOfYear <= currentYearLength )

{

if( getDay( currentYear, daySeqOfYear) == 1 )

{

printMode = 2;

}

printOneDay( currentYear, daySeqOfYear, printMode );

}

else

{

if( getDay( currentYear, daySeqOfYear - currentYearLength) == 1 )

{

printMode = 2;

if( getMonth( currentYear, daySeqOfYear - currentYearLength) == 1 )

{

printMode = 3;

}

}

printOneDay( currentYear + 1, daySeqOfYear - currentYearLength , printMode );

}

if( daySeqOfWeek == 6 )

{

printf("\n");

weekSeqOfSemester ++;

}

}

printf("\n\n\n\n%13s%s%d%s%d\n"," ","Second Semester (Spring) Calendar of Year ",inputYear - 1, " - ",inputYear );

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

sStartSeqOfYear = getThisMonday( currentYear,getDaySeq( currentYear,2,15 ) );

sEndSeqOfYear = getThisSunday( currentYear,getDaySeq( currentYear, 7,1 ) );

for( daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0 , weekSeqOfSemester = 1;

daySeqOfYear <= sEndSeqOfYear;

daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7 )

{

printMode = 1;

if( daySeqOfYear == sStartSeqOfYear )

{

printMode = 2;

}

if( daySeqOfWeek == 0 )

{

printf("[%02d]", weekSeqOfSemester );

}

if( daySeqOfYear <= currentYearLength )

{

if( getDay( currentYear, daySeqOfYear) == 1 )

{

printMode = 2;

}

printOneDay( currentYear, daySeqOfYear, printMode );

}

else

{

if( getDay( currentYear, daySeqOfYear - currentYearLength) == 1 )

{

printMode = 2;

if( getMonth( currentYear, daySeqOfYear - currentYearLength) == 1 )

{

printMode = 3;

}

}

printOneDay( currentYear + 1, daySeqOfYear - currentYearLength , printMode );

}

if( daySeqOfWeek == 6 )

{

printf("\n");

weekSeqOfSemester ++;

}

}

}

int isLeapYear( int year )

{

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

{

return 1;

}

else

{

return 0;

}

}

int getDaySeqOnJan1( int year )

{

int daySeqOnJan1 = 1;

int currentYear = 2018;

for( ; currentYear < year ; currentYear ++ )

{

daySeqOnJan1 += ( 1 + isLeapYear( currentYear ));

}

return (daySeqOnJan1 - 1) % 7 + 1;

}

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

{

int i;

int daySeq = day;

if( isLeapYear( year ) == 1 )

{

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

{

daySeq += MonthLengthInLeapYear[i];

}

}

else

{

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

{

daySeq += MonthLengthInNormalYear[i];

}

}

return daySeq;

}

int getMonth( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > MonthLengthInLeapYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > MonthLengthInNormalYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInNormalYear[month];

}

}

return month;

}

int getDay( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > MonthLengthInLeapYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > MonthLengthInNormalYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInNormalYear[month];

}

}

return daySeqOfYear;

}

int getDaySeqOfWeek( int year, int daySeqOfYear )

{

int daySeqOnJan1 = getDaySeqOnJan1( year );

return ( daySeqOfYear + daySeqOnJan1 - 2 ) % 7 + 1;

}

int getThisSunday( int year, int daySeqOfYear )

{

int daySeqOfWeek = getDaySeqOfWeek( year, daySeqOfYear );

return daySeqOfYear + ( 7 - daySeqOfWeek );

}

int getThisMonday( int year, int daySeqOfYear )

{

int daySeqOfWeek = getDaySeqOfWeek( year, daySeqOfYear );

return daySeqOfYear - daySeqOfWeek + 1;

}

void printOneDay( int year, int daySeqOfYear, int printMode )

{

int day = getDay( year, daySeqOfYear );

int month = getMonth( year, daySeqOfYear );

if( printMode == PRINT\_MODE\_MONTH )

{

printf("%4s","");

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

}

else if( printMode == PRINT\_MODE\_BRIEF )

{

printf("%8s","");

printf("%02d",day);

}

else if( printMode = PRINT\_MODE\_YEAR )

{

printf(" \*%02d.%02d.%02d",year % 100 , month , day );

}

return;

}

七，Calendar8-月份数组

#include <stdio.h>

#include <stdlib.h>

#define YEAR\_NUM 3

#define YEAR\_MIN 2018

#define YEAR\_MAX 2028

const int MonthLengthInNormalYear[13] = {0,31,28,31,30,31,30,31,31,30,31,30,31};

const int MonthLengthInLeapYear[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

//Functions on different year

int isLeapYear( int year );

int getDaySeqOnJan1( int year );

//

//Functions about month and day

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

//

//Functions for properties of one day

int getMonth( int year, int daySeqOfYear );

int getDay( int year, int daySeqOfYear );

int getDaySeqOfWeek( int year, int daySeqOfYear );

//

//Functions for day movement calculation

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

int getThisMonday( int year, int daySeqOfYear );

int getThisSunday( int year, int daySeqOfYear );

//

//Functions support school calendar display

void printOneWeek( int year, int weekSeqOfYear );

void setYearArray( int array[], int YearNum, int inputYear );

int main( void )

{

int inputYear, Years[YEAR\_NUM];

printf("Spring Calendar Comparision in %d years,please input the first year(%d-%d):\n",

YEAR\_NUM,YEAR\_MIN,YEAR\_MAX - YEAR\_NUM );

scanf("%d", &inputYear );

if( inputYear < YEAR\_MIN || inputYear > YEAR\_MAX )

{

printf("Input Error!");

return 0;

}

setYearArray( Years, YEAR\_NUM, inputYear );

int i;

int startDays[YEAR\_NUM] = {0};

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

{

printf("\n%s%s%d\n"," "," First week in fall Calendar of Year ", Years[i]);

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

printOneWeek( Years[i], getWeekSeqOfYear( Years[i],9,4 ));

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

startDays[i] = getThisMonday( Years[i], getDaySeq( Years[i], 9 ,4 ));

}

int min = 0;

for( i = 1 ; i < YEAR\_NUM ; i ++ )

{

min = ( startDays[i] < startDays[min] )? i : min;

}

printf("\n%s%s%d\n"," ","Earliest Spring Semester is in Year ",Years[min]);

return 0;

}

int isLeapYear( int year )

{

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

{

return 1;

}

else

{

return 0;

}

}

int getDaySeqOnJan1( int year )

{

int daySeqOnJan1 = 1;

int currentYear = 2018;

for( ; currentYear < year ; currentYear ++ )

{

daySeqOnJan1 += ( 1 + isLeapYear( currentYear ));

}

return (daySeqOnJan1 - 1) % 7 + 1;

}

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

{

int i;

int daySeq = day;

if( isLeapYear( year ) == 1 )

{

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

{

daySeq += MonthLengthInLeapYear[i];

}

}

else

{

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

{

daySeq += MonthLengthInNormalYear[i];

}

}

return daySeq;

}

int getMonth( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > MonthLengthInLeapYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > MonthLengthInNormalYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInNormalYear[month];

}

}

return month;

}

int getDay( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > MonthLengthInLeapYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > MonthLengthInNormalYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInNormalYear[month];

}

}

return daySeqOfYear;

}

int getDaySeqOfWeek( int year, int daySeqOfYear )

{

int daySeqOnJan1 = getDaySeqOnJan1( year );

return ( daySeqOfYear + daySeqOnJan1 - 2 ) % 7 + 1;

}

int getThisSunday( int year, int daySeqOfYear )

{

int daySeqOfWeek = getDaySeqOfWeek( year, daySeqOfYear );

return daySeqOfYear + ( 7 - daySeqOfWeek );

}

int getThisMonday( int year, int daySeqOfYear )

{

int daySeqOfWeek = getDaySeqOfWeek( year, daySeqOfYear );

return daySeqOfYear - daySeqOfWeek + 1;

}

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

{

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

return daySeq / 7 + 1;

}

void setYearArray( int array[], int YearNum, int inputYear )

{

int i;

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

{

array[i] = YEAR\_MIN + i;

}

return;

}

void printOneWeek( int year, int weekSeqOfYear)

{

int daySeqOnJan1 = getDaySeqOnJan1( year );

int mondaySeq = weekSeqOfYear \* 7 - daySeqOnJan1 - 5;

int sundaySeq = mondaySeq + 6;

int currentSeq,currentDay,currentMonth;

for( currentSeq = mondaySeq ; currentSeq <= sundaySeq ; currentSeq ++ )

{

currentDay = getDay( year, currentSeq );

currentMonth = getMonth( year, currentSeq );

printf("%02d.%02d",currentMonth,currentDay );

printf("%5s","");

}

return;

}

八，Calendar9-日期数组

#include <stdio.h>

#include <stdlib.h>

#define YEAR\_NUM 10

#define YEAR\_MIN 2018

#define YEAR\_MAX 2028

const int MonthLengthInNormalYear[13] = {0,31,28,31,30,31,30,31,31,30,31,30,31};

const int MonthLengthInLeapYear[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

//Functions on different year

int isLeapYear( int year );

int getDaySeqOnJan1( int year );

//

//Functions about month and day

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

//

//Functions for properties of one day

int getMonth( int year, int daySeqOfYear );

int getDay( int year, int daySeqOfYear );

int getDaySeqOfWeek( int year, int daySeqOfYear );

//

//Functions for day movement calculation

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

int getThisMonday( int year, int daySeqOfYear );

int getThisSunday( int year, int daySeqOfYear );

//

//Functions support birthday calendar display

void printOneWeek( int year, int weekSeqOfYear );

void setYearArray( int array[], int YearNum, int inputYear );

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

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

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

// 2nd dimension: daySeqOfYear, such as 1,2,3,...365,366

// 3rd dimension: detailed information for each day

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

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

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

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

int main( void )

{

int inputMonth, inputDay, Years[YEAR\_NUM] = {0};

int Days[YEAR\_NUM][366][4];

printf("Finding birthday in year(%d-%d),please input the month and day :\n",

YEAR\_MIN, YEAR\_MAX);

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

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

{

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

return 0;

}

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

initialDays( Years, Days, YEAR\_NUM );

int totalNum = 0, weekendNum = 0;

int currentYear = YEAR\_MIN;

for( ; currentYear < YEAR\_MAX ; currentYear ++ )

{

int birthdaySeq = getDaySeq( currentYear, inputMonth, inputDay );

//printf("%d %d\n",getDay( currentYear , birthdaySeq ),inputDay );

if( getDay( currentYear , birthdaySeq ) == inputDay )

{

int weekSeq = getWeekSeqOfYear( currentYear, inputMonth, inputDay );

printf("Birthday on %d:\n", currentYear );

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

printOneWeek( currentYear, weekSeq );

totalNum ++;

if( getDaySeqOfWeek( currentYear, birthdaySeq ) > 5 )

{

weekendNum ++;

}

printf("\n\n");

}

}

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

}

int isLeapYear( int year )

{

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

{

return 1;

}

else

{

return 0;

}

}

int getDaySeqOnJan1( int year )

{

int daySeqOnJan1 = 1;

int currentYear = 2018;

for( ; currentYear < year ; currentYear ++ )

{

daySeqOnJan1 += ( 1 + isLeapYear( currentYear ));

}

return (daySeqOnJan1 - 1) % 7 + 1;

}

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

{

int i;

int daySeq = day;

if( isLeapYear( year ) == 1 )

{

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

{

daySeq += MonthLengthInLeapYear[i];

}

}

else

{

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

{

daySeq += MonthLengthInNormalYear[i];

}

}

return daySeq;

}

int getMonth( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > MonthLengthInLeapYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > MonthLengthInNormalYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInNormalYear[month];

}

}

return month;

}

int getDay( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > MonthLengthInLeapYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > MonthLengthInNormalYear[month] ; month ++ )

{

daySeqOfYear -= MonthLengthInNormalYear[month];

}

}

return daySeqOfYear;

}

int getDaySeqOfWeek( int year, int daySeqOfYear )

{

int daySeqOnJan1 = getDaySeqOnJan1( year );

return ( daySeqOfYear + daySeqOnJan1 - 2 ) % 7 + 1;

}

int getThisSunday( int year, int daySeqOfYear )

{

int daySeqOfWeek = getDaySeqOfWeek( year, daySeqOfYear );

return daySeqOfYear + ( 7 - daySeqOfWeek );

}

int getThisMonday( int year, int daySeqOfYear )

{

int daySeqOfWeek = getDaySeqOfWeek( year, daySeqOfYear );

return daySeqOfYear - daySeqOfWeek + 1;

}

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

{

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

return daySeq / 7 + 1;

}

void setYearArray( int array[], int YearNum, int inputYear )

{

int i;

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

{

array[i] = YEAR\_MIN + i;

}

return;

}

void printOneWeek( int year, int weekSeqOfYear)

{

int daySeqOnJan1 = getDaySeqOnJan1( year );

int mondaySeq = weekSeqOfYear \* 7 - daySeqOnJan1 - 5;

int sundaySeq = mondaySeq + 6;

int currentSeq,currentDay,currentMonth;

for( currentSeq = mondaySeq ; currentSeq <= sundaySeq ; currentSeq ++ )

{

currentDay = getDay( year, currentSeq );

currentMonth = getMonth( year, currentSeq );

printf("%02d.%02d",currentMonth,currentDay );

printf("%5s","");

}

return;

}

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 ) + 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] = getDaySeqOfWeek( year, j + 1 );

}

}

return;

}

九，Calendar10-日期结构体

#include <stdio.h>

#include <stdlib.h>

#define YEAR\_NUM 3

#define DATE\_NOSHOW 1

#define DATE\_INFO\_FULL 2

#define DATE\_INFO\_BRIEF 3

#define DATE\_STAR 4

const int monthDayInNormalYear[13] = {0,31,28,31,30,31,30,31,31,30,31,30,31};

const int monthDayInLeapYear[13] = {0,31,29,31,30,31,30,31,31,30,31,30,31};

const int Years[YEAR\_NUM] = {2022,2023,2024};

const int printDayRange = 3;

typedef struct

{

int year;

int daySeq;

int month;

int day;

int weekSeq;

int weekDay;

}Day;

//Functions for years

//

int isLeapYear(int year);

int getDaySeqOnJan1(int year);

//Functions for information of dates

//

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

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

int getMonth(int year, int daySeqOfYear);

int getDay(int year,int daySeqOfYear);

int getDaySeqOfWeek(int year, int daySeqOfYear);

//Functions in main

//

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

Day setDay(int year,int month,int day );

Day getDayBefore(Day currentDay , int interval );

Day getDayAfter( Day currentDay, int interval );

int getTwoDaysInterval(Day startDay, Day endDay );

//Function as a printer

//

void printDay(Day currentDay, int displayFormat);

int main( void )

{

int inputMonth;

int inputDay;

int displayFormat;

Day birthDay;

Day prepareDay;

Day printStartDay;

Day currentDay;

printf("Finding Birthday in year(2022-2024),please input the month and day:\n");

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

int i;

for ( i = 0 ;i < YEAR\_NUM ; 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( prepareDay, prepareDay.weekDay );

printf("#W:%10s%10s%10s%10s%10s%10s%10s\n[%02d]",

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

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);

if(currentDay.weekDay == 6 && displayFormat != DATE\_STAR)

{

printf("\n #W:%10s%10s%10s%10s%10s%10s%10s\n[%02d]",

"Mon.","Tues.","Wed.","Thur.","Fri.","Sat.","Sun.",printStartDay.weekSeq + 1);

}

}

printf("\n");

}

return 0;

}

int isLeapYear(int year)

{

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

{

return 1;

}

else

{

return 0;

}

}

int getDaySeqOnJan1( int year )

{

int daySeqOnJan1 = 1;

int currentYear = 2018;

for( ; currentYear < year ; currentYear ++ )

{

daySeqOnJan1 += ( 1 + isLeapYear( currentYear ));

}

return (daySeqOnJan1 - 1) % 7 + 1;

}

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

{

int i;

int daySeq = day;

if( isLeapYear( year ) == 1 )

{

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

{

daySeq += monthDayInLeapYear[i];

}

}

else

{

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

{

daySeq += monthDayInNormalYear[i];

}

}

return daySeq;

}

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

{

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

return daySeq / 7 + 1;

}

int getMonth( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > monthDayInLeapYear[month] ; month ++ )

{

daySeqOfYear -= monthDayInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > monthDayInNormalYear[month] ; month ++ )

{

daySeqOfYear -= monthDayInNormalYear[month];

}

}

return month;

}

int getDay( int year, int daySeqOfYear )

{

int month;

if( isLeapYear( year ) == 1 )

{

for( month = 1; daySeqOfYear > monthDayInLeapYear[month] ; month ++ )

{

daySeqOfYear -= monthDayInLeapYear[month];

}

}

else

{

for( month = 1; daySeqOfYear > monthDayInNormalYear[month] ; month ++ )

{

daySeqOfYear -= monthDayInNormalYear[month];

}

}

return daySeqOfYear;

}

int getDaySeqOfWeek( int year, int daySeqOfYear )

{

int daySeqOnJan1 = getDaySeqOnJan1( year );

return ( daySeqOfYear + daySeqOnJan1 - 2 ) % 7 + 1;

}

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

{

if((month < 0 || month > 12) || day < 0)

{

return 0;

}

else if(isLeapYear(year) == 1 && day > monthDayInLeapYear[month])

{

return 0;

}

else if(isLeapYear(year) == 0 && day > monthDayInNormalYear[month])

{

return 0;

}

else

{

return 1;

}

}

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

{

Day d;

d.year = year;

d.month = month;

d.day = day;

d.daySeq = getDaySeq(year, month, day);//day in a year

d.weekSeq = getWeekSeqOfYear(year, month, day);//week in a year

d.weekDay = getDaySeqOfWeek(year, d.daySeq);//day in a week

return d;

}

Day getDayBefore(Day currentDay , int interval )

{

currentDay.day -= interval;

currentDay.weekDay -= interval;

currentDay.daySeq -= interval;

if(currentDay.weekDay <= 6)

{

currentDay.weekSeq -= (currentDay.weekDay / 7 + 1);

currentDay.weekDay = (currentDay.weekDay % 7 + 7 ) % 7;

}

if(currentDay.day < 1)

{

if(currentDay.month == 1)

{

currentDay.year --;

currentDay.month = 12;

currentDay.daySeq += (isLeapYear(currentDay.year) + 365);

currentDay.day += 31;

currentDay.weekSeq = getWeekSeqOfYear(currentDay.year, currentDay.month, currentDay.day);

}

else

{

currentDay.month --;

if(isLeapYear(currentDay.year) == 0)

{

currentDay.day += monthDayInNormalYear[currentDay.month];

}

else

{

currentDay.day += monthDayInLeapYear[currentDay.month];

}

}

}

return currentDay;

}

Day getDayAfter( Day currentDay, int interval )

{

Day dayAfter;

if(currentDay.daySeq + interval <= isLeapYear(currentDay.year) + 365)

{

dayAfter.year = currentDay.year;

dayAfter.daySeq = currentDay.daySeq + interval;

dayAfter.month = getMonth(dayAfter.year,dayAfter.daySeq);

dayAfter.day = getDay(dayAfter.year,dayAfter.daySeq);

dayAfter.weekSeq = getWeekSeqOfYear(dayAfter.year,dayAfter.month,dayAfter.day);

dayAfter.weekDay = getDaySeqOfWeek(dayAfter.year,dayAfter.daySeq);

}

else

{

dayAfter.year = currentDay.year + 1;

dayAfter.daySeq = currentDay.daySeq + interval - isLeapYear(currentDay.year) - 365;

dayAfter.month = getMonth(dayAfter.year,dayAfter.daySeq);

dayAfter.day = getDay(dayAfter.year,dayAfter.daySeq);

dayAfter.weekSeq = getWeekSeqOfYear(dayAfter.year,dayAfter.month,dayAfter.day);

dayAfter.weekDay = getDaySeqOfWeek(dayAfter.year,dayAfter.daySeq);

}

return dayAfter;

}

int getTwoDaysInterval(Day startDay, Day endDay )

{

if(startDay.year < endDay.year)

{

return -1;

}

else if(startDay.year == endDay.year)

{

if(startDay.daySeq < endDay.daySeq)

{

return -1;

}

else if (startDay.daySeq == endDay.daySeq)

{

return 0;

}

else

{

return 1;

}

}

}

void printDay(Day currentDay, int displayFormat)

{

switch (displayFormat)

{

case 1:

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

break;

case 2:

printf("%4d.%02d.%02d", (currentDay.year % 100), currentDay.month, currentDay.day);

break;

case 3:

printf("%10d", currentDay.day);

break;

case 4:

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

break;

}

return;

}