

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 | 覃祉璇 | 学号 | U202410288 |

|  |  |  |  |
| --- | --- | --- | --- |
| 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（日历系列、大数计算系列，选择其中之一）。

# 实验环境

操作系统：Windows 10

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］

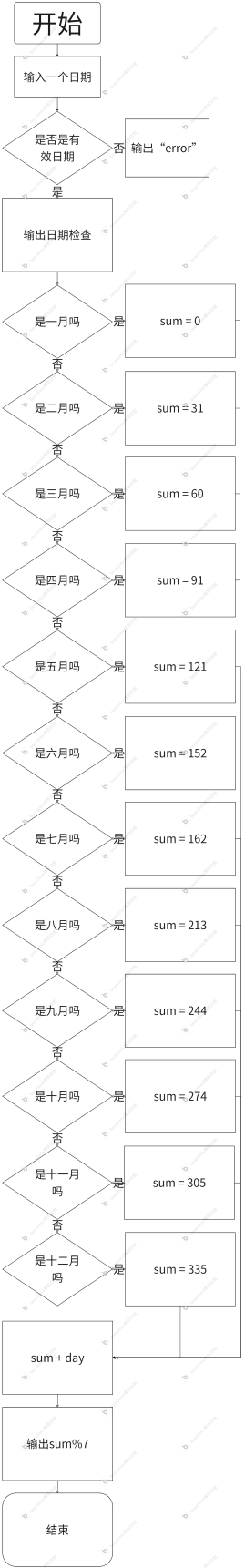
-运用所学到的知识，编写一个计算星期几的程序，计算2021年某月某日是星期几（已知2021年1月1日是星期五）

•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

•约定每个星期从周一开始，如果是周一则打印1，周二则打印2.⋯⋯，周日打印7

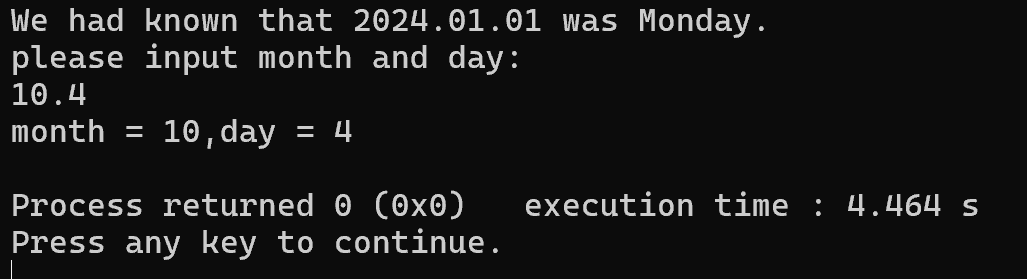
• 提示：需要提前学习if 语句，不需要用数组

## 实验步骤



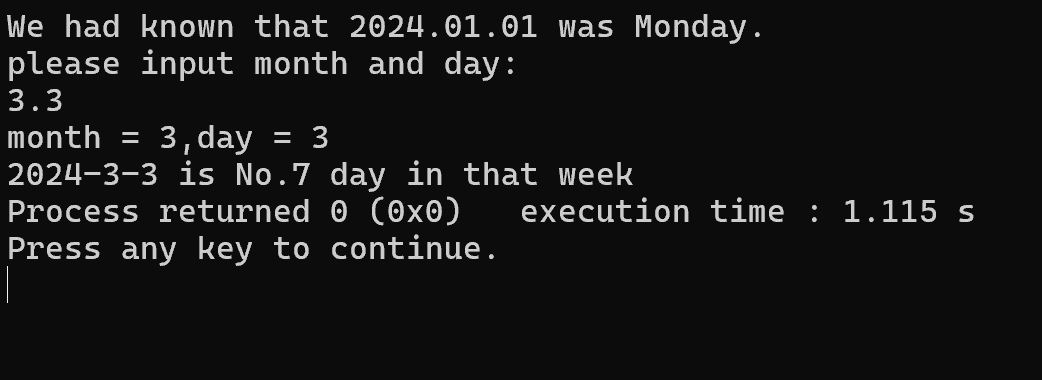
## 代码测试

### 测试点日期输入的测试结果



能正确输入日期

### 测试点星期输出的测试结果



成功输出

## 实验结论

代码能达到预期效果

## 实验总结

If语句太过繁琐，执行起来十分不方便

# 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2021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

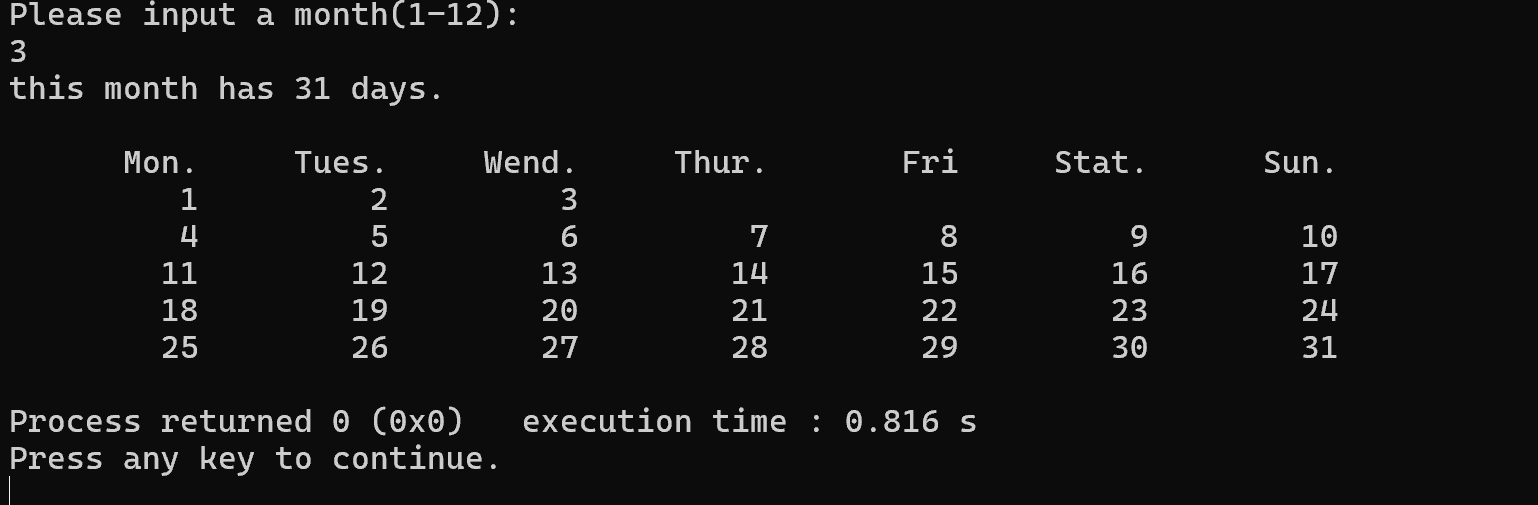
• 【提示】设置一个变量存储每个月的总天数；设置一个变量存储第一天是星期几（利用前一个实验计算）；利用循环控制语句打印该月的每一天，如果碰到周日要換行

## 实验步骤

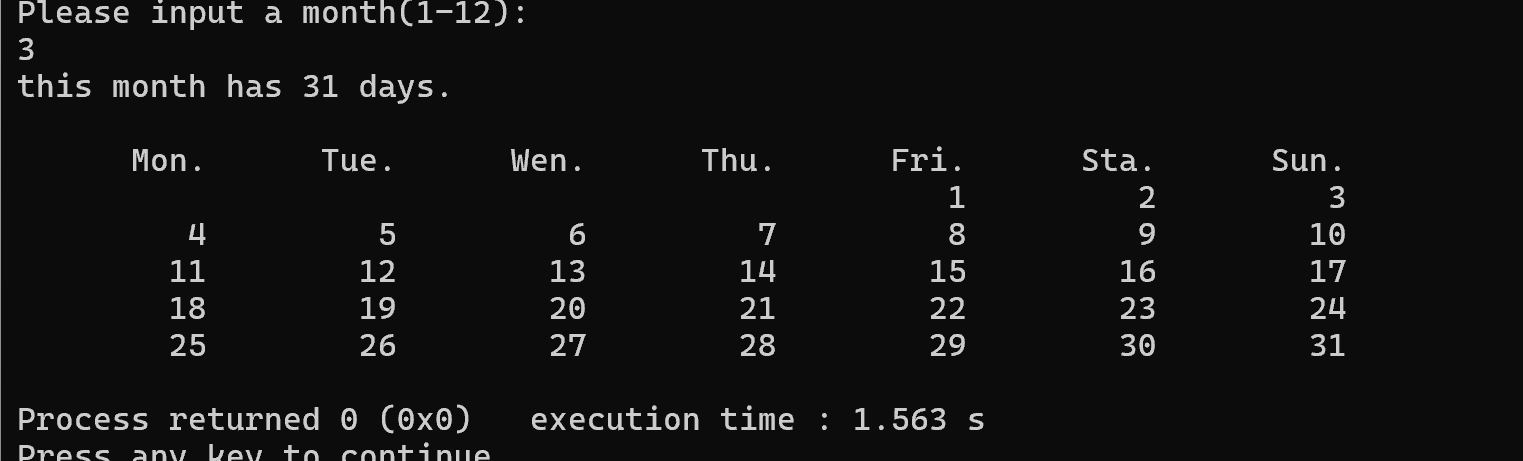


## 代码测试

### 测试点无缩进但是有换行的测试结果



### 测试点又缩进换行的测试结果



针对每个测试点，说明测试步骤，预期测试结果，实际测试结果（截图），测试结论

## 实验结论

能打印出正确月历

## 实验总结

该实验中运用到数组使程序极大简化，并且完成缩进和换行。

# 实验三

## 实验任务

•［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2021年某一周的周历（2021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；

结束日是舍年的第几天，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



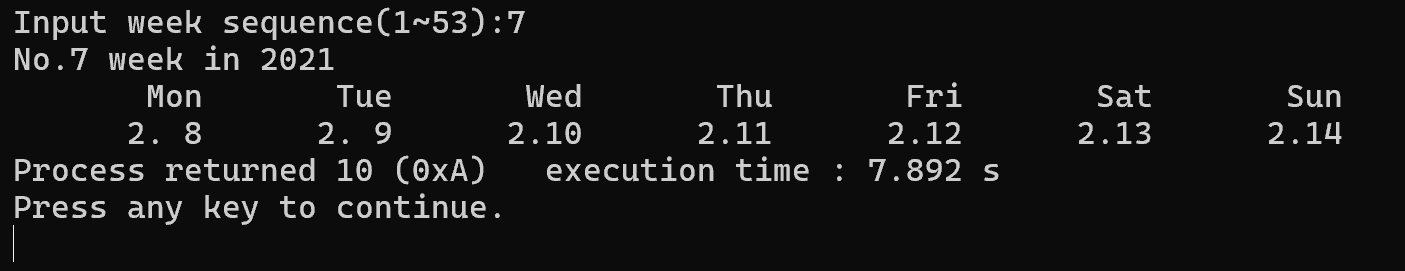
## 代码测试

### 测试点打印第一天的测试结果

### 

成功打印第一天

### 测试点打印整周的测试结果



成功打印

## 实验结论

代码达到功能目标

## 实验总结

使用数组并用累加可以定位到该周的第一天，再通过for循环打印七天已经较为方便，熟练运用循环语句。

# 实验四

## 实验任务

Calendar05. 春季校历函数版（2021春）

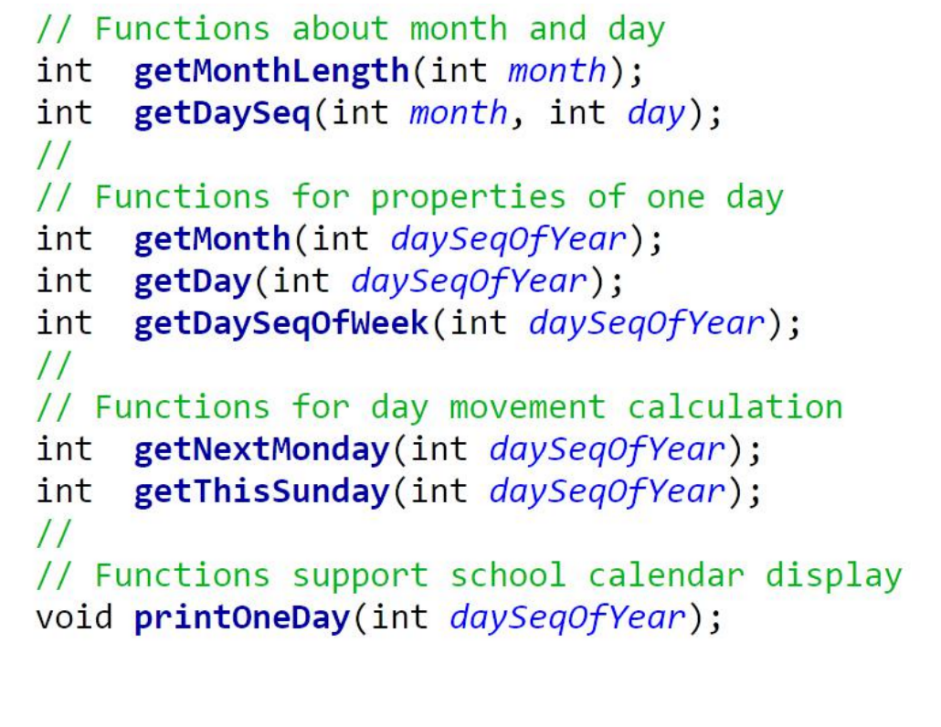
•［实验目标］

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

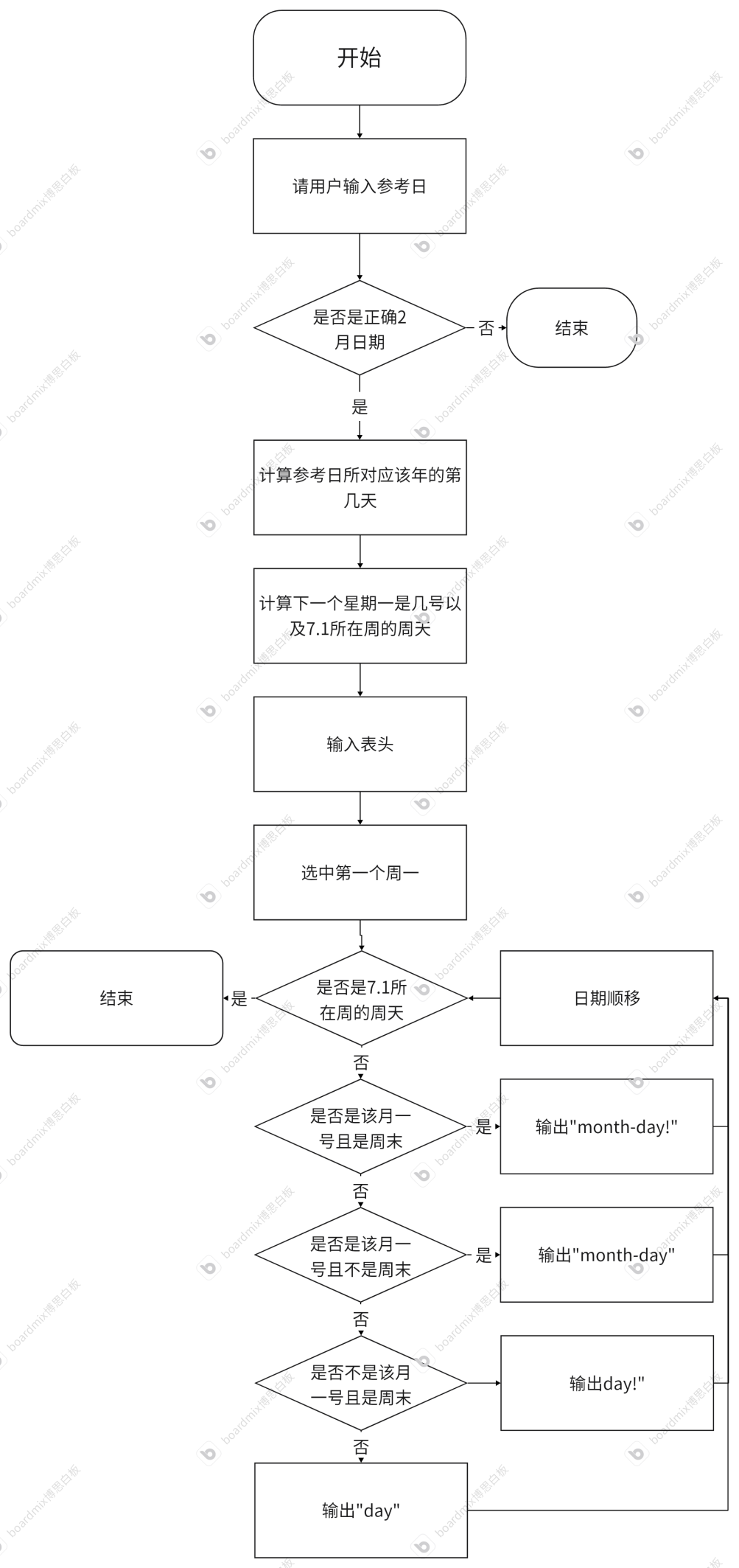
•［实验任务］

-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



## 实验步骤



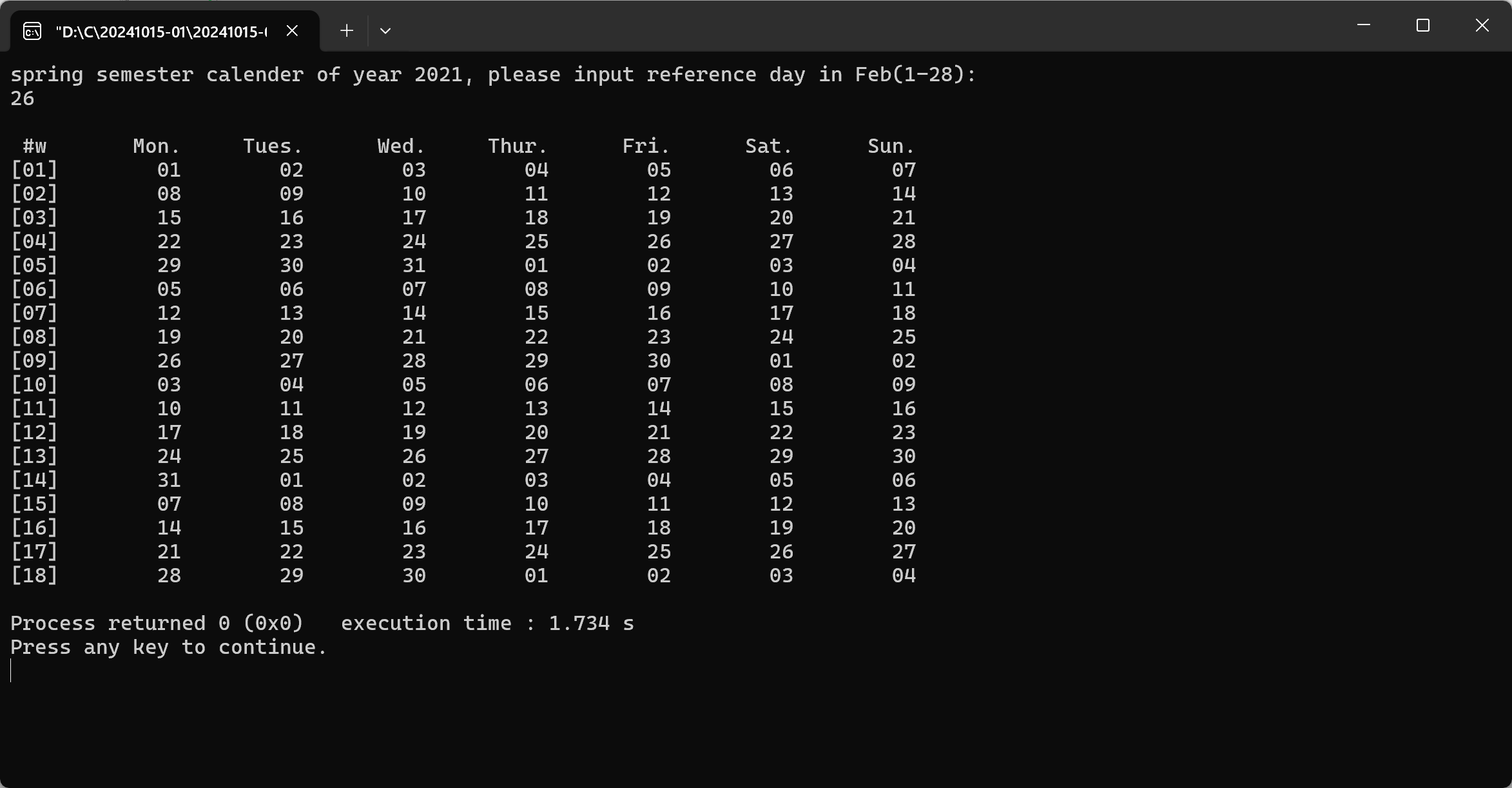
## 代码测试

### 测试点输入错误日期的测试结果

### 

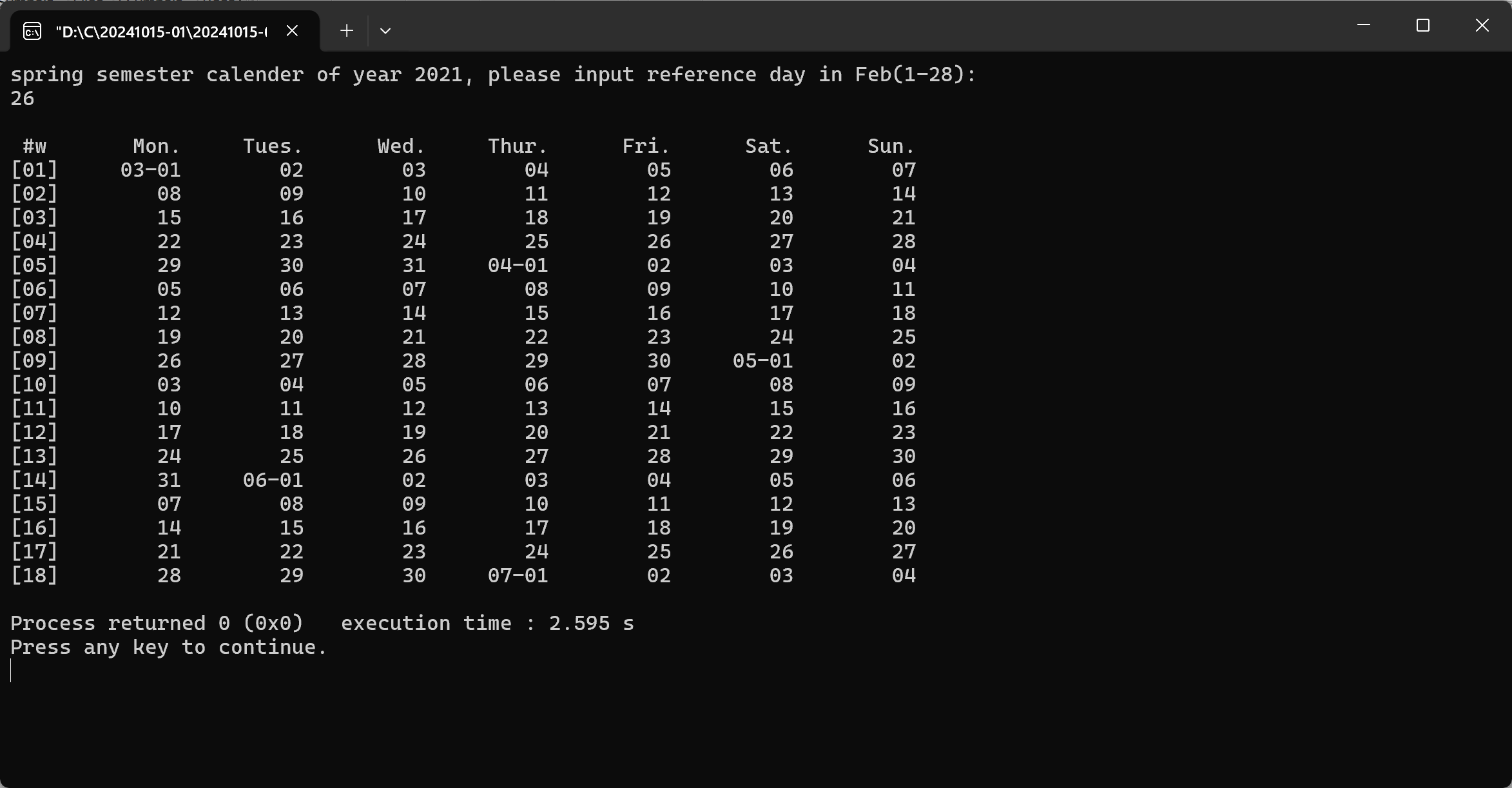
测试成功，报错弹出。

### 测试点直接打印校历的测试结果



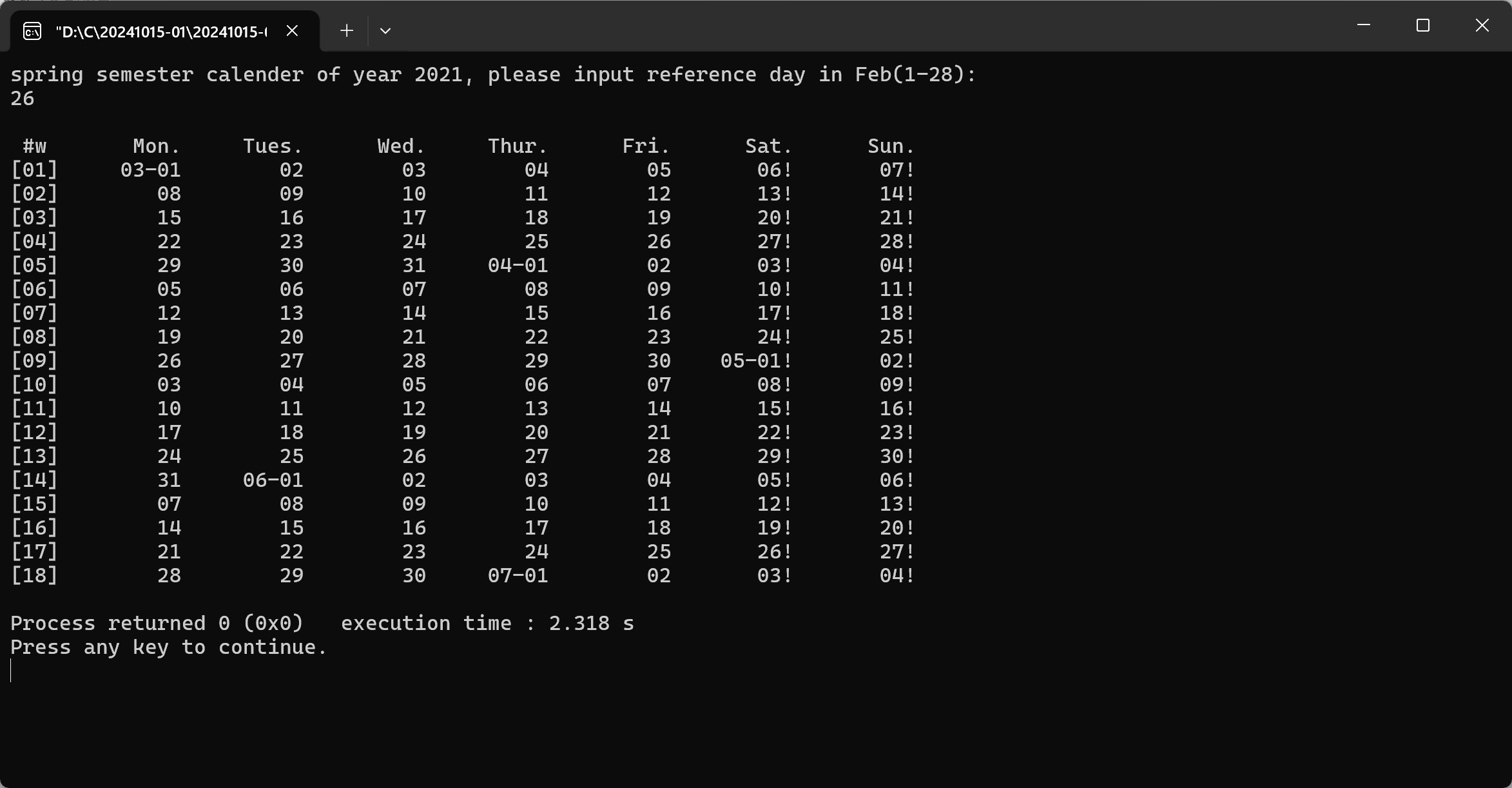
测试成功，对应的日期都正确。

### 测试点加上月首的测试结果



测试成功，月首都加上了月份。

### 测试点对周末标记的测试结果



测试成功，周末都用“！”标记。

## 实验结论

能达到实验目的

## 实验总结

利用了多个函数，可以跨月计算

# 实验五

## 实验任务

• [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

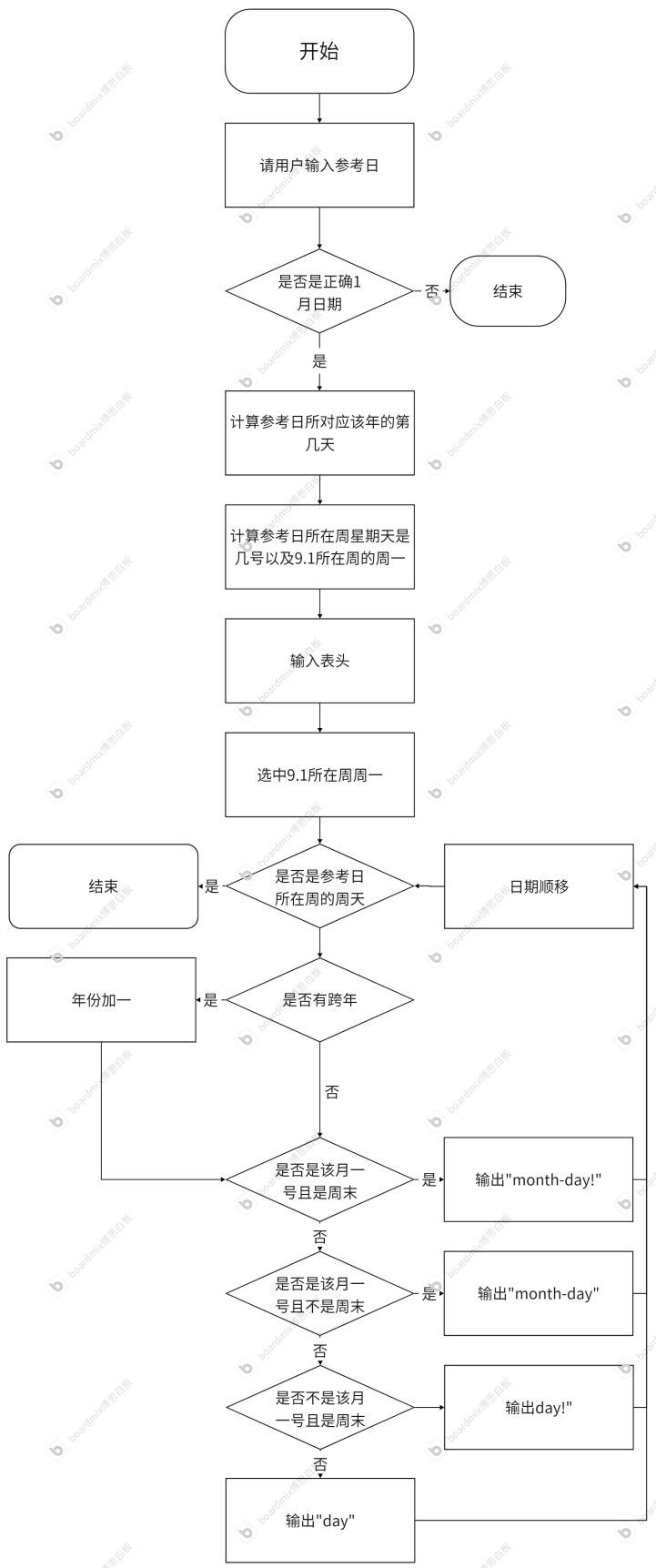
维护带来的优势

• [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。

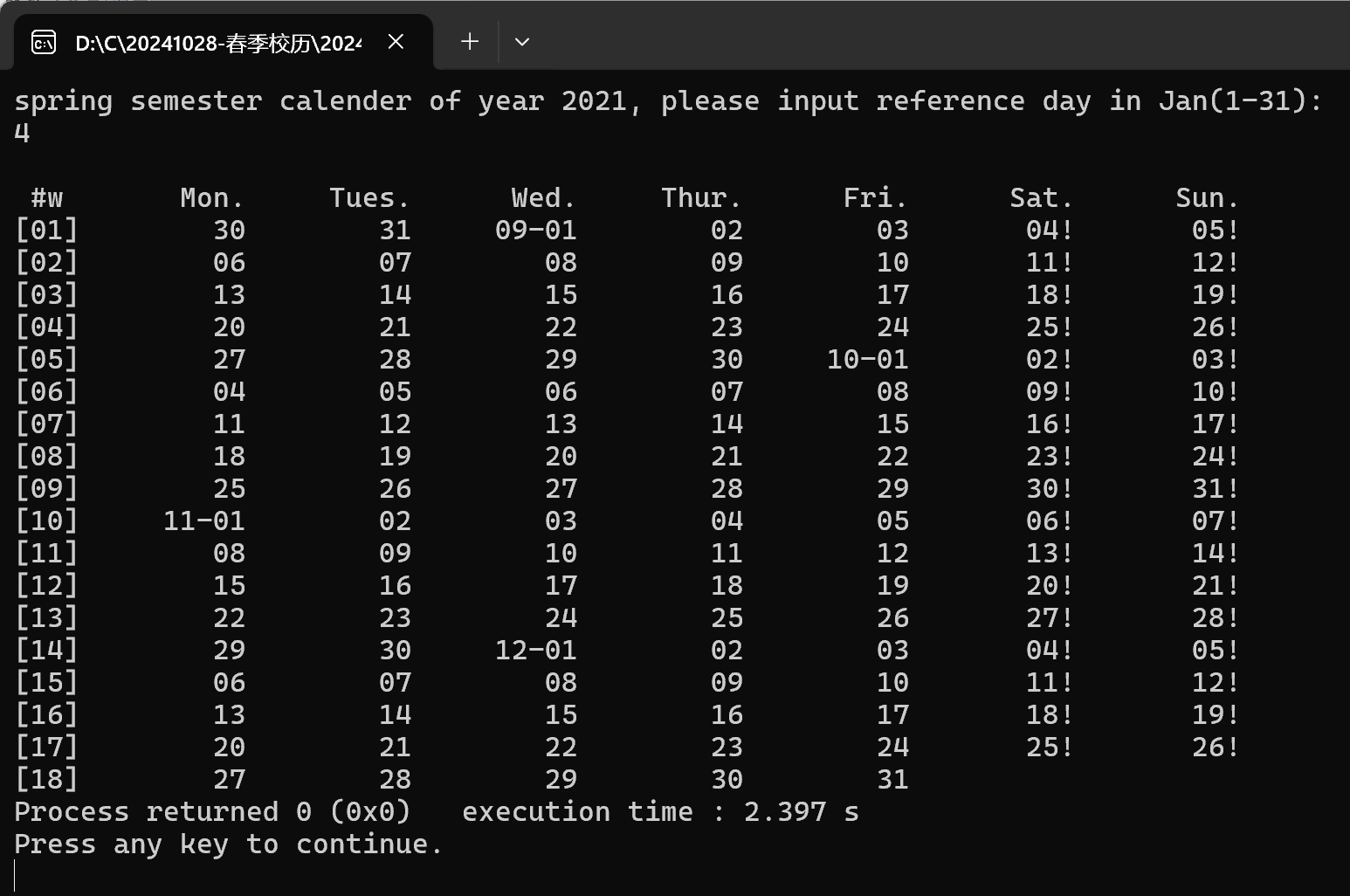
– 更新相关函数，支持不同的年份的日期，计算相关函数需要增加一个参数int year

## 实验步骤



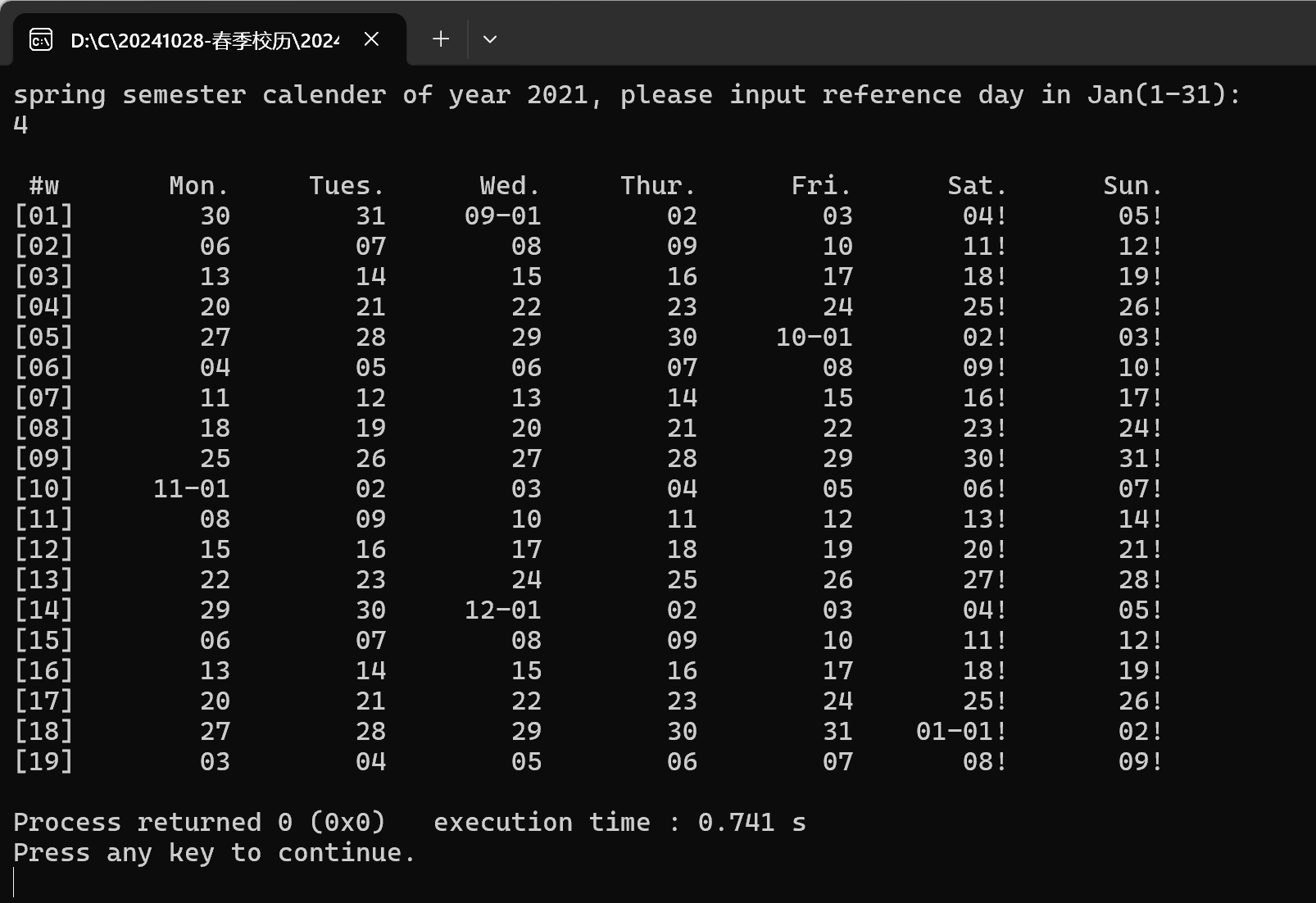
## 代码测试

### 测试点只打印2021年的测试结果



测试成功

### 测试点 打印跨年的测试结果



测试成功

## 实验结论

测试成功，可以跨年打印，并且跨月和周末也能清楚表示

## 实验总结

实验6就实验5而言多加入了跨年，函数变得麻烦了很多，每一个函数都要多加一个year的参数，但这样使得代码更加严谨。但就我的代码而言是针对2021-2022年的校历，要是要记载其他的年份，可能会要改动更多参数

# 实验六

## 实验任务

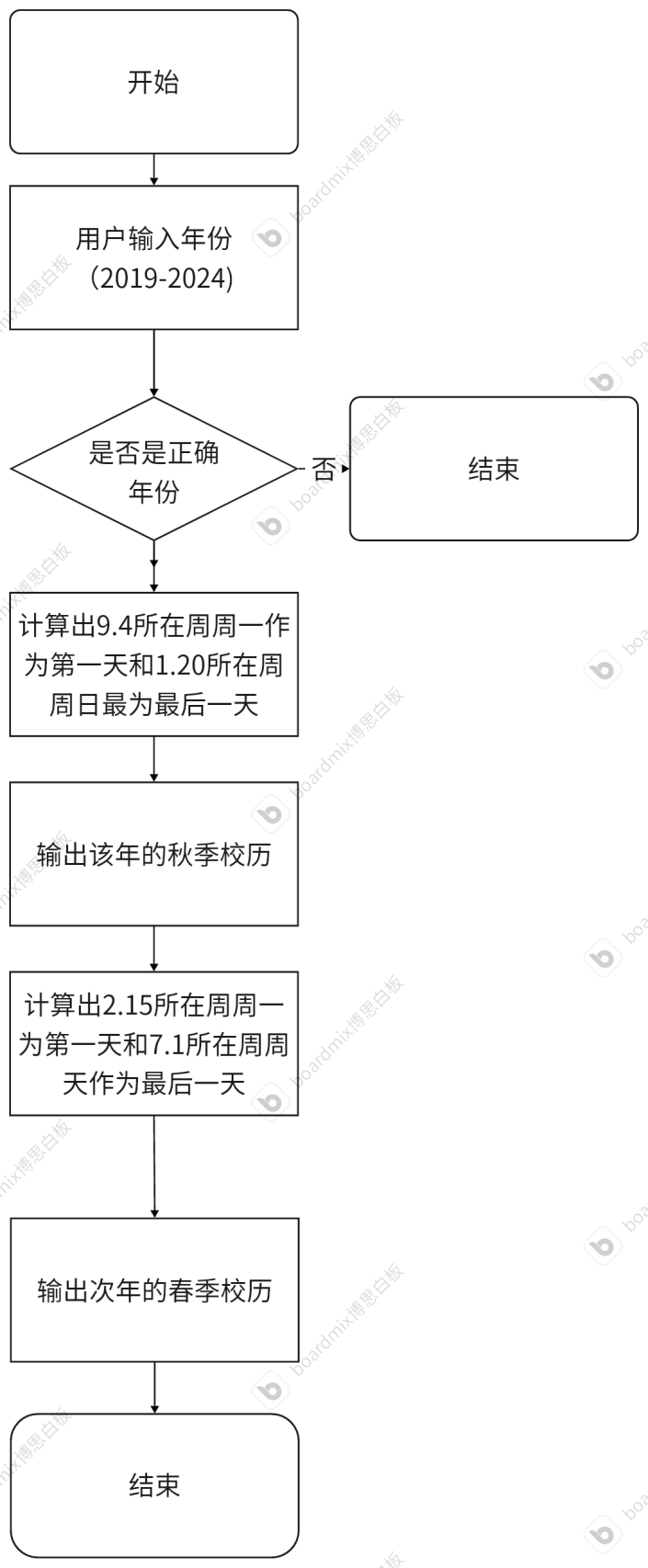
• [实验目标] – 巩固函数的使用，对现有函数进行功能升级（支持全学年校历打印），实现通用功能函数printOneWeek()，体会“面向过程编程”的设计方法

• [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。

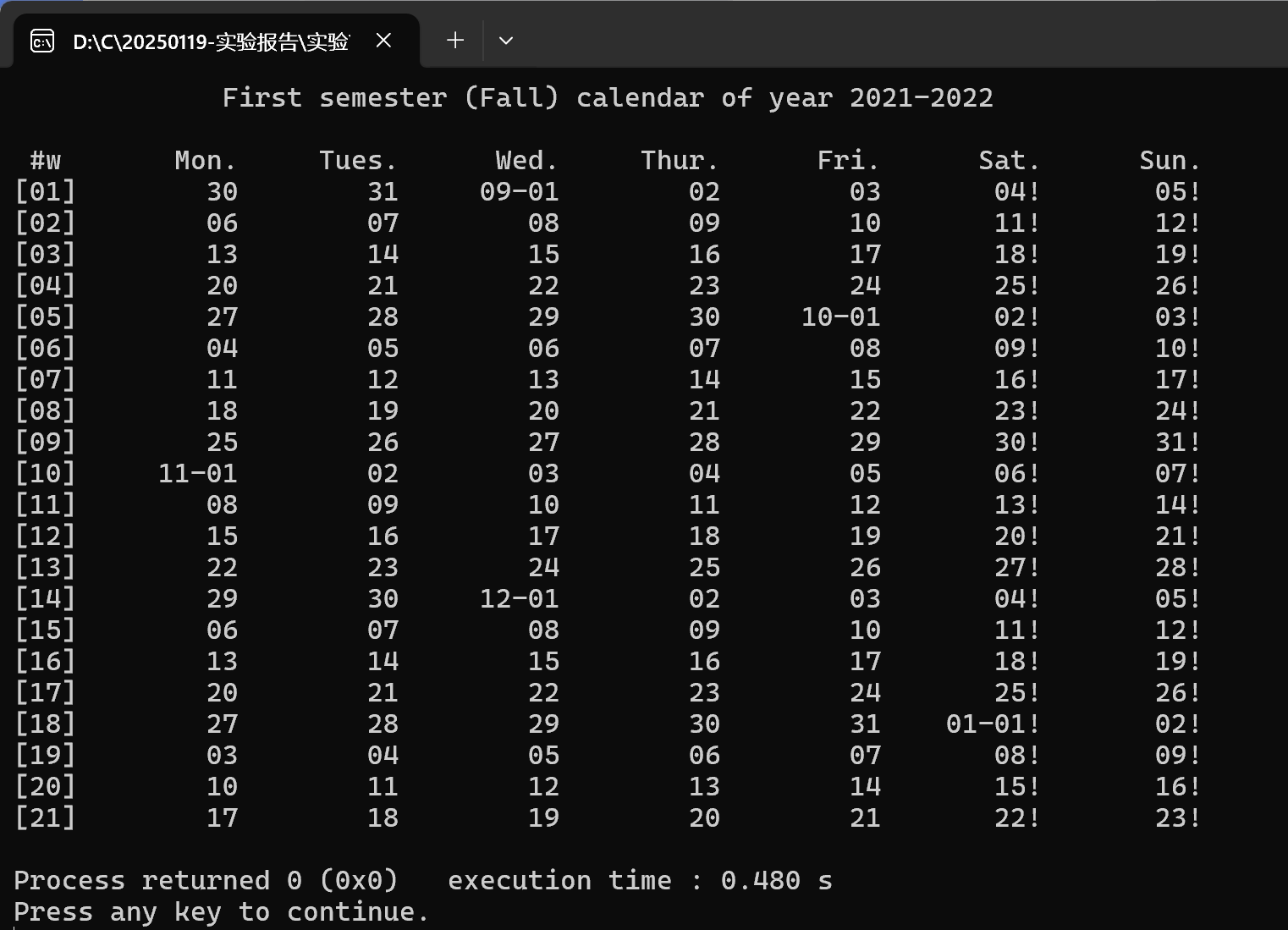
– 要求设计一个打印某周周历的函数，支持不同学期校历的打印需求；改进打印某日的函数，支持在校历首日和跨年日显示年月日信息。

## 实验步骤



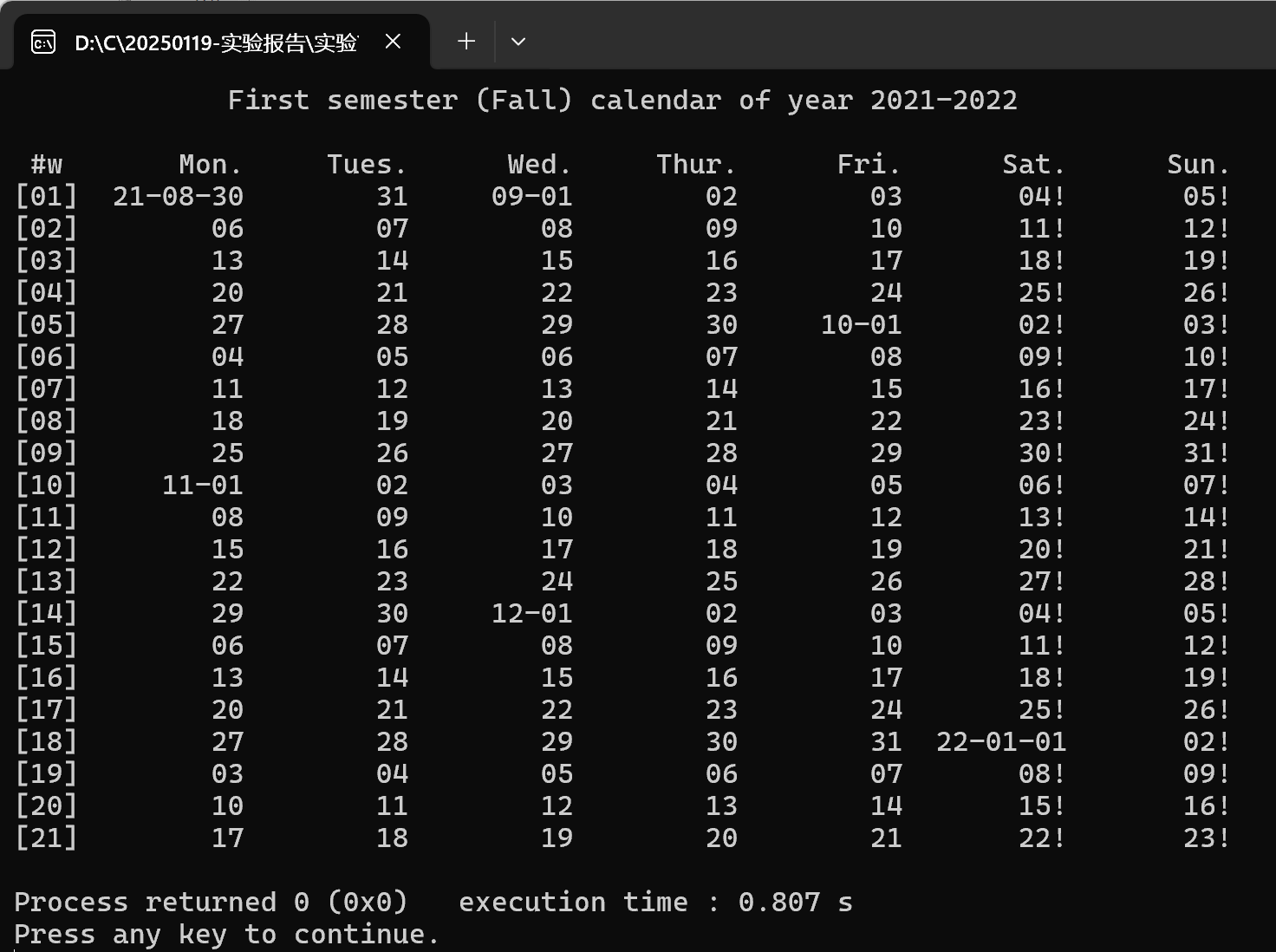
## 代码测试

### 测试点 打印秋季学期的测试结果



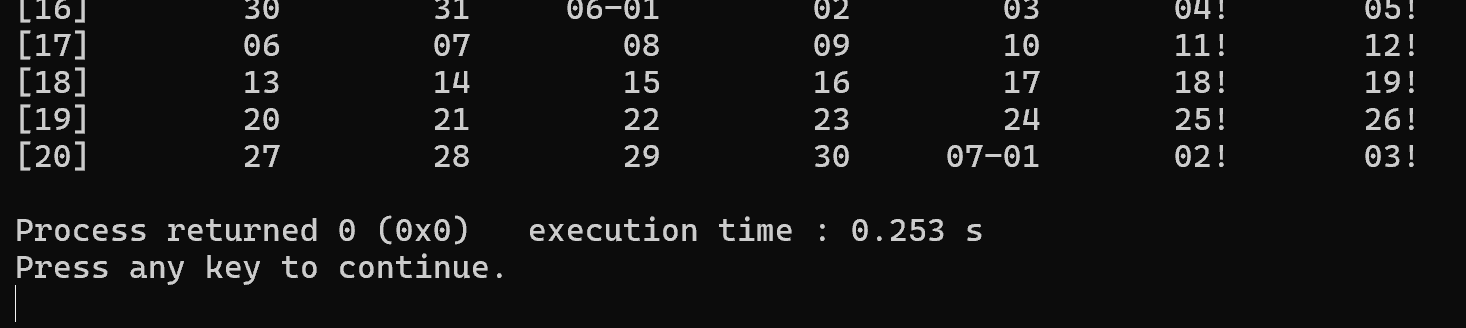
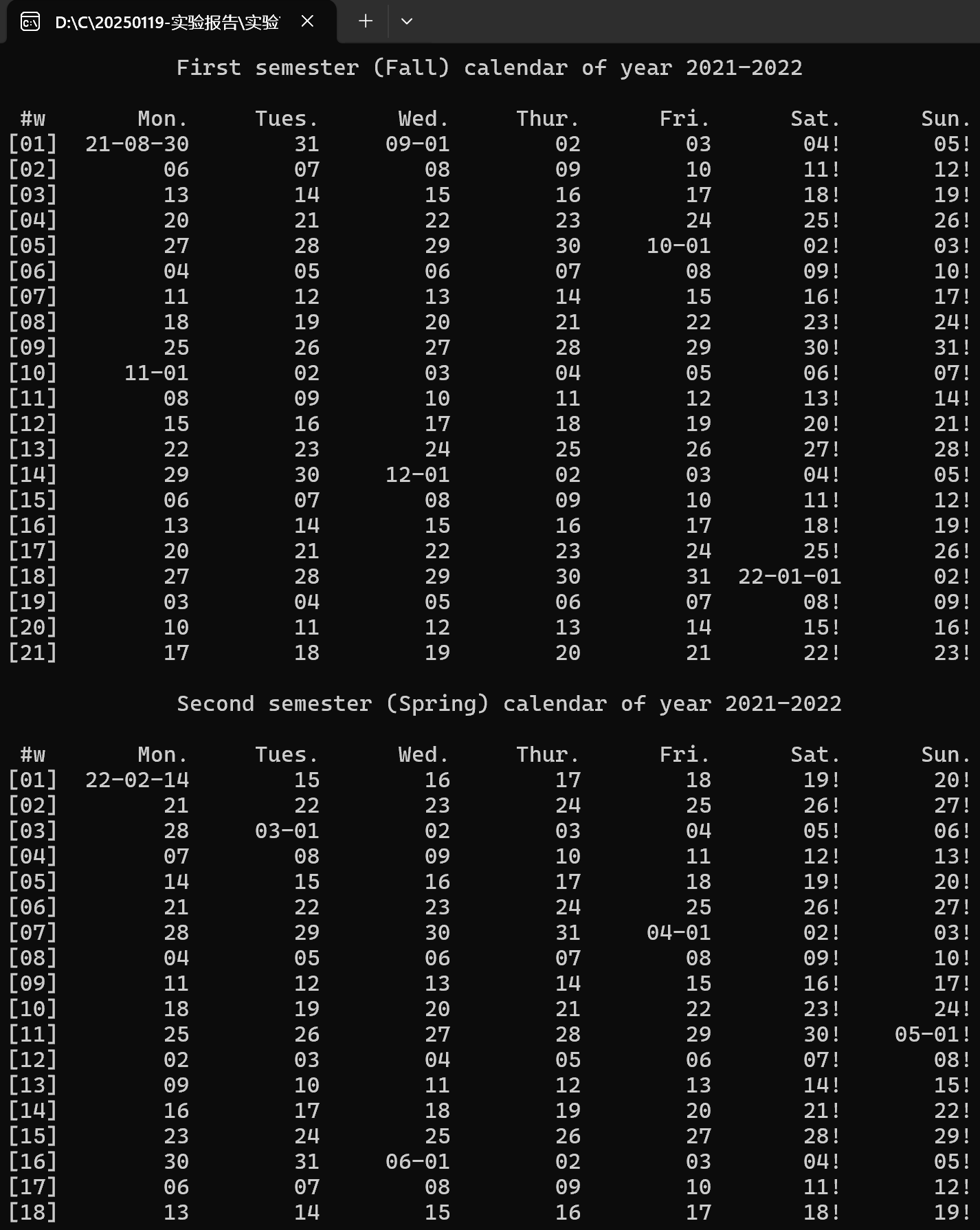
成功打印9.4所在周周一作为第一天和1.20所在周周日最为最后一天之间的日期。

### 测试点 打印年份变化的测试结果



成功打印出年份变化

### 测试点 打印年份变化的测试结果



成功

## 实验结论

能达到目的

## 实验总结

该实验就前两个就前两个而言，确定了开始和结束的日期，但是多加了一个年份变化，更符合平时所用

# 实验七

## 实验任务

•［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季校历的首日，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；约定春季学期从次年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

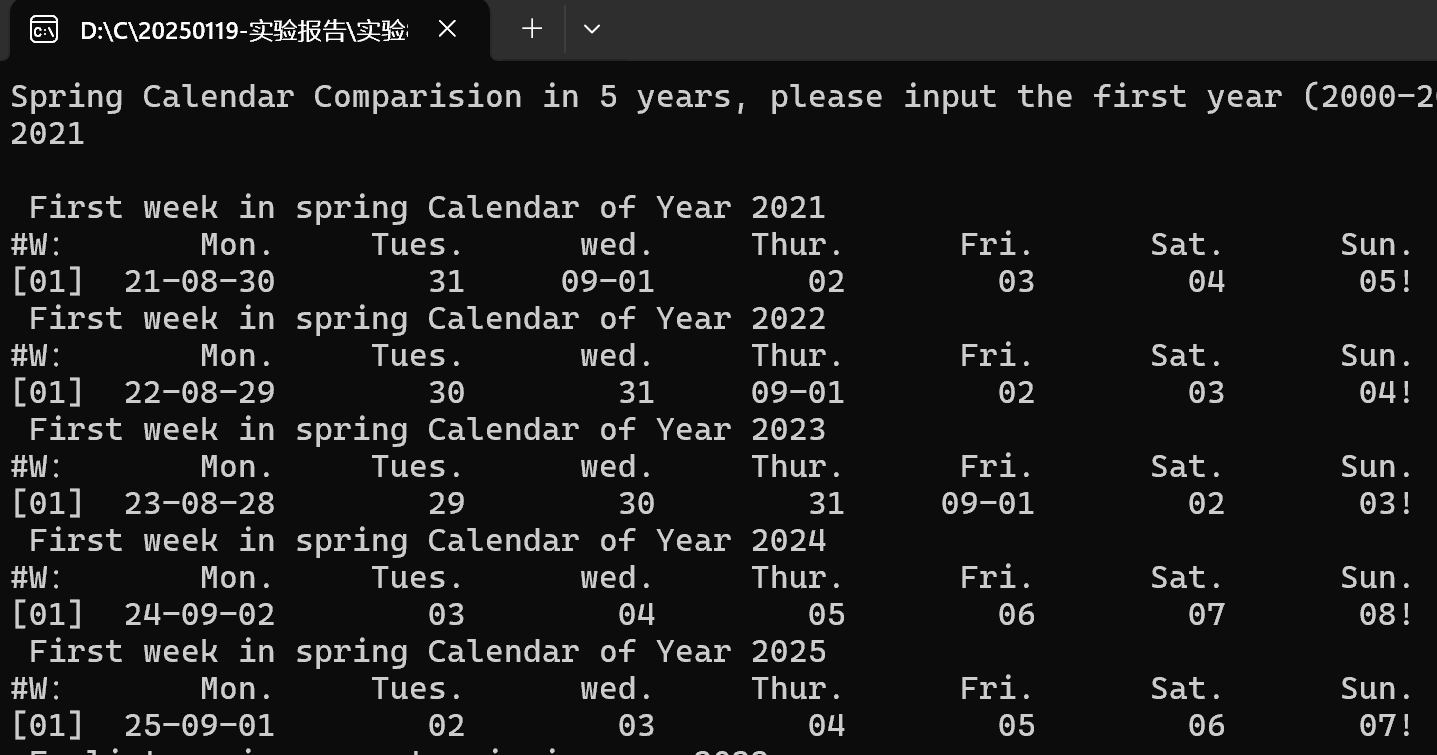
- 基于数组改进现有的日历程序，改进计算日期的函数

## 实验步骤



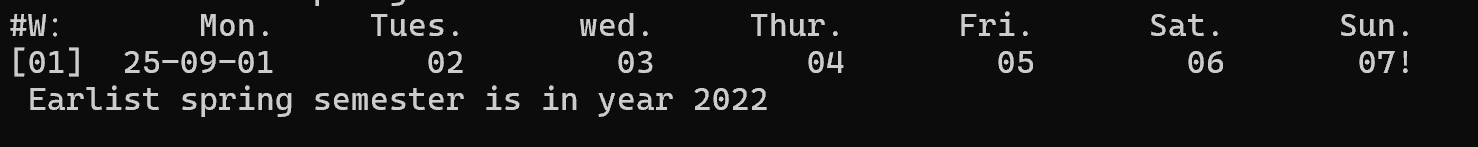
## 代码测试

### 测试点 按周输出的测试结果

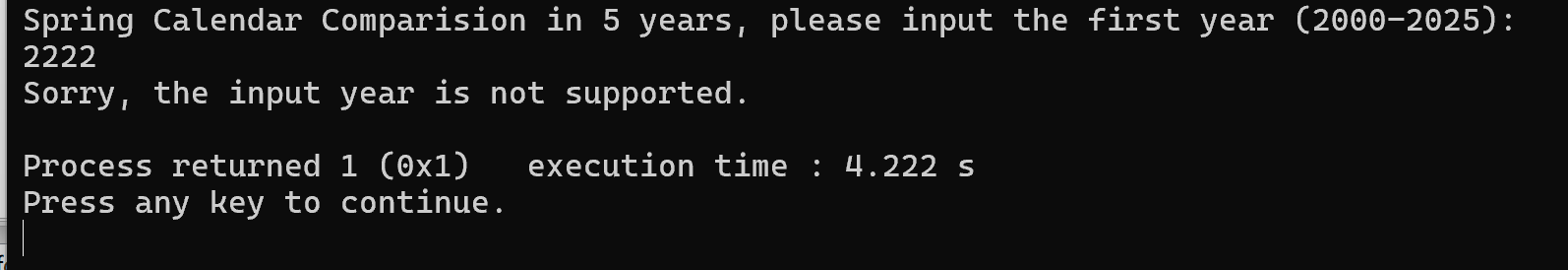


能按周输出连续5年

### 测试点 比较最早年份的测试结果



### 测试点 输入错误年份的测试结果



有报错

## 实验结论

代码能达到功能目标

## 实验总结

通过数组简化掉getmonthlength函数，其他函数也随之变化

# 实验八

## 实验任务

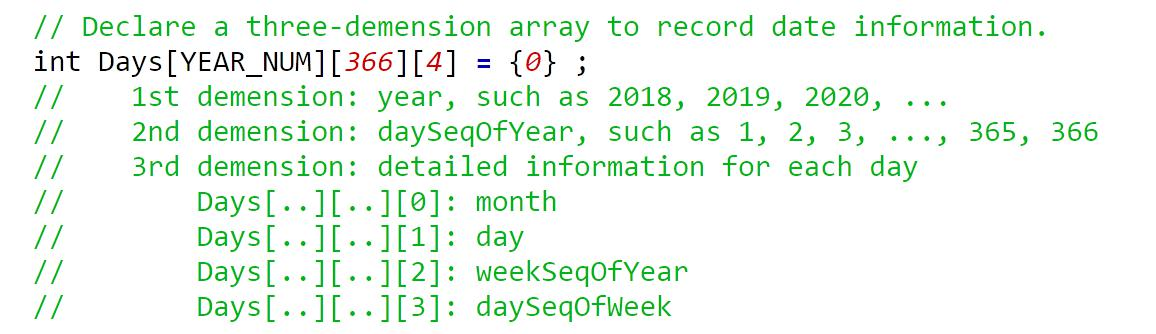
•［实验目标］

-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

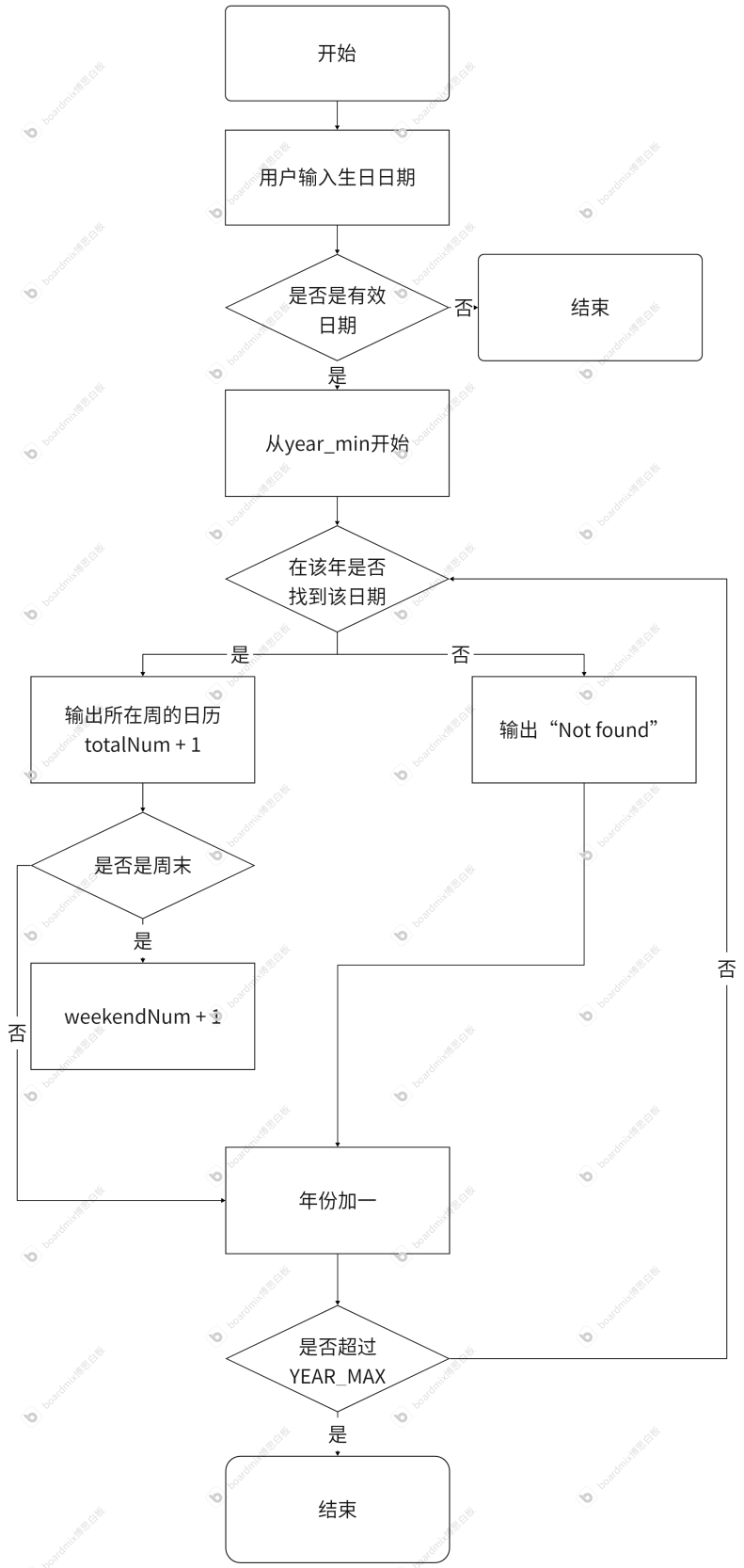
•［实验任务］

-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

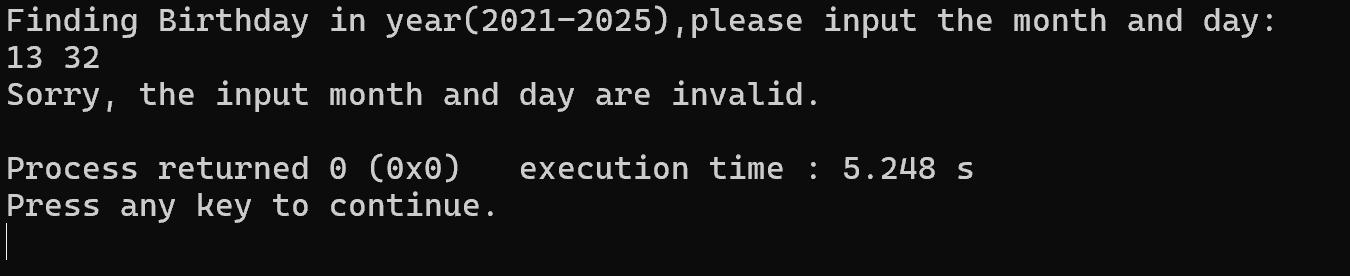


## 实验步骤



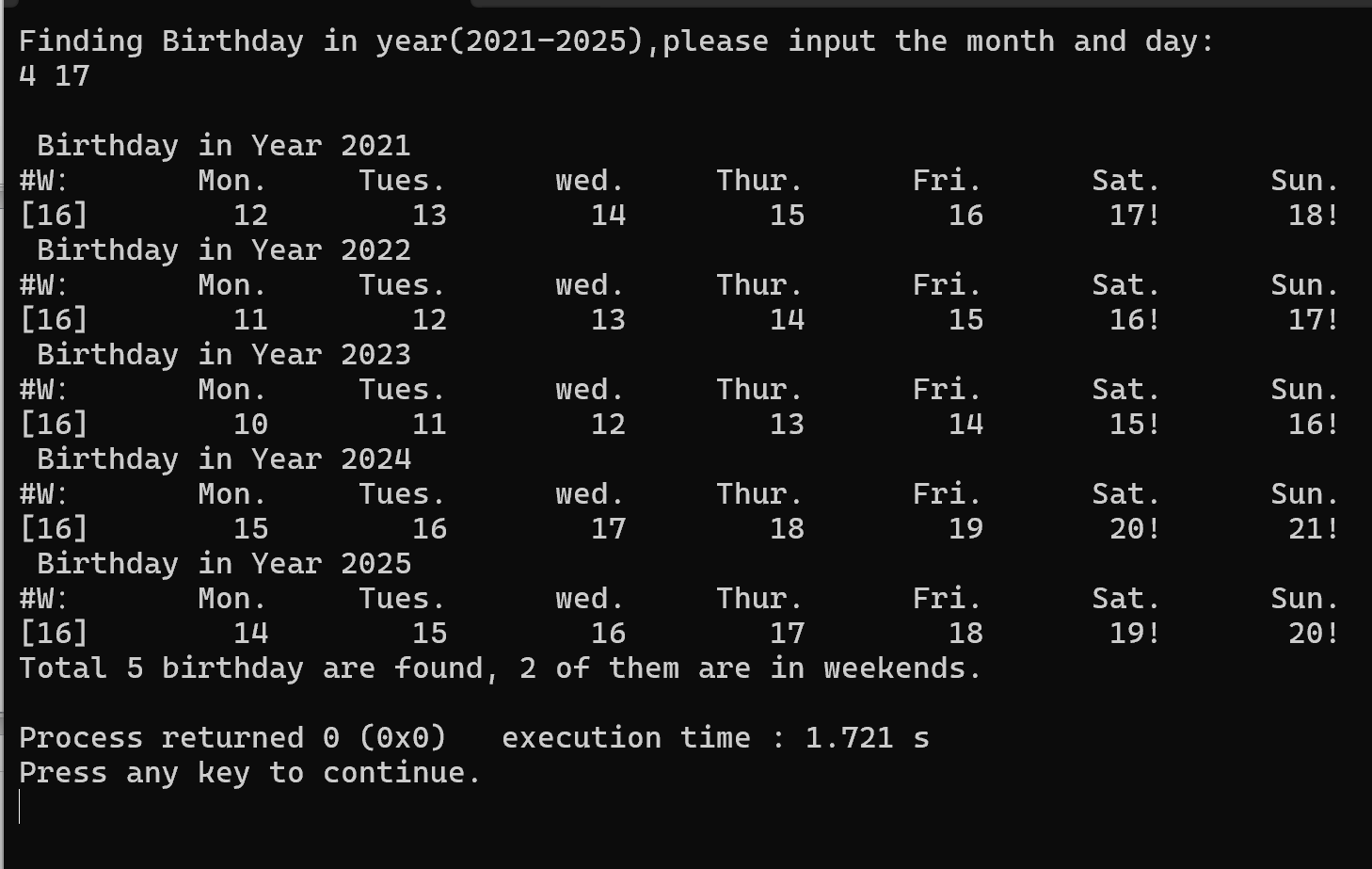
## 代码测试

### 测试点 输入错误日期的测试结果

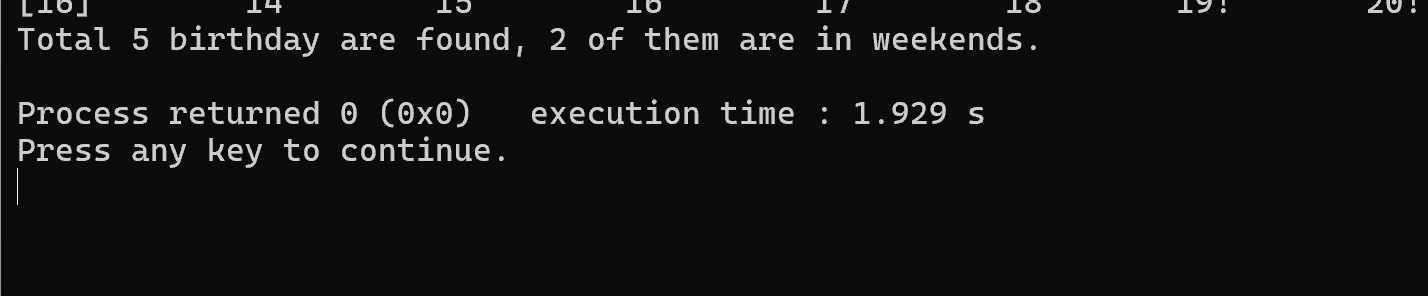


能报错并结束

### 测试点 输入正常日期的测试结果

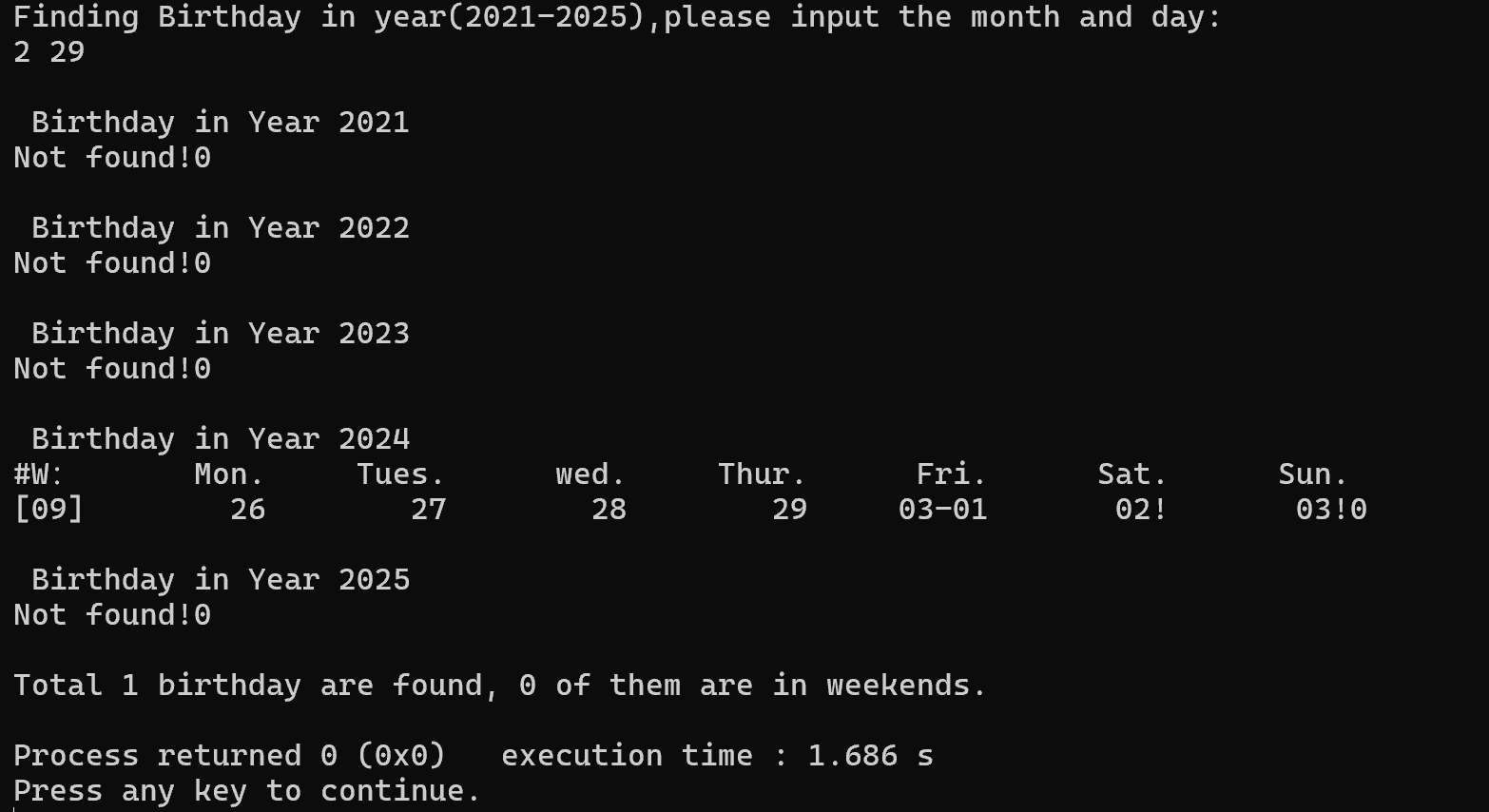


能打印该周



并且能正确记录出现次数以及在周末的次数

### 测试点 输入特殊日期（2，29）的测试结果



在闰年有记载

## 实验结论

代码能达到功能目标

## 实验总结

该实验中的各个函数的运用可以说已经较为熟练，也可以通过改变较少的数据来计算更多年份。

# 实验九

## 实验任务

• [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

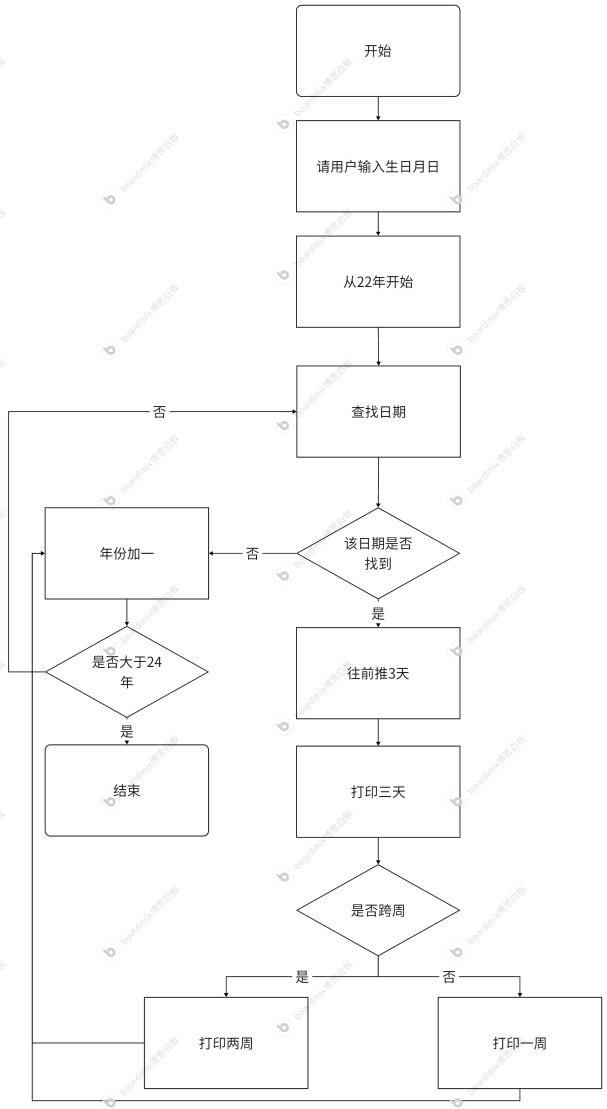
操作，体会“面向过程的编程”与“面向数据类型的编程”在设计思路上的差异

• [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：

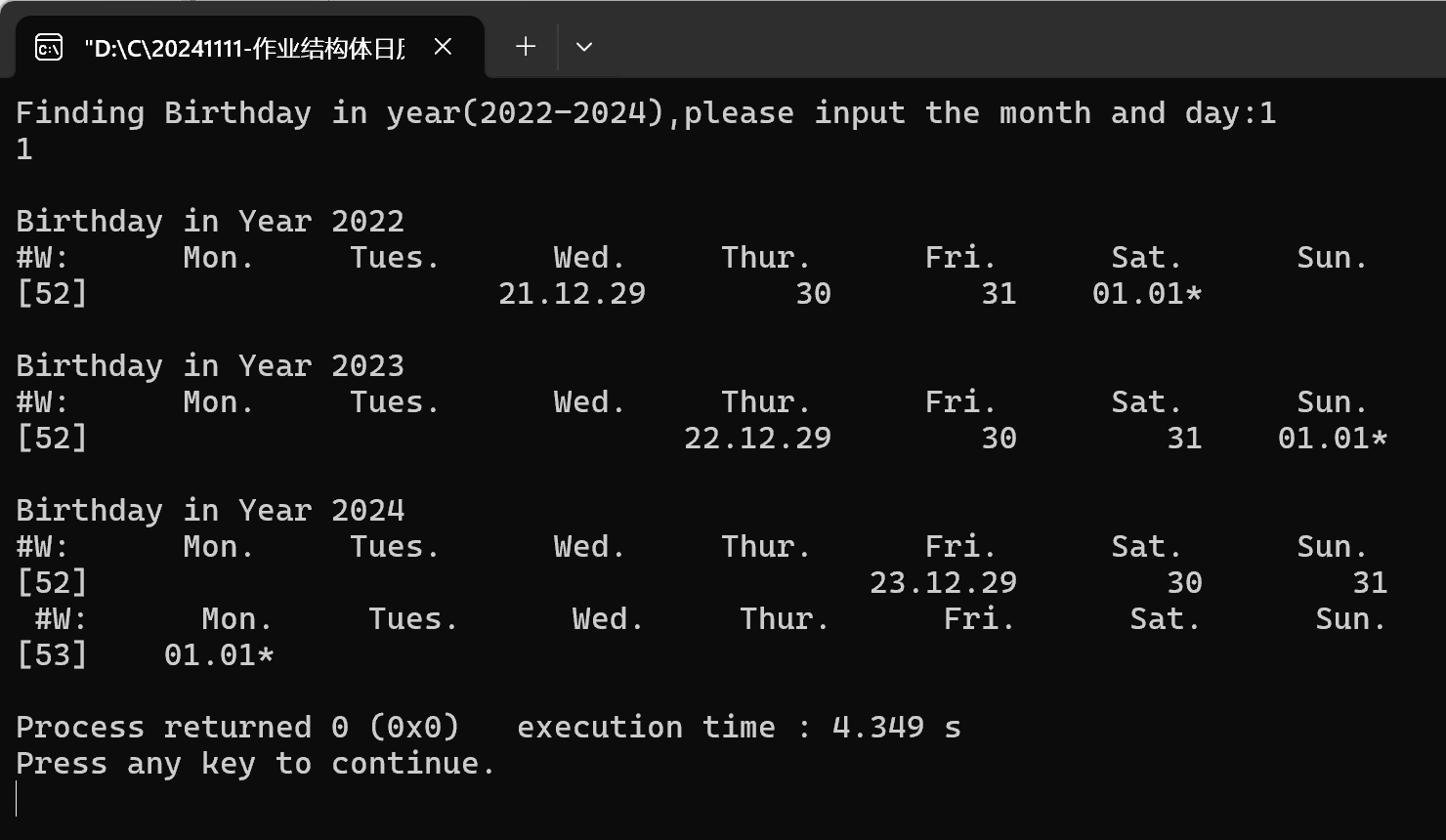
## 实验步骤



## 代码测试

### 测试点 跨年 的测试结果

针对每个测试点，说明测试步骤，预期测试结果，实际测试结果（截图），测试结论



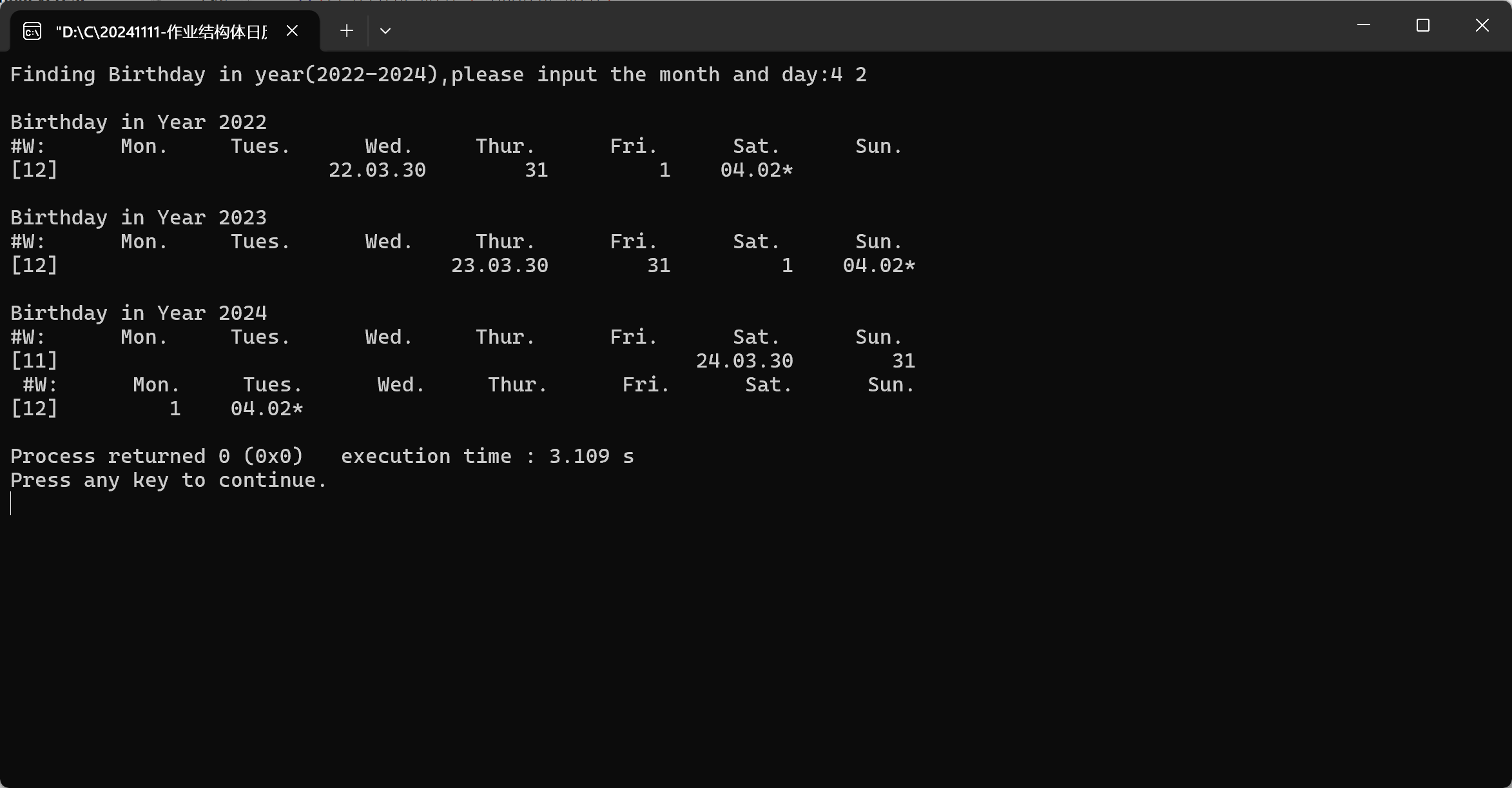
成功跨年且年份表示出来

### 测试点 2.29 的测试结果

### 

在非闰年可以显示未找到2.29

### 测试点 跨月 的测试结果



可以跨月，且每个月的日期正确

## 实验结论

代码能达到目的

## 实验总结

作为日历组的最后一个实验，用上了结构体，可以相当于是把上一个实验的三维数组用结构体来表示，简化了函数。

# 本课程学习总结

在上c语言之前，我所接触的编程语言还是上小学时的scratch，可以说是0基础学习c语言。我到现在还记得第一节课时，看着课件上的英文和一堆计算机原理，什么都不懂是多么的绝望。做第一次的日历作业，我还记得是上b站搜索“c语言日历”，翻看各个视频，结果是一堆看不懂的符号，有的用数组，有的用switch，但当我问了其他同学，竟然是反复的if，当时的我认为这十分冗杂，还在想要是都要这样的话，计算机算还不如我人脑算呢！但其实直到现在，我才能了解到，那一长串的if正是计算机的思维方式，就是看似是反复冗杂，其实是严谨有序。打一个不大恰当的比方，就像是收到的国庆放假通知一样：



当时只觉得这个通知有点莫名其妙，其实可以直接说调休，但是它把每一天的安排都一一列出，看着杂乱，但其实十分严谨，计算机语言亦是如此。

接下来随着课程的推进，学到了更多知识，c语言的体系也在逐渐完整，虽然还有很多不清楚的地方，但是慢慢的也能自己写程序了。仍记得那些被c语言作业的阴影笼罩的夜晚，c语言无疑是我付出的时间最多的学科，仍记得在头歌上交作业是，总会有锁起来的题报错，那一个红色感叹号，预示着我在接下来的一个多小时我将一直死磕这个作业。但现在想来，不通过报错都是有理由的，都是由于代码有一定的问题，在正式敲代码的时候不会有人告诉你为什么错，所以这些锁起来的题目可能会更有意义。

至于我选择日历系列的原因，日历系列贯穿整个C语言学习，可以说是伴随我度过了这一个学期，从最简单的if语句，到最后的结构体，就是少了指针的内容，但是指针的内容我也可以再数组中运用，因此虽然老师说可以只写日历8之后的内容，我还是从1开始全部梳理了一遍，这也是我脑中，C语言体系建立起来的过程。在日历11中，老师为我们介绍了这个系列经历了大约5个阶段，从面向过程到面向数据，再到面向对象，计算机是服务于用户的，因此我们的程序应该最大程度的方便用户，引入类似宏定义来方便更改，从而最大方便我们的使用。

计算机的优势就在于它可以精确计算，并做反复的工作也无妨，但是他是死脑筋，这就要我们来进行更严谨的编码，我现在所学到的只是一些皮毛，我也没办法保证我能一次性写出正确程序，但是我们可以试错，在试错的过程中学习，才能有所进步。这也要感谢我的老师，挑选出适合该阶段的题目使我的编程能力得到锻炼，虽然写的真的很痛苦，但是不可否认的是这确实是最好的提升方式。

我与计算机的缘分不会到此为止，与C语言的也是，在今后的学习、工作中我必将以此为基础，学习更多知识，希望在以后的学习中，我能很好地运用C语言这个工具！

# 附录

完整实验代码附在此处

1. Calendar1.计算星期几
2. main.c

#include <stdio.h>

#include <stdlib.h>

/\*calculate what day is it today\*/

int main()

{

int sum=0;

printf("We had known that 2024.01.01 was Monday.\n");

printf("please input month and day:\n");

int month;/\*first number to be in put by user\*/

int day;/\*second number to be in put by user\*/

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

printf("month = %d,day = %d\n",month,day);

if(month < 1 || month > 12)

{

printf("error!");

}

if(month == 1)

{

sum = 0;

}

if(month == 2)

{

sum = 31;

}

if(month == 3)

{

sum = 60;

}

if(month == 4)

{

sum = 91;

}

if(month == 5)

{

sum = 121;

}

if(month == 6)

{

sum = 152;

}

if(month == 7)

{

sum = 162;

}

if(month == 8)

{

sum = 213;

}

if(month == 9)

{

sum = 244;

}

if(month == 10)

{

sum = 274;

}

if(month == 11)

{

sum = 305;

}

if(month == 12)

{

sum = 335;

}

sum=sum+day;

int result = sum % 7;

if(result == 0)

{

result = 7;

}

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

return 0;

}

1. Calendar2.打印月历
2. main.c

#include <stdio.h>

#include <stdlib.h>

int main()

{

int month,day = 0;

printf("Please input a month(1-12):\n");

scanf("%d",&month);

if(month>12||month<1){

printf("please input corret month number.\n");

return 0;

}

//先判断这个月第一天是周几，该月多少天

int daysinmonth[12]={31,29,31,30,31,30,31,31,30,31,30,31};

printf("this month has %d days.\n\n",daysinmonth[month-1]);

int n=0;

while(n < month - 1){

n ++;

day += daysinmonth[n - 1];

}

//printf("%d\n",day);

day = day % 7 + 1;

if(day > 7){

day = day - 7;

}

//打印日历表头

printf("%10s%10s%10s%10s%10s%10s%10s\n",

"Mon.", "Tues.", "Wend.", "Thur.","Fri", "Stat.", "Sun.");

int i=0;

while(i < day - 1){

i ++;

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

}

//开始打印日历

int date = 0;

while(date < daysinmonth[n]){

date++;

printf("%10d",date);

//考虑换行

if((date+day-1)%7==0){

printf("\n");

}

}

//printf("%d",day);

return 0;

}

1. Calendar3.打印周历

#include <stdio.h>

#include <stdlib.h>

//21年周历

int main(void)

{

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

int num,month=0,counter;

//char Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday;

printf("Input week sequence(1~53):");

scanf("%d",&num);

if(num<1||num>53)

{

printf("error");

}

int day=num\*7-10;

printf("No.%d week in 2021\n",num);

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

switch(num){

case 1:printf("%38d.%-2d%7d.%-2d%7d.%-2d",1,1,1,2,1,3);break;

case 53:printf(" 12.30 12.31");break;

default:while(day > monthday[month])

{

day -= monthday[month];

month++;

}

for(counter=1;counter<=7;counter++,day++)

{

if(day==monthday[month]+1)

{

month++,day=1;

}

printf("%7d.%2d",month+1,day);

}

}

}

1. Calendar5.打印秋季校历

#include <stdio.h>

#include <stdlib.h>

// Functions about month and day

int getMonthLength (int month);

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()

{

// Calculate the first day and last day in the spring semester

int referenceDay;

printf("fall semester calender of year 2021, please input reference day in Feb(1-28):\n");

scanf("%d",&referenceDay);

if(referenceDay > 28 || referenceDay < 1)

{

printf("")

}

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

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

// Navigate every day in this semester

int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;

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

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

daySeqOfYear <= sEndSeqOfYear;

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

{

// Before Monday, print the week sequence

if ( daySeqOfWeek == 0 )

{

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

}

//打印校历周次

// Print the month and day for the current day

printOneDay( daySeqOfYear ) ;

// After Sunday, print a new line调用函数打印-天

if ( daySeqOfWeek == 6 )

{

printf("\n");

weekSeqOfSemester ++;

}

}

return 0;

}

int getMonthLength(int month)

{

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

return lengths[month];

}

// 计算一年中某一天的序列号（从1月1日开始）

int getDaySeq(int month, int day)

{

int daySeq = 0;

int i;

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

{

daySeq += getMonthLength(i);

}

daySeq += day;

return daySeq;

}

// 根据一年中某一天的序列号，计算它是哪个月

int getMonth(int daySeqOfYear)

{

int month;

for(month = 1;daySeqOfYear > getMonthLength(month);month ++)

{

daySeqOfYear -= getMonthLength(month);

}

return month;

}

int getDay(int daySeqOfYear)

{

int month;

for(month = 1;daySeqOfYear > getMonthLength(month);month ++)

{

daySeqOfYear -= getMonthLength(month);

}

return daySeqOfYear;

}

int getDaySeqOfWeek(int daySeqOfYear)

{

return ((daySeqOfYear + 5 - 1) % 7 + 1);

}

int getNextMonday(int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(daySeqOfYear);

return daySeqOfYear + (8 - dayOfWeek) + 1;

}

int getThisSunday(int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(daySeqOfYear);

return daySeqOfYear + (7 - dayOfWeek) % 7;

}

// 打印一天的信息（月份和日期）

void printOneDay(int daySeqOfYear)

{

int month = getMonth(daySeqOfYear);

int day = getDay(daySeqOfYear);

int dayOfWeek = getDaySeqOfWeek(daySeqOfYear);

if((day == 1)&&((dayOfWeek != 6)&&(dayOfWeek != 7)))

{

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

}

else if((day == 1)&&((dayOfWeek == 6)||(dayOfWeek == 7)))

{

printf(" %02d-%02d!", month, day);

}

else if((day != 1)&&((dayOfWeek == 6)||(dayOfWeek == 7)))

{

printf(" %02d!",day);

}

else

{

printf(" %02d",day);

}

}

1. Calendar6.打印春季校历

#include <stdio.h>

#include <stdlib.h>

int isLeapYear(int year);

int getDaySeqOnJan1(int year);

// Functions about month and day

int getMonthLength (int year,int month);

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 main()

{

// Calculate the first day and last day in the spring semester

int referenceDay;

printf("spring semester calender of year 2021, please input reference day in Jan(1-31):\n");

scanf("%d",&referenceDay);

if(referenceDay > 31 || referenceDay < 1)

{

printf("error!");

return 0;

}

int sStartSeqOfYear = getThisMonday( 2021,getDaySeq(2021,9,1) );

int sEndSeqOfYear = getThisSunday( 2022,getDaySeq(2022,1,referenceDay)) ;

// Navigate every day in this semester

int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;

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

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

daySeqOfYear <= sEndSeqOfYear + 365;

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

{

// Before Monday, print the week sequence

if ( daySeqOfWeek == 0 )

{

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

}

//打印校历周次

// Print the month and day for the current day

if(daySeqOfYear <= 365)

{

printOneDay( 2021,daySeqOfYear ) ;

}

else

{

printOneDay( 2022,daySeqOfYear - 365 ) ;

}

// After Sunday, print a new line调用函数打印-天

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 day = 1;

while (year > 2001)

{

day += (isLeapYear(year - 1) + 365);

year --;

}

int SeqOnJan1 = (day - 1) % 7 - 1;

return SeqOnJan1;

}

int getMonthLength(int year,int month)

{

if(isLeapYear(year) == 0)

{

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

return lengths[month];

}

if(isLeapYear(year) == 1)

{

int lengths[] = { 0,31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

return lengths[month];

}

}

// 计算一年中某一天的序列号（从1月1日开始）

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

{

int daySeq = 0;

int i;

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

{

daySeq += getMonthLength(year,i);

}

daySeq += day;

return daySeq;

}

// 根据一年中某一天的序列号，计算它是哪个月

int getMonth(int year,int daySeqOfYear)

{

int month;

for(month = 1;daySeqOfYear > getMonthLength(year,month);month ++)

{

daySeqOfYear -= getMonthLength(year,month);

}

return month;

}

int getDay(int year,int daySeqOfYear)

{

int month;

for(month = 1;daySeqOfYear > getMonthLength(year,month);month ++)

{

daySeqOfYear -= getMonthLength(year,month);

}

return daySeqOfYear;

}

int getDaySeqOfWeek(int year,int daySeqOfYear)

{

if(year ==2021)

{

return ((daySeqOfYear + 4 - 1) % 7 + 1);

}

else

{

return((daySeqOfYear + 5 - 1) % 7 + 1);

}

}

int getThisMonday(int year,int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

return daySeqOfYear - dayOfWeek + 1;

}

int getThisSunday(int year,int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

return daySeqOfYear + (7 - dayOfWeek) % 7;

}

// 打印一天的信息（月份和日期）

void printOneDay(int year,int daySeqOfYear)

{

int month = getMonth(year,daySeqOfYear);

int day = getDay(year,daySeqOfYear);

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

if((day == 1)&&((dayOfWeek != 7)&&(dayOfWeek != 6)))

{

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

}

else if((day == 1)&&((dayOfWeek == 7)||(dayOfWeek == 6)))

{

printf(" %02d-%02d!", month, day);

}

else if((day != 1)&&((dayOfWeek == 7)||(dayOfWeek == 6)))

{

printf(" %02d!",day);

}

else

{

printf(" %02d",day);

}

}

1. Calendar7.打印全年校历

#include <stdio.h>

#include <stdlib.h>

int isLeapYear(int year);

int getDaySeqOnJan1(int year);

// Functions about month and day

int getMonthLength (int year,int month);

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 main()

{

// Calculate the first day and last day in the spring semester

int inputYear = 2021;

printf("%13s%s%d%s%d\n"," ","First semester (Fall) calendar of year ",inputYear,"-",inputYear + 1);

int sStartSeqOfYear1 = getThisMonday( 2021,getDaySeq(2021,9,4) );

int sEndSeqOfYear1 = getThisSunday( 2022,getDaySeq(2022,1,20)) ;

// Navigate every day in this semester

int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;

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

for ( daySeqOfYear = sStartSeqOfYear1, daySeqOfWeek =0, weekSeqOfSemester = 1;

daySeqOfYear <= sEndSeqOfYear1 + 365;

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

{

// Before Monday, print the week sequence

if ( daySeqOfWeek == 0 )

{

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

}

//打印校历周次

// Print the month and day for the current day

if(daySeqOfYear <= 365)

{

if(daySeqOfYear == sStartSeqOfYear1)

{

printf(" %d-%02d-%02d",(inputYear % 100),getMonth(inputYear,daySeqOfYear),getDay(inputYear,daySeqOfYear));

}

else

{

printOneDay( 2021,daySeqOfYear ) ;

}

}

else

{

if((daySeqOfYear - 365) == 1)

{

printf(" %d-%02d-%02d",((inputYear + 1) % 100),getMonth(inputYear + 1,daySeqOfYear - 365),getDay(inputYear + 1,daySeqOfYear - 365));

}

else

{

printOneDay( 2022,daySeqOfYear - 365 ) ;

}

}

// After Sunday, print a new line调用函数打印-天

if ( daySeqOfWeek == 6 )

{

printf("\n");

weekSeqOfSemester ++;

}

}

printf("\n%13s%s%d%s%d\n"," ","Second semester (Spring) calendar of year ",inputYear,"-",inputYear + 1);

int sStartSeqOfYear2 = getThisMonday( 2022,getDaySeq(2022,2,15) );

int sEndSeqOfYear2 = getThisSunday( 2022,getDaySeq(2022,7,1)) ;

// Navigate every day in this semester

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

for ( daySeqOfYear = sStartSeqOfYear2, daySeqOfWeek =0, weekSeqOfSemester = 1;

daySeqOfYear <= sEndSeqOfYear2;

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

{

// Before Monday, print the week sequence

if ( daySeqOfWeek == 0 )

{

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

}

//打印校历周次

// Print the month and day for the current day

if(daySeqOfYear <= 365)

{

if(daySeqOfYear == sStartSeqOfYear2)

{

printf(" %d-%02d-%02d",((inputYear + 1) % 100),getMonth(inputYear,daySeqOfYear),getDay(inputYear,daySeqOfYear));

}

else

{

printOneDay( 2022,daySeqOfYear ) ;

}

}

else

{

if((daySeqOfYear - 365) == 1)

{

printf(" %d-%02d-%02d",((inputYear + 1) % 100),getMonth(inputYear + 1,daySeqOfYear - 365),getDay(inputYear + 1,daySeqOfYear - 365));

}

else

{

printOneDay( 2022,daySeqOfYear ) ;

}

}

// After Sunday, print a new line调用函数打印-天

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 day = 1;

while (year > 2001)

{

day += (isLeapYear(year - 1) + 365);

year --;

}

int SeqOnJan1 = (day - 1) % 7 - 1;

return SeqOnJan1;

}

int getMonthLength(int year,int month)

{

if(isLeapYear(year) == 0)

{

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

return lengths[month];

}

if(isLeapYear(year) == 1)

{

int lengths[] = { 0,31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

return lengths[month];

}

}

// 计算一年中某一天的序列号（从1月1日开始）

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

{

int daySeq = 0;

int i;

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

{

daySeq += getMonthLength(year,i);

}

daySeq += day;

return daySeq;

}

// 根据一年中某一天的序列号，计算它是哪个月

int getMonth(int year,int daySeqOfYear)

{

int month;

for(month = 1;daySeqOfYear > getMonthLength(year,month);month ++)

{

daySeqOfYear -= getMonthLength(year,month);

}

return month;

}

int getDay(int year,int daySeqOfYear)

{

int month;

for(month = 1;daySeqOfYear > getMonthLength(year,month);month ++)

{

daySeqOfYear -= getMonthLength(year,month);

}

return daySeqOfYear;

}

int getDaySeqOfWeek(int year,int daySeqOfYear)

{

return ((daySeqOfYear + getDaySeqOnJan1(year)) % 7 + 1);

}

int getThisMonday(int year,int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

return daySeqOfYear - dayOfWeek + 1;

}

int getThisSunday(int year,int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

return daySeqOfYear + (7 - dayOfWeek) % 7;

}

// 打印一天的信息（月份和日期）

void printOneDay(int year,int daySeqOfYear)

{

int month = getMonth(year,daySeqOfYear);

int day = getDay(year,daySeqOfYear);

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

if((day == 1)&&((dayOfWeek != 7)&&(dayOfWeek != 6)))

{

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

}

else if((day == 1)&&((dayOfWeek == 7)||(dayOfWeek == 6)))

{

printf(" %02d-%02d!", month, day);

}

else if((day != 1)&&((dayOfWeek == 7)||(dayOfWeek == 6)))

{

printf(" %02d!",day);

}

else

{

printf(" %02d",day);

}

}

1. Calendar8.月份数组

#include <stdio.h>

#include <stdlib.h>

#define YEAR\_NUM 5

#define YEAR\_MIN 2000

#define YEAR\_MAX 2030

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] = {2020,2021,2022,2023,2024};

//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 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);

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

void setYearArray(int yearArray[],int yearNum,int yearStart);

int main()

{

// Declare variables

int inputYear,Years[YEAR\_NUM];

// Display the program information

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\_NUM > YEAR\_MAX )

{

printf("Sorry, the input year is not supported. \n");

return 1;

}

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 spring Calendar of Year ", Years[i]);

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

printOneWeek(Years[i], getWeekSeqOfYear ( Years[i], 9, 4) , 1);

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"," ","Earlist 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 day = 1;

while (year > 2001)

{

day += (isLeapYear(year - 1) + 365);

year --;

}

int SeqOnJan1 = (day - 1) % 7 - 1;

return SeqOnJan1;

}

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

{

int daySeq = day;

int i;

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

{

if(isLeapYear(year) == 1)

{

daySeq += monthDayInLeapYear[i];

}

else

{

daySeq += monthDayInNormalYear[i];

}

}

return daySeq;

}

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

{

return (getDaySeq( year , month , day ) + getDaySeqOnJan1(year) - 2) / 7 + 1;

}

int getMonth(int year, int daySeqOfYear)

{

int month\_num;

if(isLeapYear(year) == 1)

{

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

{

daySeqOfYear -= monthDayInLeapYear[month\_num];

}

}

else if(isLeapYear(year) == 0)

{

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

{

daySeqOfYear -= monthDayInNormalYear[month\_num];

}

}

return month\_num;

}

int getDay(int year,int daySeqOfYear)

{

int month\_num;

if(isLeapYear(year) == 1)

{

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

{

daySeqOfYear -= monthDayInLeapYear[month\_num];

}

}

else if(isLeapYear(year) == 0)

{

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

{

daySeqOfYear -= monthDayInNormalYear[month\_num];

}

}

return daySeqOfYear;

}

int getDaySeqOfWeek(int year, int daySeqOfYear)

{

int Seq;

Seq = (getDaySeqOnJan1(year) + daySeqOfYear) % 7;

return Seq;

}

int getThisMonday(int year,int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

return daySeqOfYear - dayOfWeek + 1;

}

int getThisSunday(int year,int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

return daySeqOfYear + (7 - dayOfWeek) % 7;

}

// 打印一天的信息（月份和日期）

void printOneDay(int year,int daySeqOfYear)

{

int month = getMonth(year,daySeqOfYear);

int day = getDay(year,daySeqOfYear);

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

if((day == 1)&&((dayOfWeek != 7)&&(dayOfWeek != 6)))

{

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

}

else if((day == 1)&&((dayOfWeek == 7)||(dayOfWeek == 6)))

{

printf(" %02d-%02d!", month, day);

}

else if((day != 1)&&((dayOfWeek == 7)||(dayOfWeek == 6)))

{

printf(" %02d!",day);

}

else

{

printf(" %02d",day);

}

}

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

{

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

int mondaySeq = (weekSeqOfYear - 1) \* 7 - getDaySeqOnJan1(year);

int sundaySeq = mondaySeq + 6;

int seq,day,month;

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

{

month = getMonth(year,seq);

day = getDay(year,seq);

if(seq == mondaySeq)

{

printf(" %d-%02d-%02d",year % 100,month,day);

}

else if(day == 1)

{

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

}

else

{

printOneDay(year,seq);

}

}

}

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

{

int i;

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

{

yearArray[i] = yearStart + i;

}

}

1. Calendar9.日期数组

#include <stdio.h>

#include <stdlib.h>

#define YEAR\_NUM 5

#define YEAR\_MIN 2021

#define YEAR\_MAX 2025

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] = {2020,2021,2022,2023,2024};

//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 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);

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

void setYearArray(int yearArray[],int yearNum,int yearStart);

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

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

int main()

{

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

int weekendNum = 0;

int i;

for(i = YEAR\_MIN; i <= YEAR\_MAX;i ++)

{

int weekSeqOfYear = getWeekSeqOfYear(i,inputMonth,inputDay);

int daySeqOfYear = getDaySeq(i,inputMonth,inputDay);

printf("\n%s%s%d\n", " ", "Birthday in Year " , i );

if(isDay(i,inputMonth,inputDay) == 0)

{

printf("Not found!");

}

else

{

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

printOneWeek(i,weekSeqOfYear,weekSeqOfYear);

totalNum ++;

if((getDaySeqOfWeek(i,daySeqOfYear) == 5 )||(getDaySeqOfWeek(i,daySeqOfYear) == 6))

{

weekendNum ++;

}

}

printf("%d\n",weekendNum);

}

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

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 day = 1;

while (year > 2001)

{

day += (isLeapYear(year - 1) + 365);

year --;

}

int SeqOnJan1 = (day - 1) % 7 - 1;

return SeqOnJan1;

}

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

{

int daySeq = day;

int i;

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

{

if(isLeapYear(year) == 1)

{

daySeq += monthDayInLeapYear[i];

}

else

{

daySeq += monthDayInNormalYear[i];

}

}

return daySeq;

}

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

{

return (getDaySeq( year , month , day ) + getDaySeqOnJan1(year) - 2) / 7 + 1;

}

int getMonth(int year, int daySeqOfYear)

{

int month\_num;

if(isLeapYear(year) == 1)

{

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

{

daySeqOfYear -= monthDayInLeapYear[month\_num];

}

}

else if(isLeapYear(year) == 0)

{

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

{

daySeqOfYear -= monthDayInNormalYear[month\_num];

}

}

return month\_num;

}

int getDay(int year,int daySeqOfYear)

{

int month\_num;

if(isLeapYear(year) == 1)

{

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

{

daySeqOfYear -= monthDayInLeapYear[month\_num];

}

}

else if(isLeapYear(year) == 0)

{

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

{

daySeqOfYear -= monthDayInNormalYear[month\_num];

}

}

return daySeqOfYear;

}

int getDaySeqOfWeek(int year, int daySeqOfYear)

{

int Seq;

Seq = (getDaySeqOnJan1(year) + daySeqOfYear) % 7;

return Seq;

}

int getThisMonday(int year,int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

return daySeqOfYear - dayOfWeek + 1;

}

int getThisSunday(int year,int daySeqOfYear)

{

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

return daySeqOfYear + (7 - dayOfWeek) % 7;

}

// 打印一天的信息（月份和日期）

void printOneDay(int year,int daySeqOfYear)

{

int month = getMonth(year,daySeqOfYear);

int day = getDay(year,daySeqOfYear);

int dayOfWeek = getDaySeqOfWeek(year,daySeqOfYear);

if((day == 1)&&((dayOfWeek != 5)&&(dayOfWeek != 6)))

{

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

}

else if((day == 1)&&((dayOfWeek == 5)||(dayOfWeek == 6)))

{

printf(" %02d-%02d!", month, day);

}

else if((day != 1)&&((dayOfWeek == 5)||(dayOfWeek == 6)))

{

printf(" %02d!",day);

}

else

{

printf(" %02d",day);

}

}

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

{

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

int mondaySeq = (weekSeqOfYear - 1) \* 7 - getDaySeqOnJan1(year);

int sundaySeq = mondaySeq + 6;

int seq,day,month;

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

{

printOneDay(year,seq);

}

}

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

{

int i;

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

{

yearArray[i] = yearStart + 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] = getDaySeqOfWeek (year, j + 1 );

}

}

return;

}

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;

}

}

1. 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()

{

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:");

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 day = 1;

while (year > 2001)

{

day += (isLeapYear(year - 1) + 365);

year --;

}

int SeqOnJan1 = (day - 1) % 7 - 1;

return SeqOnJan1;

}

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

{

int daySeq = day;

int i;

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

{

if(isLeapYear(year) == 1)

{

daySeq += monthDayInLeapYear[i];

}

else

{

daySeq += monthDayInNormalYear[i];

}

}

return daySeq;

}

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

{

return (getDaySeq( year , month , day ) + getDaySeqOnJan1(year) - 2) / 7 + 1;

}

int getMonth(int year, int daySeqOfYear)

{

int month\_num;

if(isLeapYear(year) == 1)

{

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

{

daySeqOfYear -= monthDayInLeapYear[month\_num];

}

}

else if(isLeapYear(year) == 0)

{

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

{

daySeqOfYear -= monthDayInNormalYear[month\_num];

}

}

return month\_num;

}

int getDay(int year,int daySeqOfYear)

{

int month\_num;

if(isLeapYear(year) == 1)

{

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

{

daySeqOfYear -= monthDayInLeapYear[month\_num];

}

}

else if(isLeapYear(year) == 0)

{

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

{

daySeqOfYear -= monthDayInNormalYear[month\_num];

}

}

return daySeqOfYear;

}

int getDaySeqOfWeek(int year, int daySeqOfYear)

{

int Seq;

Seq = (getDaySeqOnJan1(year) + daySeqOfYear) % 7;

return Seq;

}

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;

}