

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 | 王富临 | 学号 | U202413454 |

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 2025.1.15 | 地点 | 华中科技大学 |

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

# 实验目的

# 完成日历系列代码

# 实验环境

操作系统：Windows 10

编程工具：CodeBlocks 16.01

# 实验一（贯穿实例-日历2-打印月历）

**3.1 实验任务**

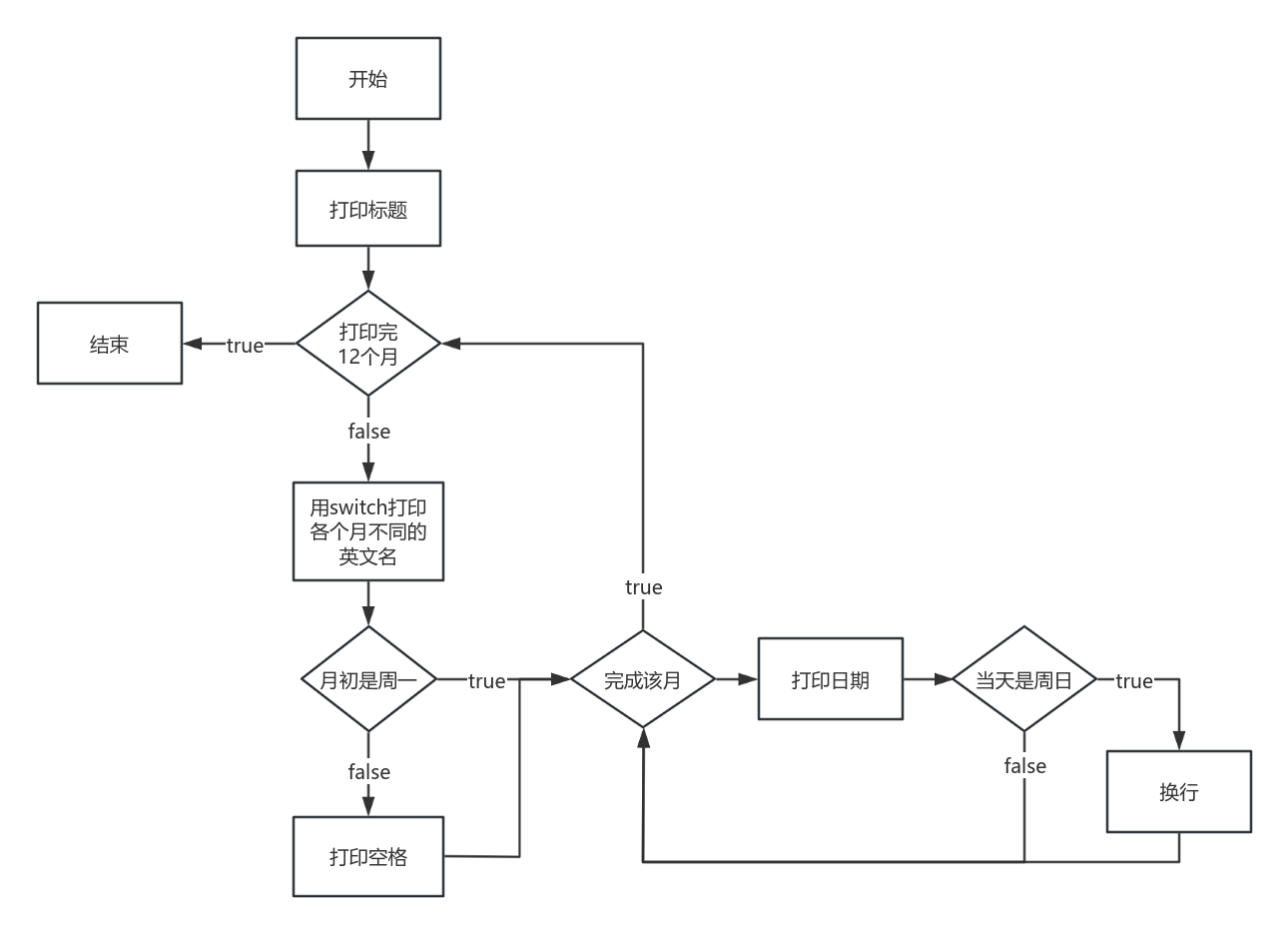
学习使用循环、条件等控制语句、算术表达式，完成简单的功能。运用所学到的知识，编写打印月历的程序，打印2021年1到12月的某个月的月历（已知2021年1月1日周五）

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，如果碰到周日要换行

**3.2 实验步骤**



**3.3 代码测试**

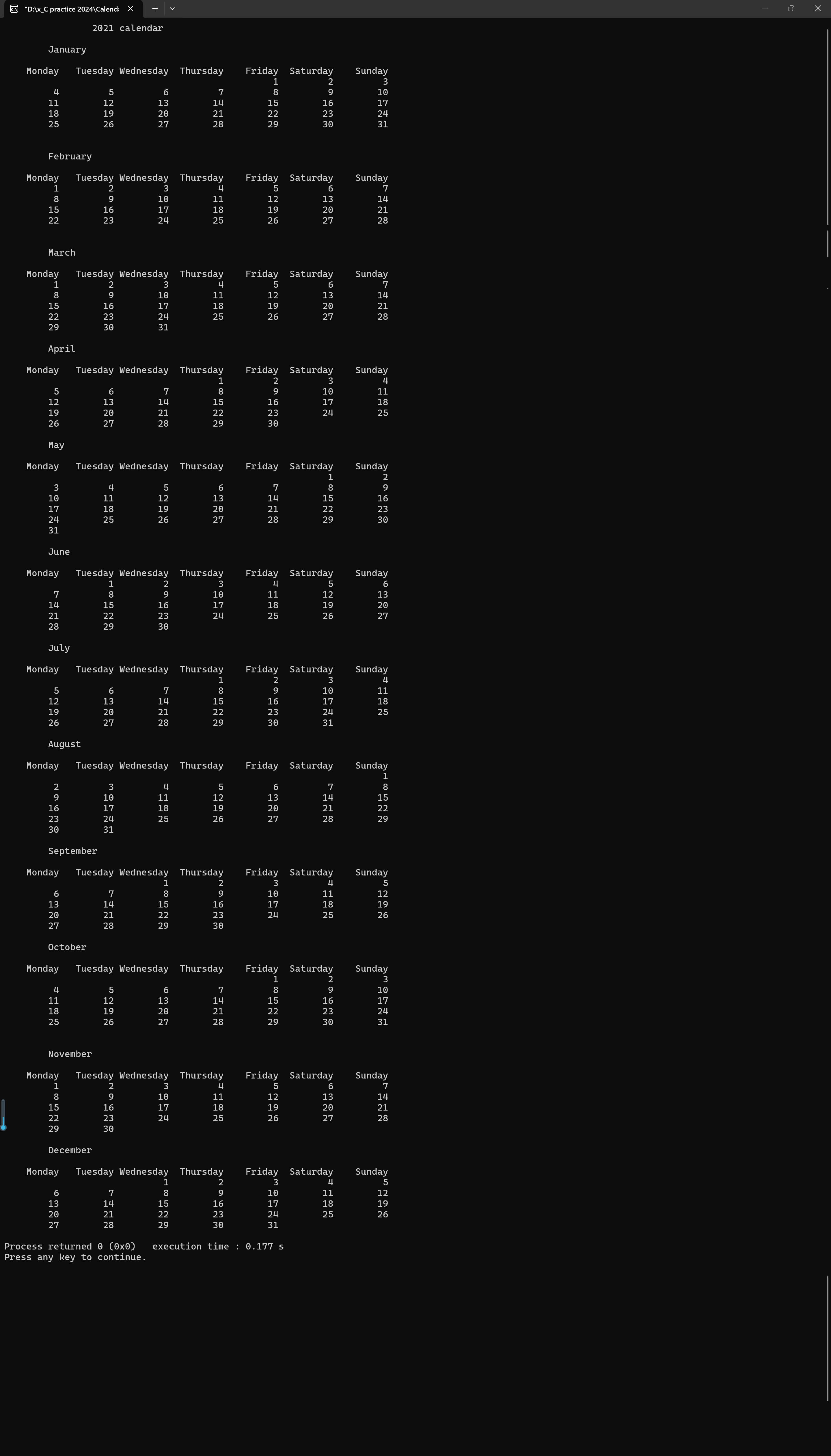
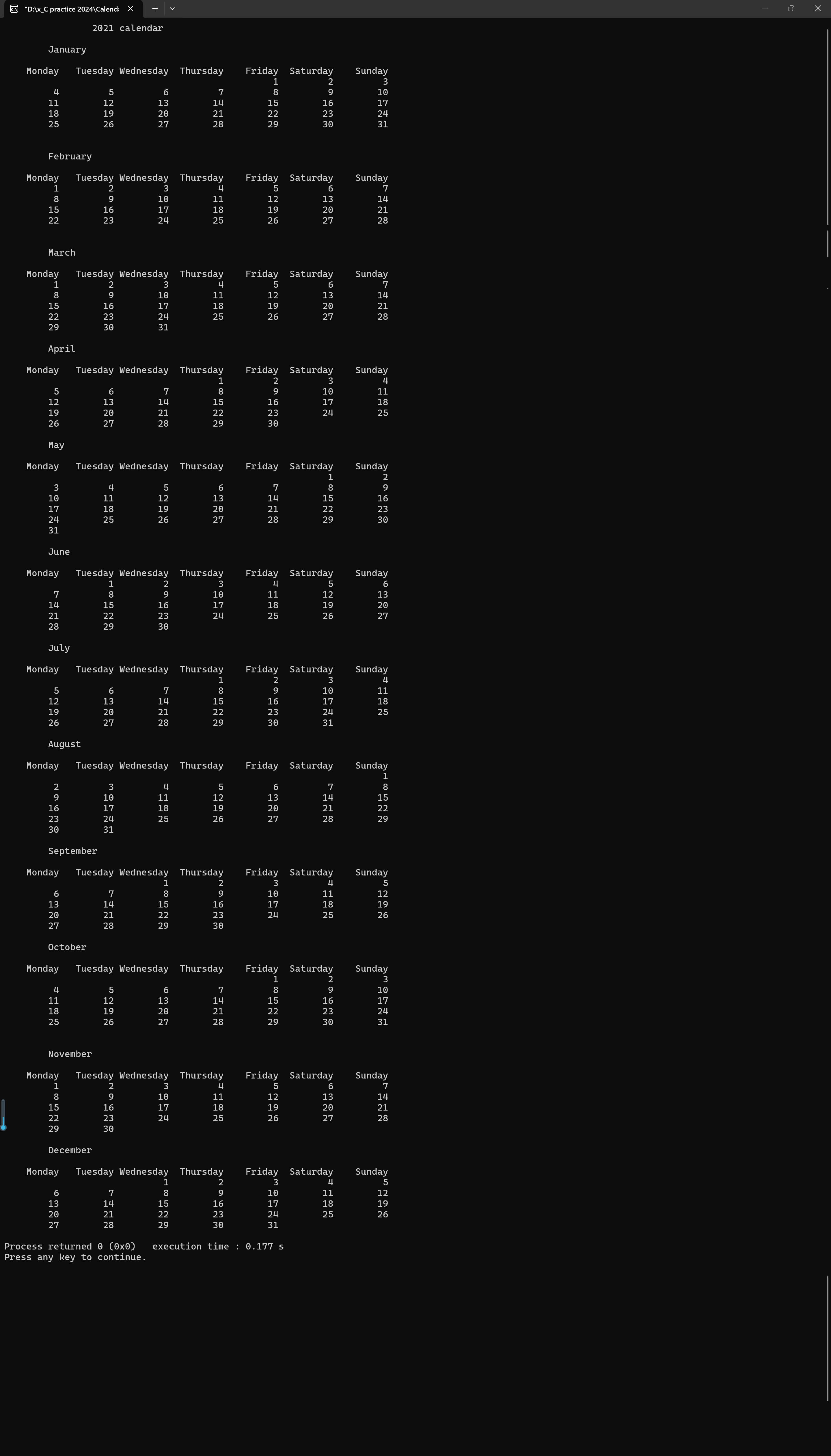
测试步骤：

该代码比较简单，直接运行，与标准日历比较。

预期测试结果：

以正确格式打印2021年日历。

实际测试结果：



测试结论：

日历正确。

## 3.4 实验结论

代码达到功能目标

# 实验二（贯穿实例-日历9-日期数组）

## 实验任务

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。

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。

## 组织结构图(2)实验步骤

**是平年2.29**

## 代码测试

### 4.3.1 测试点一（测试isLeapYear函数）

测试步骤：

int year1, year2;

scanf("%d，%d", &year1, &yaer2);

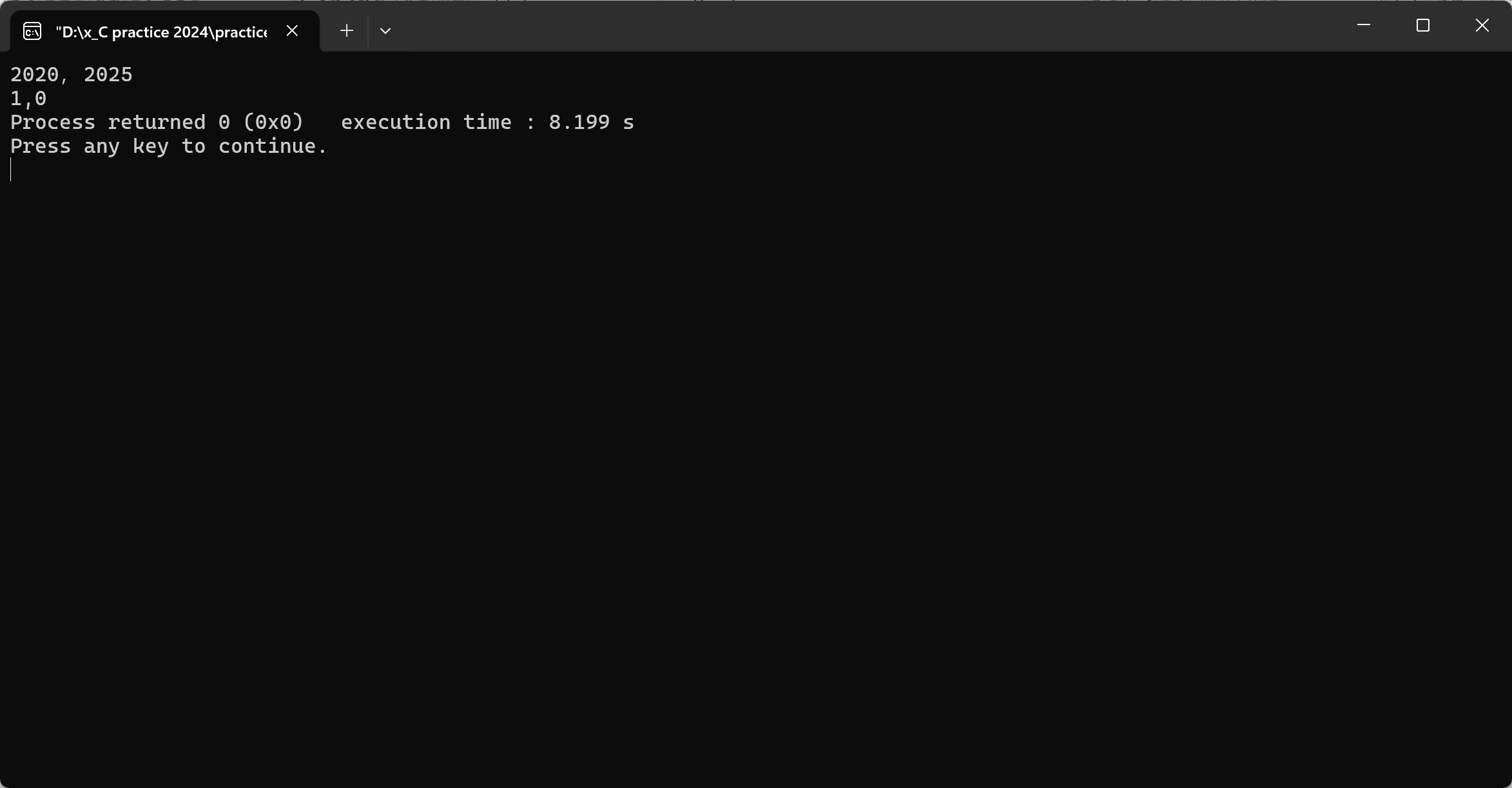
printf("%d, %d", isLeapYear(year1), isLeapYear(year2));

输入“2020，2025”

预期测试结果：

闰年和平年，分别返回1和0

实际测试结果：



测试结论：

isLeapYear能正确判断平年和闰年。

3.3.2 测试点二（测试程序对特殊日期的处理）

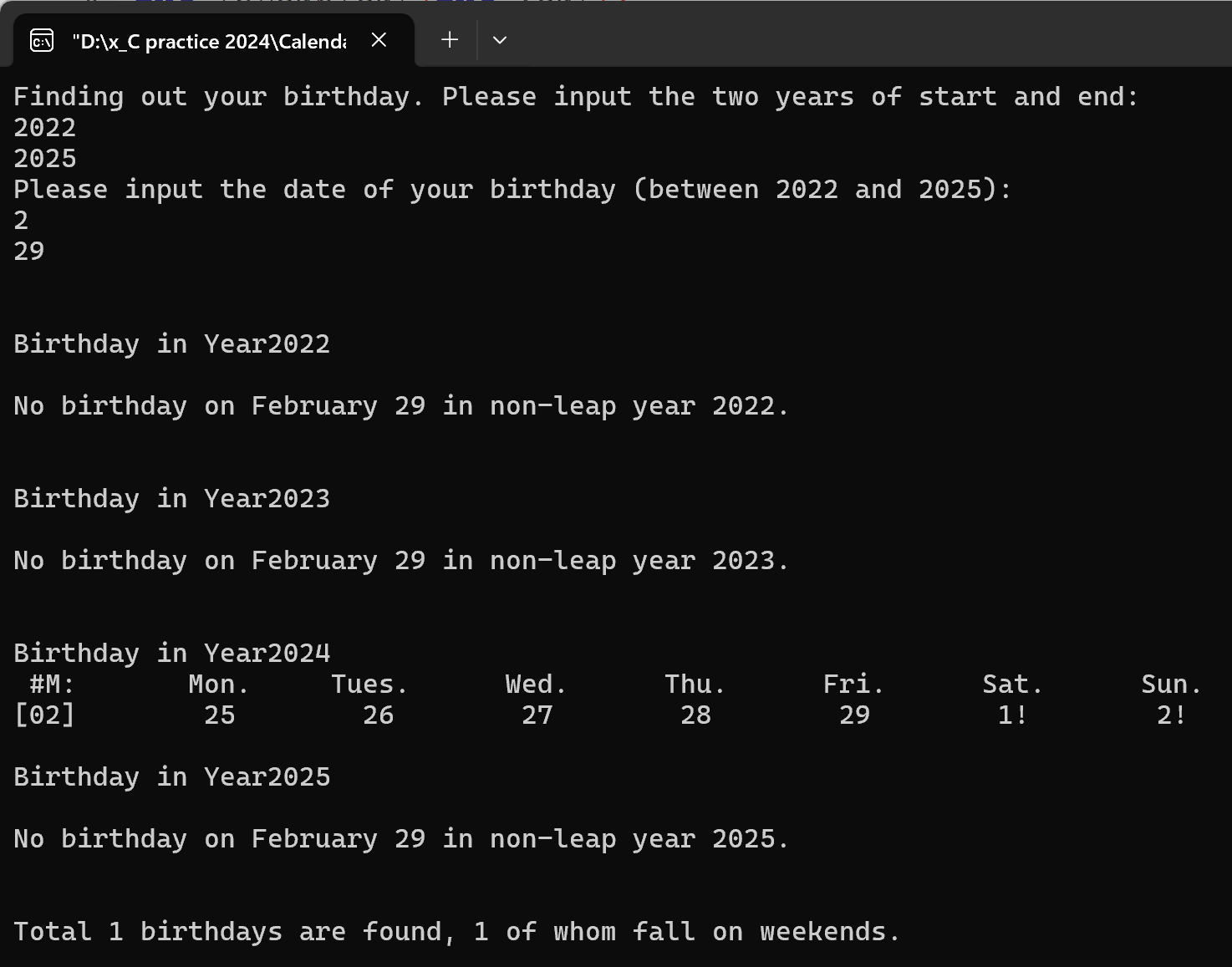
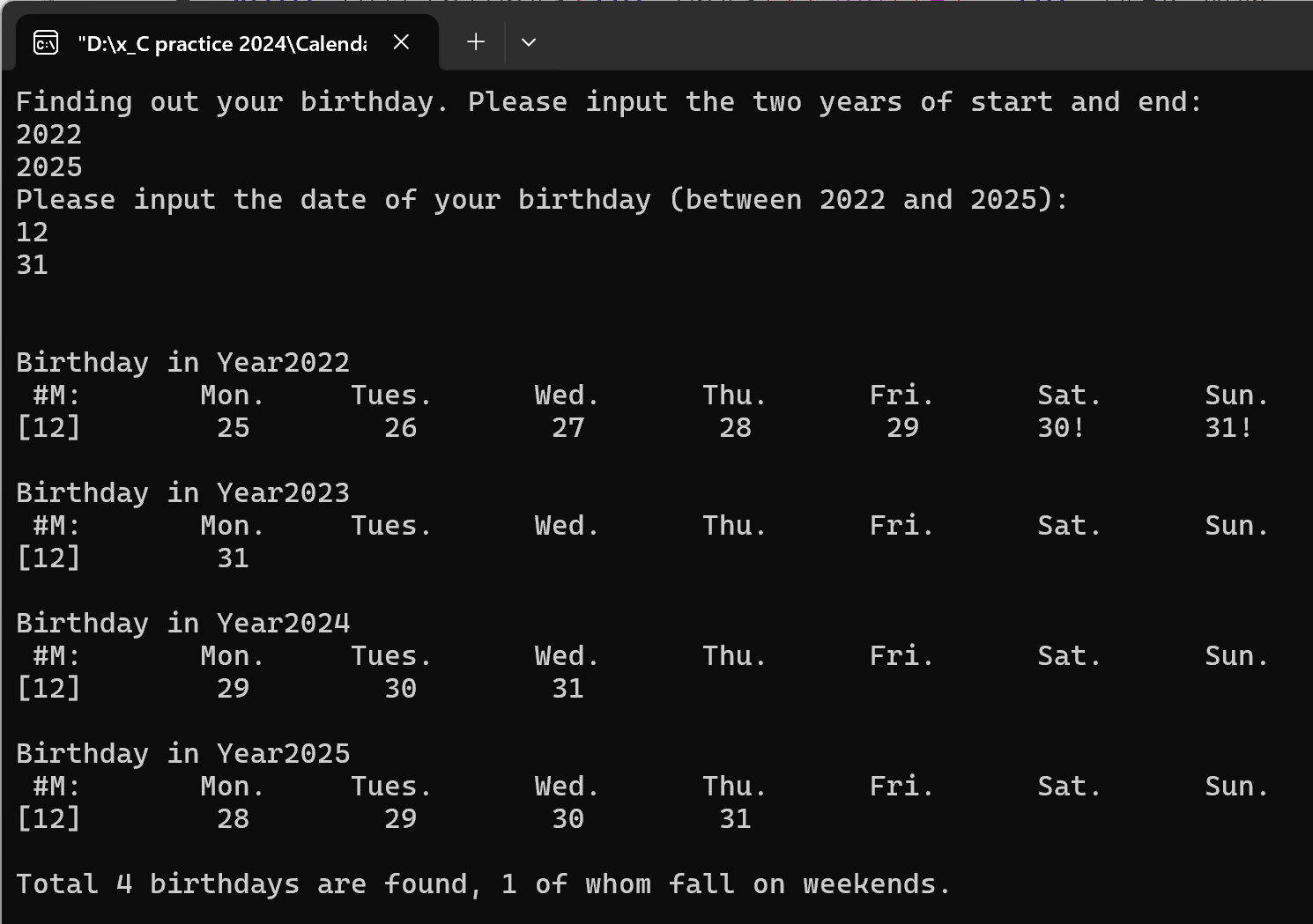
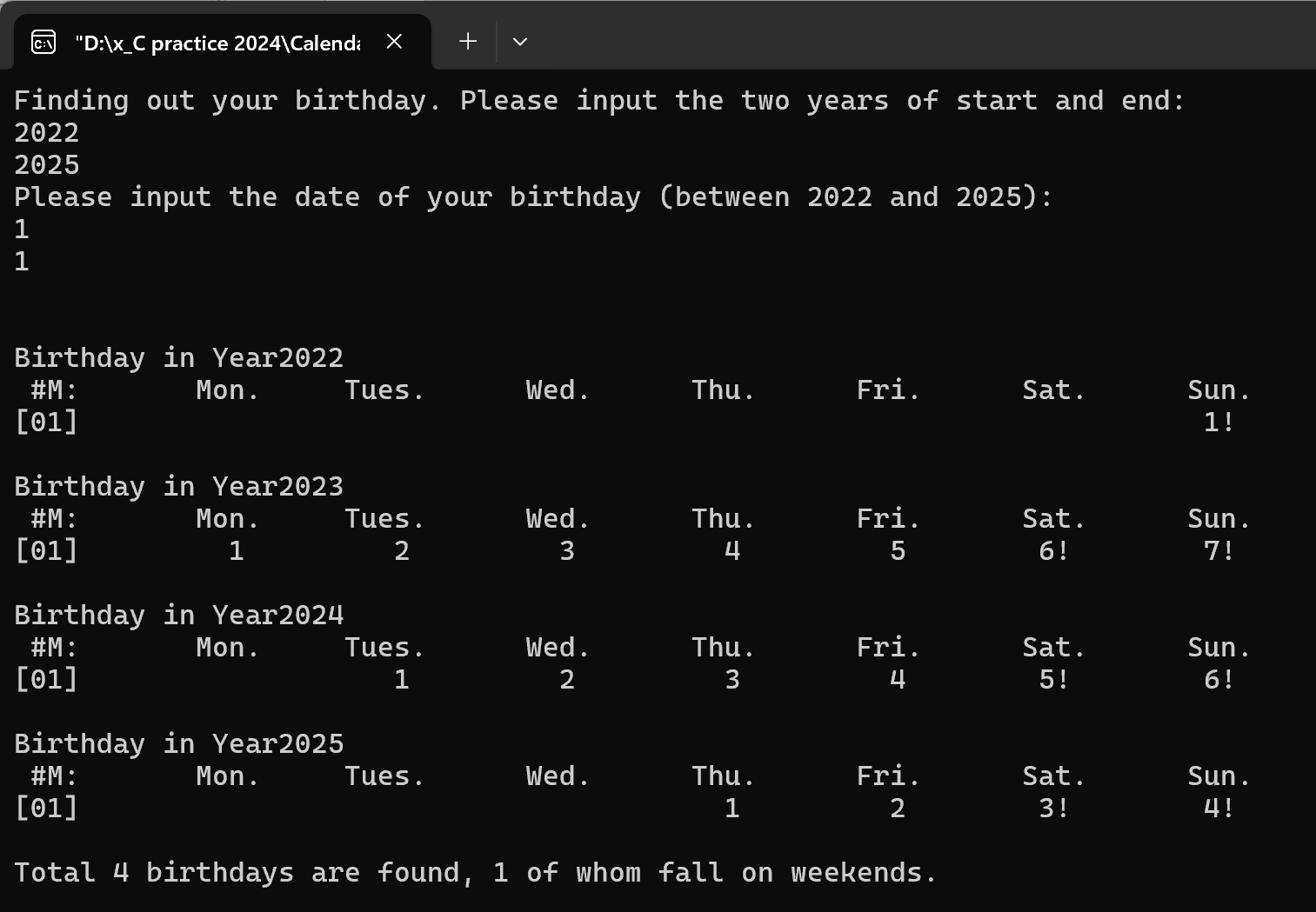
测试步骤：

在2022~2025这四年，分别输入1.1，12.31，2.29三个特殊日期

预期测试结果：

非本年的日期呈现为空格；平年2.29打印未找到

实际测试结果：



测试结论：

该程序能正确处理特殊日期。

## 实验结论

代码达到功能目标

## 实验总结

void initialDays(int Days[][366][4], int YEAR\_NUM, int YEAR\_MIN);

void printfOneWeek(int year, int j, int Days[][366][4]);

int isLeapYear(int year);

int getMonth(int year, int day);

int getDay(int year, int day);

int getweekSeqofYear(int year, int day);

int getDaySeqofweek(int year, int day);

int getJan1WeekSeq(int year);

使用的函数过多，降低了效率，如getMonth和getDay，大部分内容是一样的。应当把使用同一功能模块的函数组合，使代码更简洁高效。或者利用指针，在函数间传递信息，避免重复计算。总之，此代码有较大的改进空间。

# 实验三 （贯穿实例-日历10-日期结构体）

## 实验任务

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。

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。开发和改造下列函数：

// 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);

int getWeekSeqOfYear(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 support multiple year

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

// Functions support struct Day

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);

void printDay(Day currentDay, int displayFormat);

## 实验报告-组织结构图-图片实验步骤

## 代码测试

### 5.3.1 测试点一（测试getDay, getDayBefore, getDayAfter三个函数的效果）

测试步骤：

Day d1 = setDay(2005, 12, 25);

printf("%d.%d.%d\n",d1.year, d1.month, d1.day);

Day d2 = getDayBefore(d1, 3);

printf("%d.%d.%d\n",d2.year, d2.month, d2.day);

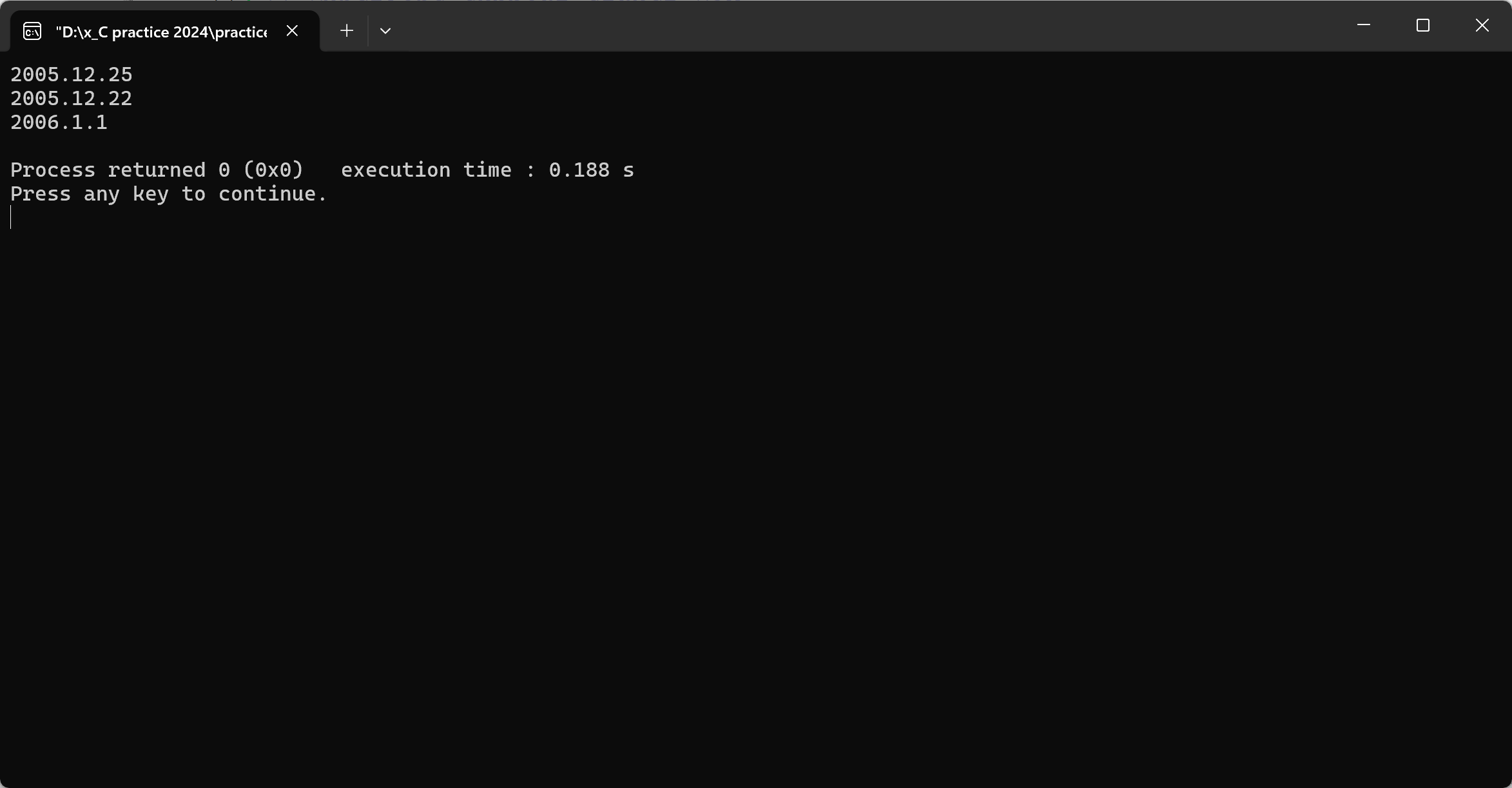
Day d3 = getDayAfter(d1,7);

printf("%d.%d.%d\n",d3.year, d3.month, d3.day);

预期测试结果：

正确计算、打印三个日期

实际测试结果：



测试结论：

getDay, getDayBefore, getDayAfter三个函数运行正常，甚至能处理跨年的情况。

5.3.2 测试点二（测试不同的打印格式）

测试步骤：

enum{DATE\_NOSHOW,DATE\_INFO\_FULL,DATE\_INFO\_BRIEF,DATE\_STAR};

Day d = setDay(2005, 12, 25);

printDay(d, DATE\_NOSHOW);

printf("\tDATE\_NOSHOW\n");

printDay(d, DATE\_INFO\_FULL);

printf("\tDATE\_INFO\_FULL\n");

printDay(d, DATE\_INFO\_BRIEF);

printf("\tDATE\_INFO\_BRIEF\n");

printDay(d, DATE\_STAR);

printf("\tDATE\_STAR\n");

预期测试结果：

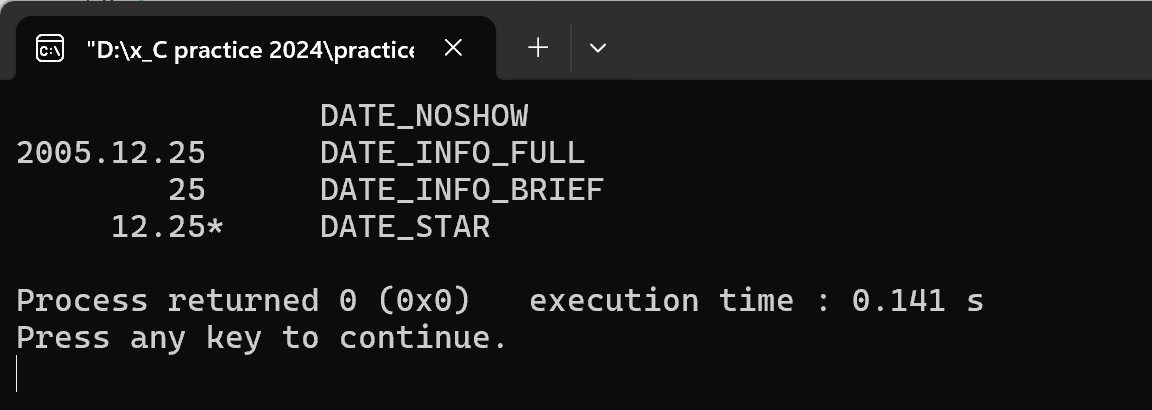
DATE\_NOSHOW

2005.12.25 DATE\_INFO\_FULL

25 DATE\_INFO\_BRIEF

12.25\* DATE\_STAR

实际测试结果：



测试结论：

四个打印格式均正确。

## 实验结论

代码达到功能目标

## 实验总结

日历系列圆满完成！

# 本课程学习总结

我在上大学之前从没有接触过C语言，并且老师讲的也很快，因此一开始我觉得它很难，无从下手。但是慢慢的，我熬过来了。通过大量的作业，在动辄数个小时的煎熬中，我逐渐找到了C语言的风格，并且能赶上老师的节奏了。陌生的事物才是最可怕的，在熟悉之后，我反而能从编程中找到乐趣——逻辑的趣味。编程就是：痛，并快乐着。

我最开始的代码没有任何“多余”的空格，绝对紧凑。后来我按照标准格式修改，形成了紧凑为主，必要处空格的风格，看起来更加美观，算是一个进步。

用编译器自带的debug功能并不方便，所以我一般会用printf来检测。调试时可以将部分代码在另一个文件里测试，这样不会影响原文件。

**附录**

1. **贯穿实例-日历2-打印月历**

#include<stdio.h>

int weekdayseqOf1th(int month)

{

int Month[12] = {31,28,31,30,31,30,31,31,30,31,30,31};

int i = 1, x = 5;

for (;i < month; i++){

x += Month[i - 1];

}

return (x - 1) % 7 + 1;

}

int main()

{

printf("\t\t2021 calendar\n");

int i, j;

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

switch(i){

case 0:

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

break;

case 1:

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

break;

case 2:

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

break;

case 3:

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

break;

case 4:

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

break;

case 5:

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

break;

case 6:

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

break;

case 7:

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

break;

case 8:

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

break;

case 9:

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

break;

case 10:

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

break;

case 11:

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

break;

}

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

"Monday","Tuesday","Wednesday","Thursday","Friday","Saturday","Sunday");

int Month[12] = {31,28,31,30,31,30,31,31,30,31,30,31};

int weekdayseq = weekdayseqOf1th(i + 1);

if (weekdayseq != 1)

printf("%\*s", (weekdayseq - 1) \* 10, " ");//ÔÂ³õ´òÓ¡¿Õ¸ñ

for (j = 0; j < Month[i]; j++){

printf("%10d", j + 1);

if ((j + weekdayseq) % 7 == 0){

printf("\n");

}

}

printf("\n");

}

return 0;

}

1. **贯穿实例-日历9-日期数组**

#include <stdio.h>

#include <stdlib.h>

void initialDays(int Days[][366][4], int YEAR\_NUM, int YEAR\_MIN);

void printfOneWeek(int year, int j, int Days[][366][4]);

int isLeapYear(int year);

int getMonth(int year, int day);

int getDay(int year, int day);

int getweekSeqofYear(int year, int day);

int getDaySeqofweek(int year, int day);

int getJan1WeekSeq(int year);

int main()

{

int YEAR\_MAX, YEAR\_MIN, inputMonth, inputDay;

printf("Finding out your birthday. Please input the two years of start and end:\n");

scanf("%d%d", &YEAR\_MIN, &YEAR\_MAX);

printf("Please input the date of your birthday (between %d and %d):\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 1;

}

const int YEAR\_NUM = YEAR\_MAX - YEAR\_MIN + 1;

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

initialDays(Days, YEAR\_NUM, YEAR\_MIN);

int totalNum = 0, weekendNum = 0, i, j;

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

for (j = 0; j <= 365 + isLeapYear(i + YEAR\_MIN); j++){

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

break;

}

}//确定生日为第j天

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

if (!isLeapYear(i + YEAR\_MIN) && inputMonth == 2 && inputDay == 29) {

printf("\nNo birthday on February 29 in non-leap year %d.\n", YEAR\_MIN + i);

} else {

printf(" #M: %10s%10s%10s%10s%10s%10s%10s\n",

"Mon.", "Tues.", "Wed.", "Thu.", "Fri.", "Sat.", "Sun.");

printf("[%02d]", Days[i][j][0]);

printfOneWeek(i, j, Days);//为减少输入printfOneWeek的变量，将部分打印内容移至main

totalNum++;

}

if (Days[i][j][3] >= 6) {

weekendNum++;

}

}

printf("\n\nTotal %d birthdays are found, %d of whom fall on weekends.\n", totalNum, weekendNum);

return 0;

}

void initialDays(int Days[][366][4], int YEAR\_NUM, int YEAR\_MIN) {

int year, yearLength;

int i, j;

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

year = i + YEAR\_MIN;

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, j + 1);

Days[i][j][3] = getDaySeqofweek(year, j + 1);

}

}

}

void printfOneWeek(int year, int j, int Days[][366][4]) {

int x = Days[year][0][3]; //x是当年第一天周几

int weekseq = Days[year][j][3], i = 1;

if (Days[year][j][2] == 1 && x != 1){

printf("%\*c", (x - 1) \* 10, ' ');

i += x - 1;

}

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

if (Days[year][j - weekseq +i][3] < 6){

printf("%10d", Days[year][j - weekseq +i][1]);

} else {

printf("%9d!", Days[year][j - weekseq +i][1]);

}

if (Days[year][j - weekseq +i][1] == 31 && Days[year][j - weekseq +i][0] == 12) {

break;

}

}

}

int isLeapYear(int year) {

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

return 1;

else

return 0;

}

int getMonth(int year, int day) {

int i = 1;

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

month[2] += isLeapYear(year);

while (day > month[i]) {

day -= month[i];

i = (i % 12) + 1;

}

return i;

}

int getDay(int year, int day) {

int i = 1;

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

month[2] += isLeapYear(year);

while (day > month[i]) {

day -= month[i];

i++;

}

return day;

}

int getweekSeqofYear(int year, int day) {

return (day + getJan1WeekSeq(year) - 1) / 7 + ((day + getJan1WeekSeq(year) - 1) % 7 == 0 ? 0 : 1);

}

int getDaySeqofweek(int year, int day) {

int weekday = (day + getJan1WeekSeq(year) - 1) % 7;

return weekday == 0 ? 7 : weekday;

}

int getJan1WeekSeq(int year) {

return ((year - 1) + (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400 + 1) % 7 + 1;

}

1. **贯穿实例-日历10-日期结构体**

#include <stdio.h>

#include <stdlib.h>

#define YEAR\_NUM 5

typedef struct{

int year;

int month;

int day;

int daySeq;

int weekSeq;

int weekDay;

}Day;

Day birthday,prepareDay,currentDay;

// 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);

int getWeekSeqOfYear(int year, int daySeqOfYear);

// 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 support multiple year

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

// Functions support struct Day

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 subtract(Day startDay, Day endDay);//求两天之差（前减后）

void printDay(Day currentDay, int displayFormat);

int main()

{

int inputMonth,inputDay,interval;

printf("Finding birthday in year 2020~2024."

"Please input your birthday: (month.day)\n");

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

printf("How many days do you need to prepare?"

" Please input the interval:\n");

scanf("%d",&interval);

int Years[YEAR\_NUM]={2020,2021,2022,2023,2024};

int displayFormat,i;

enum{DATE\_NOSHOW,DATE\_INFO\_FULL,DATE\_INFO\_BRIEF,DATE\_STAR};

//便于利用switch打印不同格式

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

printf("\n%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, interval);

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

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

for(currentDay = getDayBefore(prepareDay, prepareDay.weekDay-1); //从周一开始

subtract(currentDay, birthday) <= 0; currentDay = getDayAfter(currentDay, 1)){

if (subtract(currentDay, prepareDay) < 0){

displayFormat = DATE\_NOSHOW;

} else if (subtract(currentDay, prepareDay) == 0){

displayFormat = DATE\_INFO\_FULL;

} else if (subtract(currentDay, birthday) < 0){

displayFormat = DATE\_INFO\_BRIEF;

} else {

displayFormat = DATE\_STAR;

}

printDay(currentDay, displayFormat);

}

printf("\n");

}

return 0;

}

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

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

monthLength[1] += isLeapYear(year);

if (year<0||month<=0||month>12||day<=0||day>monthLength[month-1]){

return -1;

}else{

return 1;

}

}

int isLeapYear(int year) {

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

}

int getDaySeqOnJan1(int year) {

return ((year-1)+(year-1)/4-(year-1)/100+(year-1)/400+1)%7+1;

}

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

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

monthLength[1] += isLeapYear(year);

int daySeq=0,i = 0;

for(; i < month-1 ; i++) {

daySeq += monthLength[i];

}

daySeq += day;

return daySeq;

}

int getWeekSeqOfYear(int year, int day){

int totalDays = day + getDaySeqOnJan1(year) - 1;

int weekDay = totalDays / 7;

int remainder = (totalDays%7 == 0)?0:1;

return weekDay + remainder;

}

int getMonth(int year,int day){

int i=1;

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

month[2]+=isLeapYear(year);

while(day>month[i]){

day-=month[i];

i++;

if(i==13){

i=1;

}

}

return i;

}

int getDay(int year,int day){

int i=1;

int month[13]={0,31,28,31,30,31,30,31,31,30,31,30,31}; //为使标号与实际月份一致

month[2]+=isLeapYear(year);

while(day>month[i]){

day-=month[i];

i++;

}

return day;

}

int getDaySeqOfWeek(int year,int day){

int weekday=(day+getDaySeqOnJan1(year)-1)%7;

return weekday==0?7:weekday;

}

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);

d.weekSeq = getWeekSeqOfYear(year,day);

d.weekDay = getDaySeqOfWeek(year,day);

return d;

}

Day getDayBefore(Day d, int interval){

if(interval>=d.daySeq){

d.year--;

d.month=12;

d.day=d.day-interval+31;

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

}else{

d.daySeq -= interval;

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

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

}

d.weekSeq = getWeekSeqOfYear(d.year,d.daySeq);

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

return d;

}

Day getDayAfter(Day d,int interval){

int yearLength=365+isLeapYear(d.year);

if (interval>yearLength-d.daySeq){

d.year++;

d.month=1;

d.day=interval+d.daySeq-yearLength;

d.daySeq=d.day;

}else{

d.daySeq += interval;

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

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

}

d.weekSeq = getWeekSeqOfYear(d.year,d.daySeq);

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

return d;

}

int subtract(Day d1, Day d2){

int i,dayNum=0;

for(i=0;d1.year+i<d2.year;i++){

dayNum-=365+isLeapYear(d1.year+i);

}

for(i=0;d1.year>d2.year+i;i++){

dayNum+=365+isLeapYear(d2.year+i);

}

dayNum+=d1.daySeq-d2.daySeq;

return dayNum;

}

void printDay(Day d, int displayFormat) {

switch(displayFormat){

case 0:

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

break;

case 1:

printf("%4d.%2d.%2d",d.year,d.month,d.day);

break;

case 2:

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

break;

case 3:

printf("%7d.%2d\*",d.month,d.day);

}

if (d.weekDay == 7){

printf("\n");

}

}