

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 | 黄南楠 | 学号 | U202413685 |

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 2024.1.14 | 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

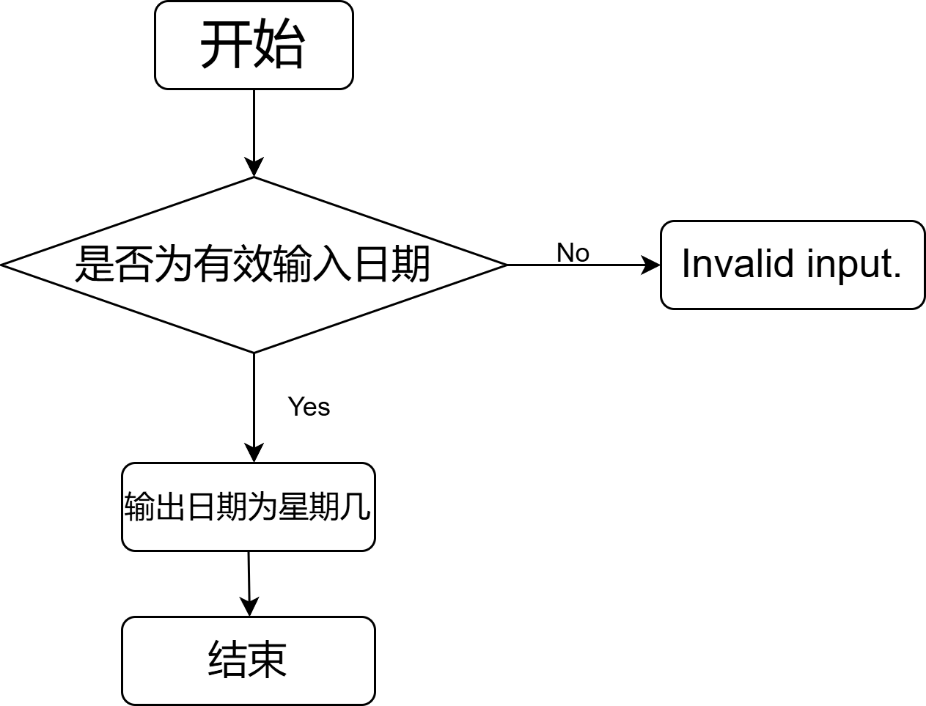
编程工具：CodeBlocks 16.01

# 实验一：计算星期几（Calendar01）

## 实验任务

计算2024年某月某日是星期几（已知2024-1-1是星期一）。

## 实验步骤



主函数部分：

1. #define current\_year 2024
2. #define weekday\_JAN1st 1
3. int main()
4. {
5. int month,day,dayseq,daynum;
6. printf("This year is 2024,please input your month(1-12) and day(1-31) and the DaySeqOfWeek will be presented(MM-DD):\n");
7. scanf("%d-%d",&month,&day);
8. if(month < 1 || month > 12 || day < 1 || day > 31){
9. printf("Invalid input.");
10. }
11. /\*
12. 此处省略计算总天数daynum代码（见附录源代码）
13. \*/
14. daynum = daynum % 7;
15. if(daynum == 0){
16. daynum = 7;
17. }
18. dayseq = weekday\_JAN1st + daynum - 1;
19. printf("%d-%d-%d is No.%d day in that week ",current\_year,month,day,dayseq);
20. return 0;
21. }

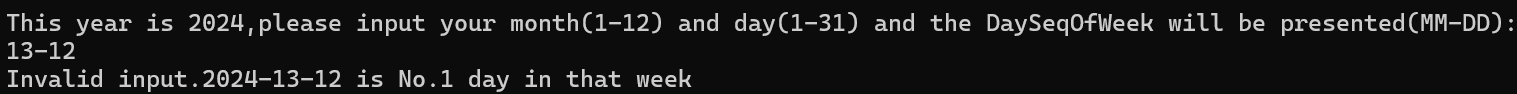
## 代码测试

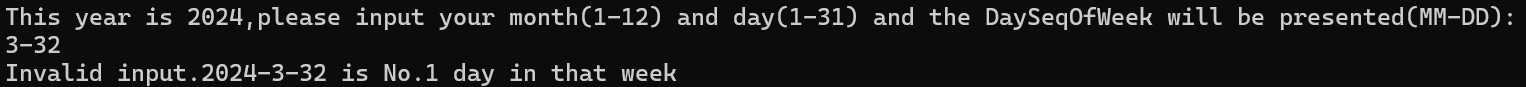
### 测试点1（无效输入）的测试结果

测试步骤：输入无效日期，观察输出情况

预期测试结果：输出“Invalid input”

实际输出结果（截图）：





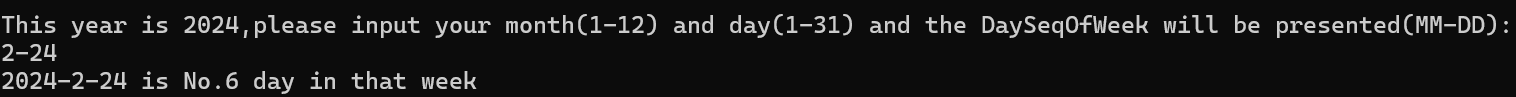
测试结论：正常输出报错信息。

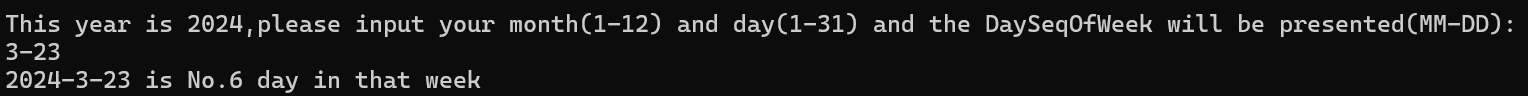
### 测试点2（正常输入）的测试结果

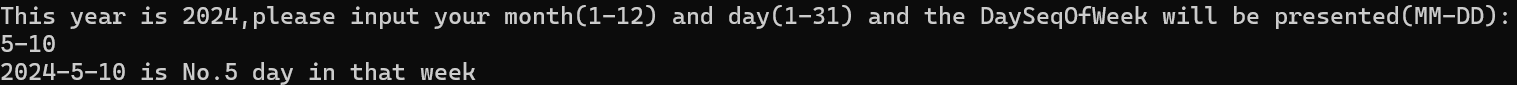
测试步骤：输入正常日期，观察输出情况

预期测试结果：输出该日期为星期几。

实际输出结果（截图）：







测试结论：正常输出该日期为星期几。

## 实验结论

代码达到功能目标。

## 实验总结

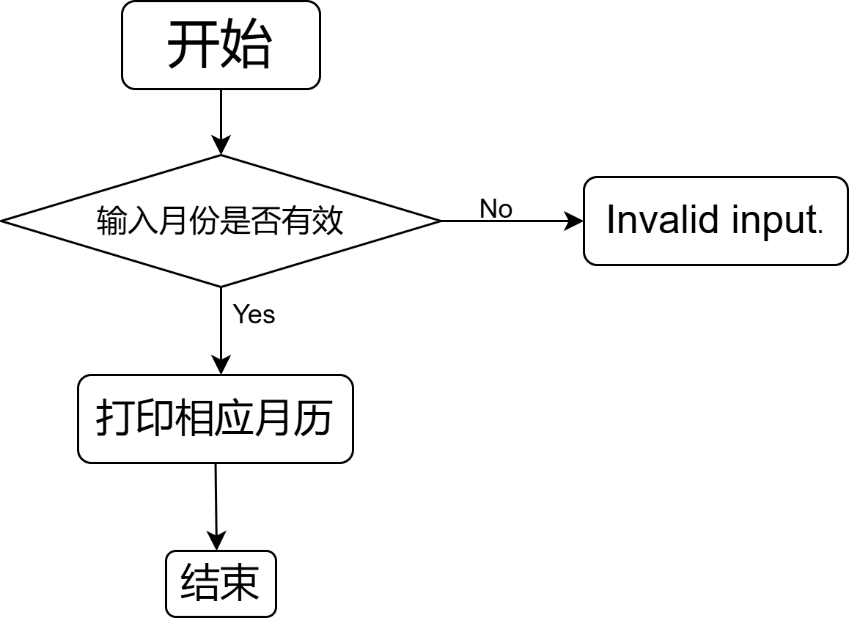
在实验中，虽然测试点1的给定输入能正确提示“Invalid input”，但实际上存在缺陷，比如输入6-31，输出并不会报错。在学习完数组后，缺陷可以得到改进。

# 实验二：打印月历（Calendar02）

## 实验任务

编写打印月历的程序，打印2024年1-12月中某月的月历（约定2024-1-1为周一）。

## 实验步骤



主函数部分：

1. #define current\_year 2024
2. #define weekday\_JAN1st 1
3. int main()
4. {
5. int month,day,dayseq,daynum;
6. printf("This year is 2024,please input your month(1-12) and day(1-31) and the DaySeqOfWeek will be presented(MM-DD):\n");
7. scanf("%d-%d",&month,&day);
8. if(month < 1 || month > 12 || day < 1 || day > 31){
9. printf("Invalid input.");
10. }
11. /\*
12. 此处省略计算总天数daynum代码（见附录源代码）
13. \*/
14. daynum = daynum % 7;
15. if(daynum == 0){
16. daynum = 7;
17. }
18. dayseq = weekday\_JAN1st + daynum - 1;
19. printf("%d-%d-%d is No.%d day in that week ",current\_year,month,day,dayseq);
20. return 0;
21. }

## 代码测试

### 测试点 1（无效月份）的测试结果

测试步骤：输入月份13，观察输出情况。

预期测试结果：输出“Please input a proper month!”

实际测试结果（截图）：



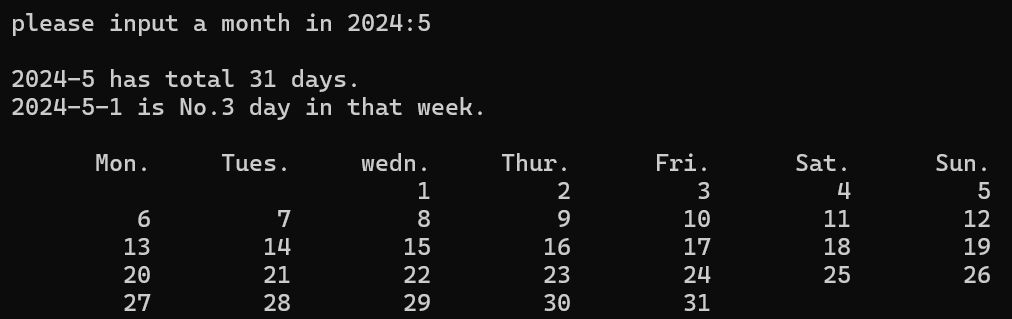
测试结论：正确提示报错信息

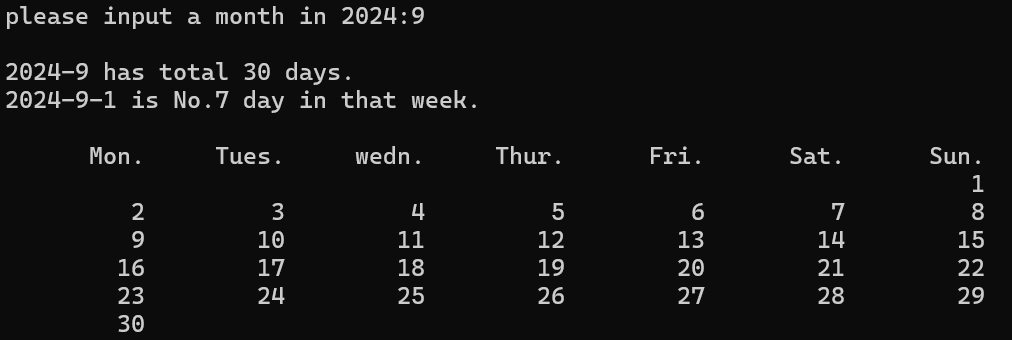
### 测试点2（正常月份）的测试结果

测试步骤：分别输入正常月份5和9，观察输出情况。

预期测试结果：打印相关月份的月历。

实际测试结果（截图）：





测试结论：正确输出该月份总天数，该月第一天为星期几，打印该月月历。

## 实验结论

代码达到功能目标。

## 实验总结

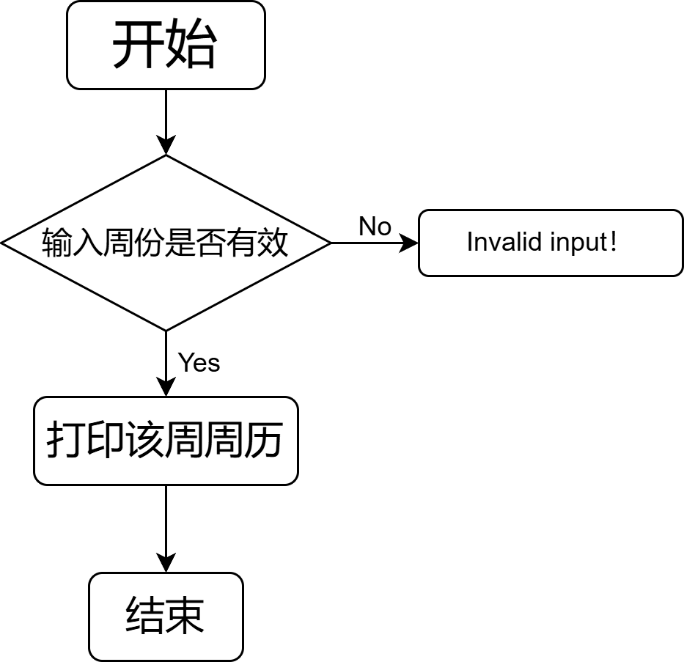
1. 代码中对于daynum（总天数）的过程还是很繁琐重复，在学习数组之后会得到改进。
2. 代码中，打印月历第一天时，先要计算进位，源代码中采用计算当月第一天为星期几，再用循环打印空格的方式进行处理。可以改进为printf(“%\*d”, ( date + dayseq1) \* 10)，简化打印空格的过程。

# 实验三：打印周历（Calendar03）

## 实验任务

编写一个打印周历的程序，打印2021年某一周的周历（2021-1-1为周四）。

## 实验步骤



主函数部分：

1. int main()
2. {
3. int WEEKDAY\_JAN1;
4. WEEKDAY\_JAN1 = (CURRENT\_YEAR + (CURRENT\_YEAR - 1) / 4 - (CURRENT\_YEAR - 1) / 100 + (CURRENT\_YEAR - 1) / 400) % 7;
5. int IS\_LEAPYEAR;
6. IS\_LEAPYEAR = CURRENT\_YEAR % 4;
7. if(IS\_LEAPYEAR != 0){
8. IS\_LEAPYEAR = 1;
9. }
10. int lengthJan = 31;
11. /\*
12. 此处省略length\*\*\*的定义（见附录源代码）
13. \*/
14. int weekSeqOfYear,month,day;
15. printf("Week Calandar of Year %d,please input week sequence(1-53): ",CURRENT\_YEAR);
16. scanf("%d",&weekSeqOfYear);
17. if(weekSeqOfYear < 1 || weekSeqOfYear > 53){
18. printf("Invalid input!");
19. return -1;
20. }
21. int dayfirst;
22. dayfirst = (weekSeqOfYear == 1)? 1 : (7 - WEEKDAY\_JAN1 + 1) + 1 + 7 \* (weekSeqOfYear - 2);
23. if(dayfirst <= lengthJan){
24. month = 1;
25. if(weekSeqOfYear == 1){
26. day = 1;
27. }else{
28. /\*
29. 此处省略其他情况对month和day的计算（见附录源代码）
30. \*/
31. }
32. printf("%10s%10s%10s%10s%10s%10s%10s\n","Mon.","Tues.","Wedn.","Thur.","Fri.","Sat.","Sun.");
33. int spacenum,count;
34. spacenum = (WEEKDAY\_JAN1 - 1) \* 10;
35. if(weekSeqOfYear == 1){
36. for(count = 1; count < spacenum ;count ++){
37. printf(" ");
38. }
39. for(spacenum ; spacenum < 70 ; spacenum = spacenum + 10){
40. printf("%7d.%2d",month,day);
41. day ++;
42. }
43. }
44. int Feb = (IS\_LEAPYEAR == 1)? 28 : 29;
45. int monthday[] = {0,31,Feb,31,30,31,30,31,31,30,31,30,31};
46. if(weekSeqOfYear > 1){
47. for(count = 1;count <= 7;count ++){
48. if(day == monthday[month] + 1){
49. month ++;
50. day = 1;
51. }
52. printf("%7d.%02d",month,day);
53. day ++;
54. if( month == 12 && day == 32){
55. break;
56. }
57. }
58. }
59. return 0;
60. }

## 代码测试

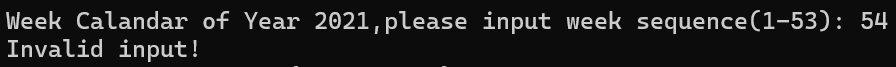
### 测试点1（无效输入）的测试结果

测试步骤：输入无效周份（如0，54），观察输出情况

预期测试结果：输出“Invalid input!”

实际测试结果（截图）：





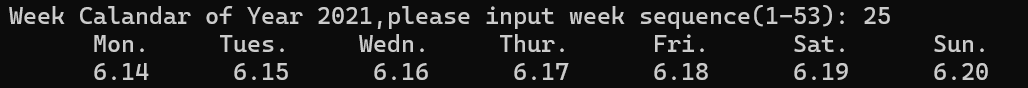
测试结论：正常提示报错信息。

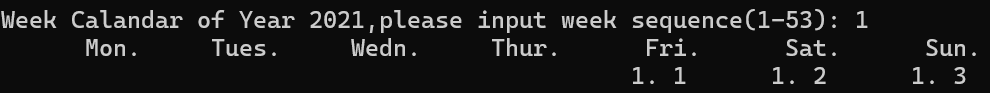
### 测试点2（正常输入）的测试结果

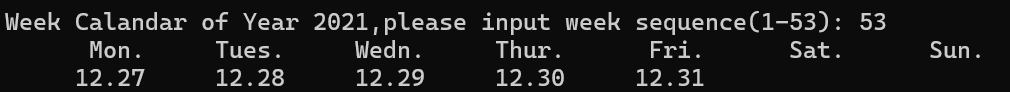
测试步骤：输入有效周份，观察输出情况

预期输出结果：正确打印该周周历

实际输出结果（截图）：







测试结论：正确打印输入周份的周历。

## 实验结论

代码达到功能目标

## 实验总结

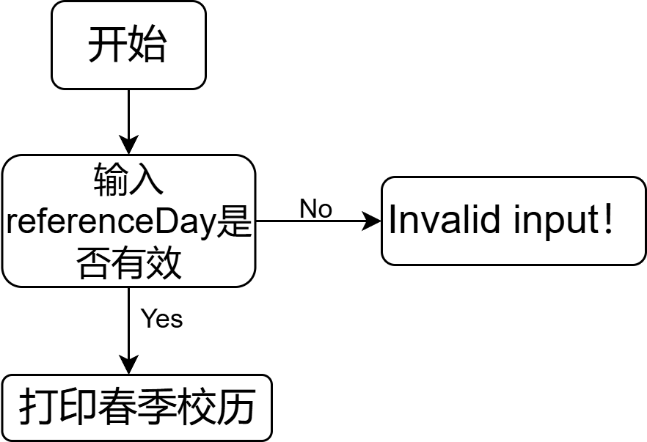
在源代码中，对于总天数的计算仍然重复繁杂，需要用数组循环/函数进行改进。

# 实验四：打印春季校历（Calendar04）

## 实验任务

编写一个打印校历的程序，打印2021年华中科技大学春季校历。

## 实验步骤



主函数部分：

1. int main()
2. {
3. int WEEKDAY\_JAN1;
4. WEEKDAY\_JAN1 = (CURRENT\_YEAR + (CURRENT\_YEAR - 1) / 4 - (CURRENT\_YEAR - 1) / 100 + (CURRENT\_YEAR - 1) / 400) % 7;
5. int IS\_LEAPYEAR;
6. IS\_LEAPYEAR = CURRENT\_YEAR % 4;
7. if(IS\_LEAPYEAR != 0){
8. IS\_LEAPYEAR = 1;
9. }
10. int lengthJan = 31;
11. int lengthFeb = (IS\_LEAPYEAR == 1)? lengthJan + 28 : lengthJan + 29;
12. //此处省略length\*\*\*的计算过程（见附录源代码）
13. int referenceDay,sumday,dayseq,day,month;
14. month = 2;
15. printf("Spring Semester Clandar of Year %d,\
16. please input reference day in Feb(1-%d):\n",CURRENT\_YEAR,lengthFeb - lengthJan);
17. scanf("%d",&referenceDay);
18. if(referenceDay < 1 || referenceDay > lengthFeb - lengthJan){
19. printf("Invalid Input!\n");
20. return 0;
21. }
22. printf("%18s%s%d\n"," ","Spring Semester Calandar of Year ",CURRENT\_YEAR);
23. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n","Mon.","Tues.","Wed.","Thur.","Fri.","Sat.","Sun.");
24. sumday = lengthJan + referenceDay;
25. dayseq = (sumday % 7 + WEEKDAY\_JAN1) % 7 ;
26. dayseq = (dayseq == 0)? 7 : dayseq;
27. day = referenceDay;
28. for(dayseq ; dayseq <= 7 ; dayseq ++){
29. day ++;
30. if(day >= lengthFeb - lengthJan){
31. day = 1;
32. month ++;
33. }
34. }
35. int length = lengthJun;
36. int endday = 0;
37. length = (length % 7 + WEEKDAY\_JAN1) % 7 ;
38. while(length <= 7){
39. length ++;
40. endday ++;
41. }
42. int Feb = (IS\_LEAPYEAR == 1)? 28 : 29;
43. int monthday[] = {0,31,Feb,31,30,31,30,31,31,30,31,30,31};
44. int count = 0;
45. int i = 0;
46. for(day; day <= monthday[month]; day ++){
47. //此处省略打印日历过程
48. if(day == monthday[month]){
49. day = 0;
50. month ++;
51. }
52. if(month == 7 && day == endday){
53. break;
54. }
55. }
56. return 0;
57. }

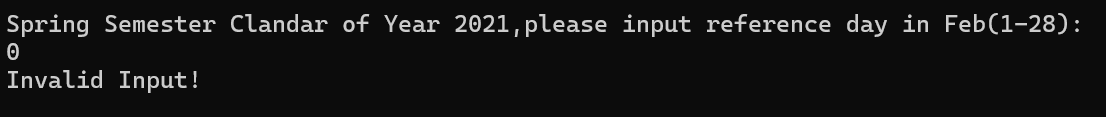
## 代码测试

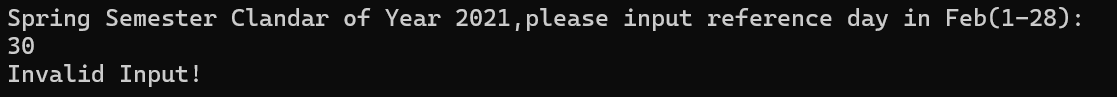
### 测试点1（无效输入）的测试结果

测试步骤：输入无效天数，观察输出情况

预期测试结果：输出“Invalid input!”

实际测试结果（截图）：





测试结论：正常提示报错信息。

### 测试点2（正常输入）的测试结果

测试步骤：输入有效天数，观察输出情况

预期输出结果：正确打印春季校历

实际输出结果（截图）：



测试结论：正确打印春季校历。

## 实验结论

代码达到功能目标。

## 实验总结

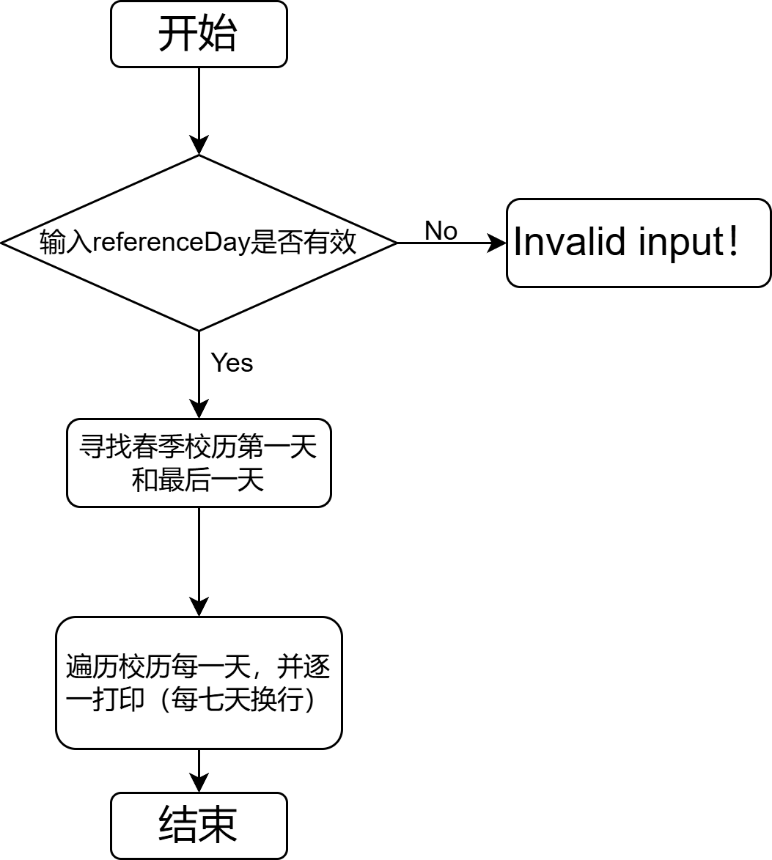
在源代码中，对于总天数的计算仍然重复繁杂，需要用数组循环改进。

# 实验五：打印春季校历【函数版】（Calendar05）

## 实验任务

用函数打印华中科技大学校历的程序，打印2021年春季学期的校历。

## 实验步骤



主函数部分：

1. int main()
2. {
3. int referenceDay;
4. printf("Spring Semester Calendar of Year 2021, please input referenceday in Feb(1-28):");
5. scanf("%d", &referenceDay);
6. if(referenceDay < 1 || referenceDay > 28){
7. printf("Invalid input!");
8. return -1；
9. }
10. int sStartSeqOfYear = getNextMonday( getDaySeq(2, referenceDay) );
11. int sEndSeqOfYear = getThisSunday( getDaySeq(7, 1) );
12. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
13. int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;
14. for( daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0,weekSeqOfSemester = 1;
15. daySeqOfYear <= sEndSeqOfYear;
16. daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7)
17. {
18. if( daySeqOfWeek == 0){
19. printf("[%02d]", weekSeqOfSemester );
20. }
21. printOneDay( daySeqOfYear );
22. if(daySeqOfWeek == 6){
23. printf("\n");
24. weekSeqOfSemester ++;
25. }
26. }
27. return 0;
28. }

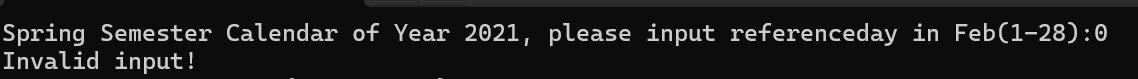
## 代码测试

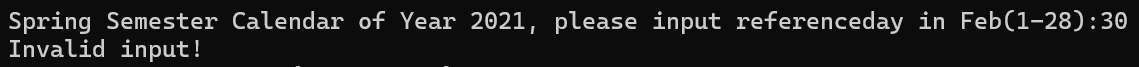
### 测试点 1（无效输入）的测试结果

测试步骤：输入无效的referenceDay， 观察输出情况。

预期测试结果：输出“Invalid input！”

实际测试结果（截图）：





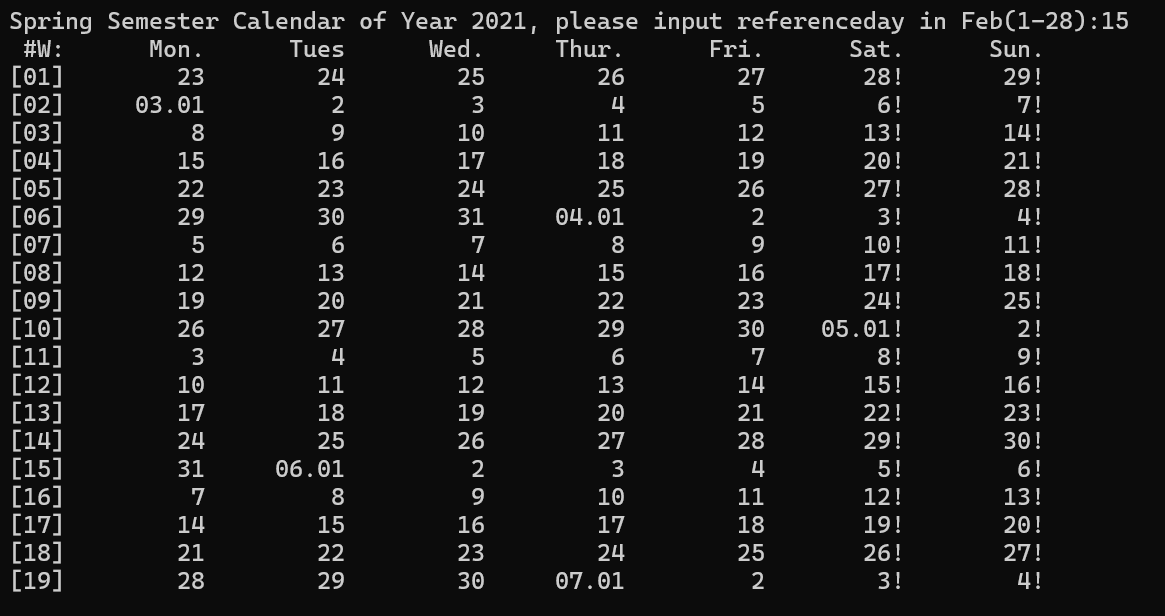
测试结论：正确提示报错信息。

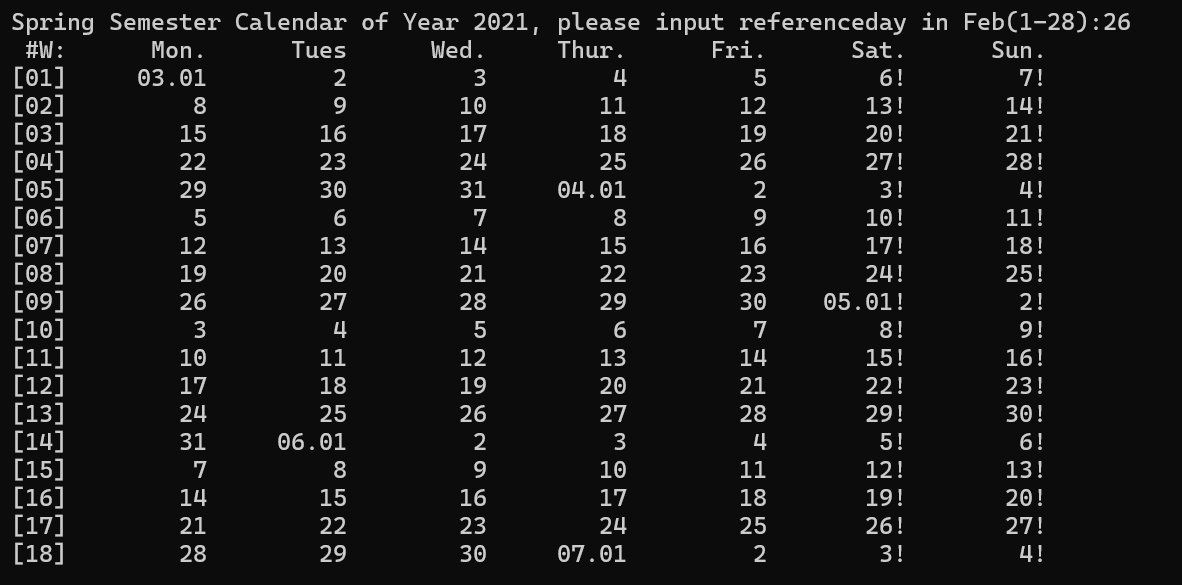
### 测试点 2（有效输入）的测试结果

测试步骤：输入有效referenceDay，观察输出情况。

预期测试结果：正确打印2021年华科春季校历

实际测试结果（截图）：





测试结论：正确打印2021年华中科技大学春季校历。

## 实验结论

代码达到功能目标

## 实验总结

在该实验中，用getMonthLength函数对计算天数长度过程进行封装，极大简化了主函数，增强了主函数的可读性。但是子函数仍然很重复啰嗦，需要用数组进行改进。总体来说，函数版的日历打印程序，基于函数编程，使主函数更能反映主体编程思路。

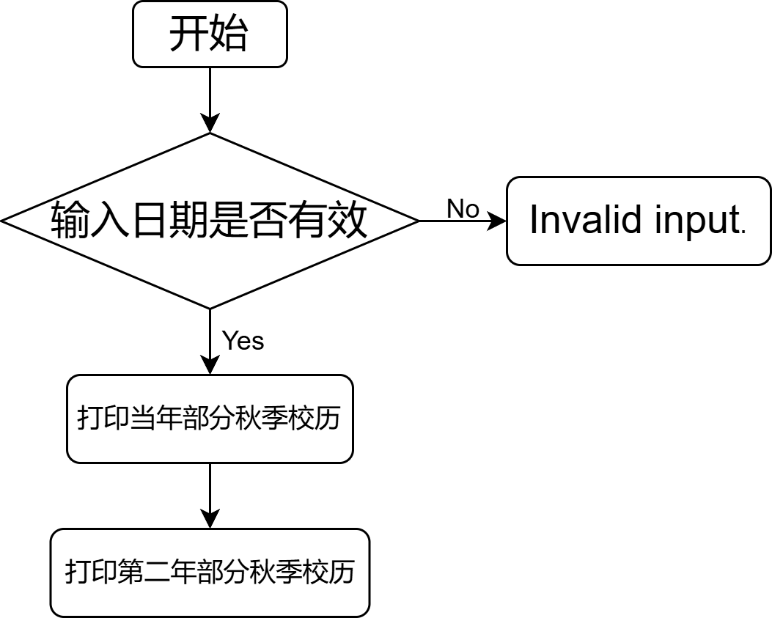
# 实验六：打印秋季校历【函数版】（Calendar06）

## 实验任务

用函数改写校历程序，打印2021年秋季学期校历。

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



主函数部分：

1. int main()
2. {
3. int year, referenceDay;
4. printf("Autumn Semester Calendar, please input referenceYear(2000- 2030): and referenceday in Jan(1-31) eg.[YYYY-DD] :");
5. scanf("%d-%d", &year, &referenceDay);
6. if( year < 2000 || year > 2030 || referenceDay < 1 || referenceDay > 31 ){
7. printf("Invalid input.\n");
8. return 0;
9. }
10. printf("Autumn Semester Calendar of %d\n", year);
11. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
12. int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;
13. int sStartSeqOfYear = getThisMonday( year, getDaySeq(year, 9, 1) );
14. int sEndSeqOfYear = getThisSunday( year + 1, getDaySeq(year + 1, 1, referenceDay) );
15. for(daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0, weekSeqOfSemester = 1;
16. daySeqOfYear <= getDaySeq(year, 12, 31);
17. daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7)
18. {
19. if( daySeqOfWeek == 0 ){
20. printf("[%02d]", weekSeqOfSemester);
21. }
22. printOneDay(year, daySeqOfYear);
23. if( daySeqOfWeek == 6 ){
24. printf("\n");
25. weekSeqOfSemester ++;
26. }
27. }
28. for(daySeqOfYear = 1, daySeqOfWeek, weekSeqOfSemester;
29. daySeqOfYear <= sEndSeqOfYear;
30. daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7)
31. {
32. if( daySeqOfWeek == 0){
33. printf("[%02d]", weekSeqOfSemester);
34. }
35. if(weekSeqOfSemester == 13){
36. ;
37. }
38. printOneDay(year + 1, daySeqOfYear);
39. if( daySeqOfWeek == 6 ){
40. printf("\n");
41. weekSeqOfSemester ++;
42. getThisMonday( year, getDaySeq(year, 9, 1) );
43. }
44. }
45. return 0;
46. }

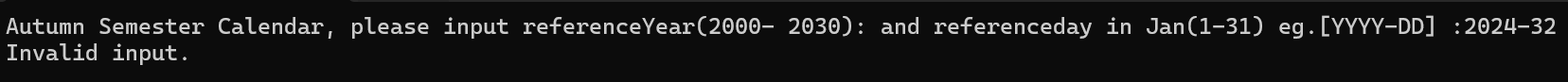
## 代码测试

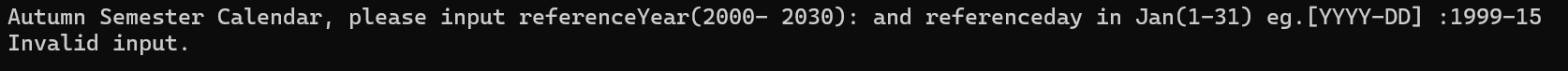
### 测试点1(无效输入)的测试结果

测试步骤：输入无效日期，观察输出情况。

预期测试结果：输出“Invalid input.”

实际测试结果（截图）：





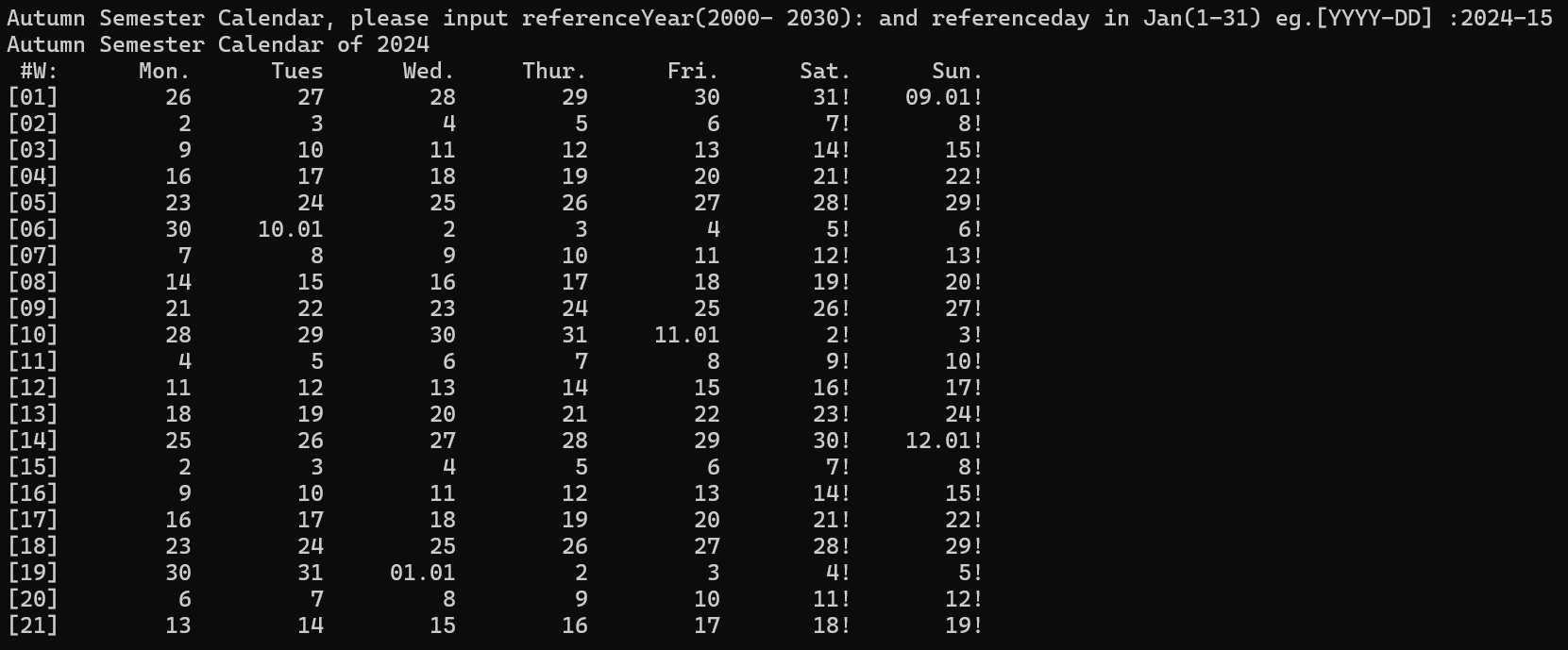
测试结论：正确提示报错信息。

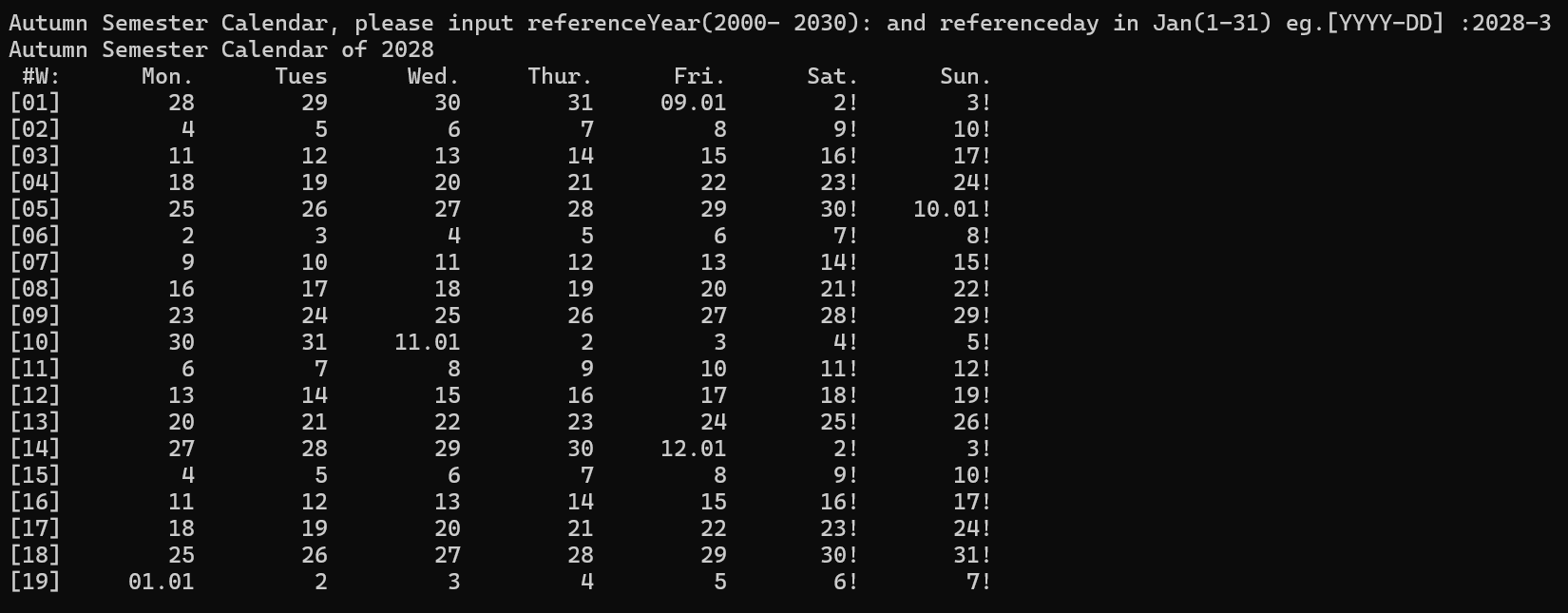
### 测试点2（有效输入）的测试结果

测试步骤：输入有效日期，观察输出情况。

预期测试结果：正确打印秋季校历

实际测试结果（截图）：





测试结论：正确打印秋季校历。

## 实验结论

代码达到功能目标。

## 实验总结

* 1. 在编写代码时，由于没有学习到数组，getMonthLength函数仍然比较重复。不过可以选择定义lengthMon等变量进行适当简化，达到改进效果。
  2. 在主函数中，打印当年秋季校历和后一年秋季校历用了两个极为相似的循环，降低了主函数的可读性。我认为可以修改循环控制条件进行优化或者编写一个新的子函数进行封装。
  3. 在getMonth函数中，我最开始没有注意到调用GetMonthLength函数时参数的范围限制问题（没有在循环内写if语句判断month==12时进行break跳出），因此出现隐蔽的错误（11月30日打印之后）就导致程序结束。If语句判断month==12有必要添加进getMonth函数。

# 实验七：打印全学年校历【函数版】（Calendar07）

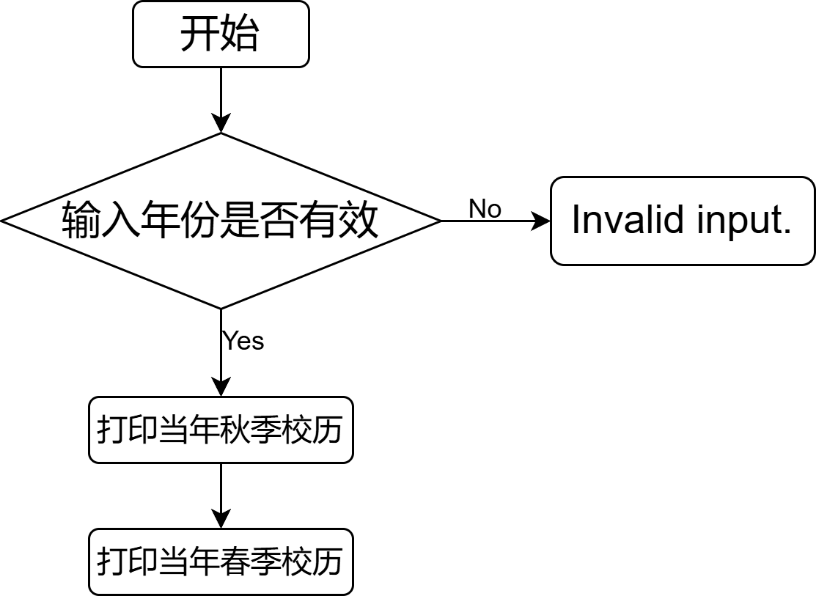
## 实验任务

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。

## 实验步骤



主函数部分：

1. int main()
2. {
3. int inputYear;
4. printf("School Calendar, please input the year (%d-%d):\n", YEAR\_MIN, YEAR\_MAX);
5. scanf("%d", &inputYear);
6. if(inputYear < YEAR\_MIN || inputYear > YEAR\_MAX){
7. printf("Invalid input.");
8. return -1;
9. }
10. int startWeek, endWeek, maxWeek, week, i;
11. printf("%13s%s%d%s%d\n", " ", "First Semester (Fall) Calendar of Year ", inputYear, "-", inputYear + 1);
12. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
13. startWeek = getWeekSeqOfYear(inputYear, 9, 4);
14. maxWeek = getWeekSeqOfYear(inputYear, 12, 31);
15. endWeek = getWeekSeqOfYear(inputYear + 1, 1, 20) - 1;
16. for(week = startWeek, i = 1; week <= maxWeek; week ++, i ++){
17. printOneWeek(inputYear, week, i);
18. }
19. for(week = 2 ; week <= endWeek; week ++, i ++){
20. printOneWeek(inputYear + 1, week, i);
21. }
22. inputYear ++;
23. printf("%13s%s%d%s%d\n", " ", "Second Semester (Spring) Calendar of Year ", inputYear - 1, "-", inputYear);
24. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
25. startWeek = getWeekSeqOfYear(inputYear, 2, 15) + 1;
26. endWeek = getWeekSeqOfYear(inputYear, 7, 1) ;
27. for(week = startWeek, i = 1; week <= endWeek; week ++, i ++){
28. printOneWeek(inputYear, week, i);
29. }
30. return 0;
31. }

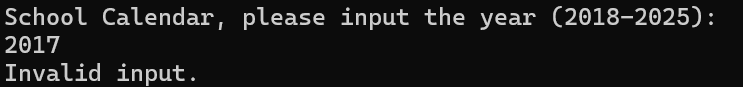
## 代码测试

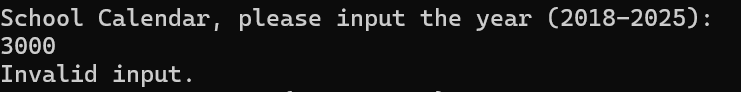
### 测试点1（无效输入）的测试结果

测试步骤：输入无效年份，观察输出情况。

预期测试结果：输出“Invalid input.”

实际测试结果（截图）：





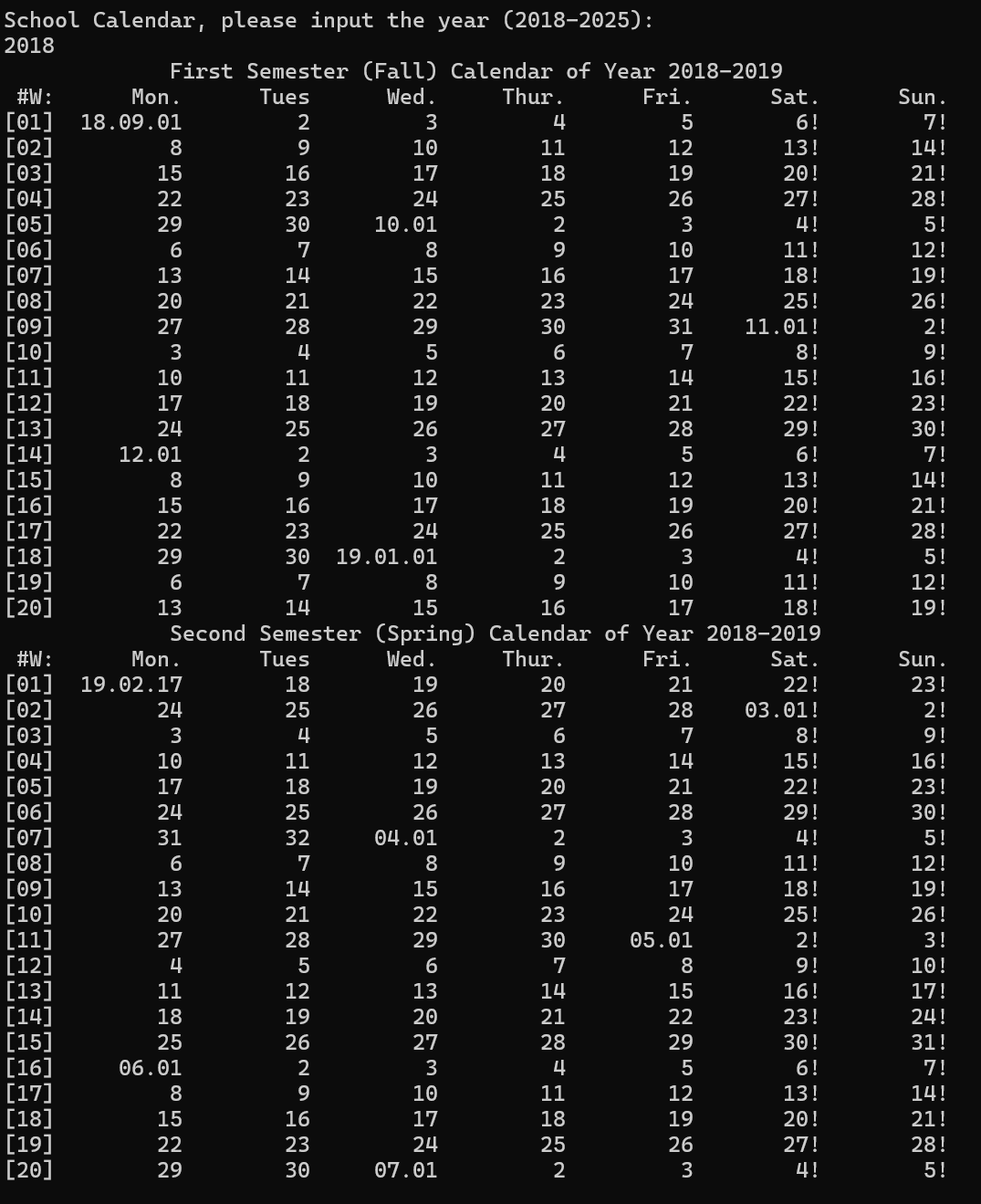
测试结论：正确提示报错信息。

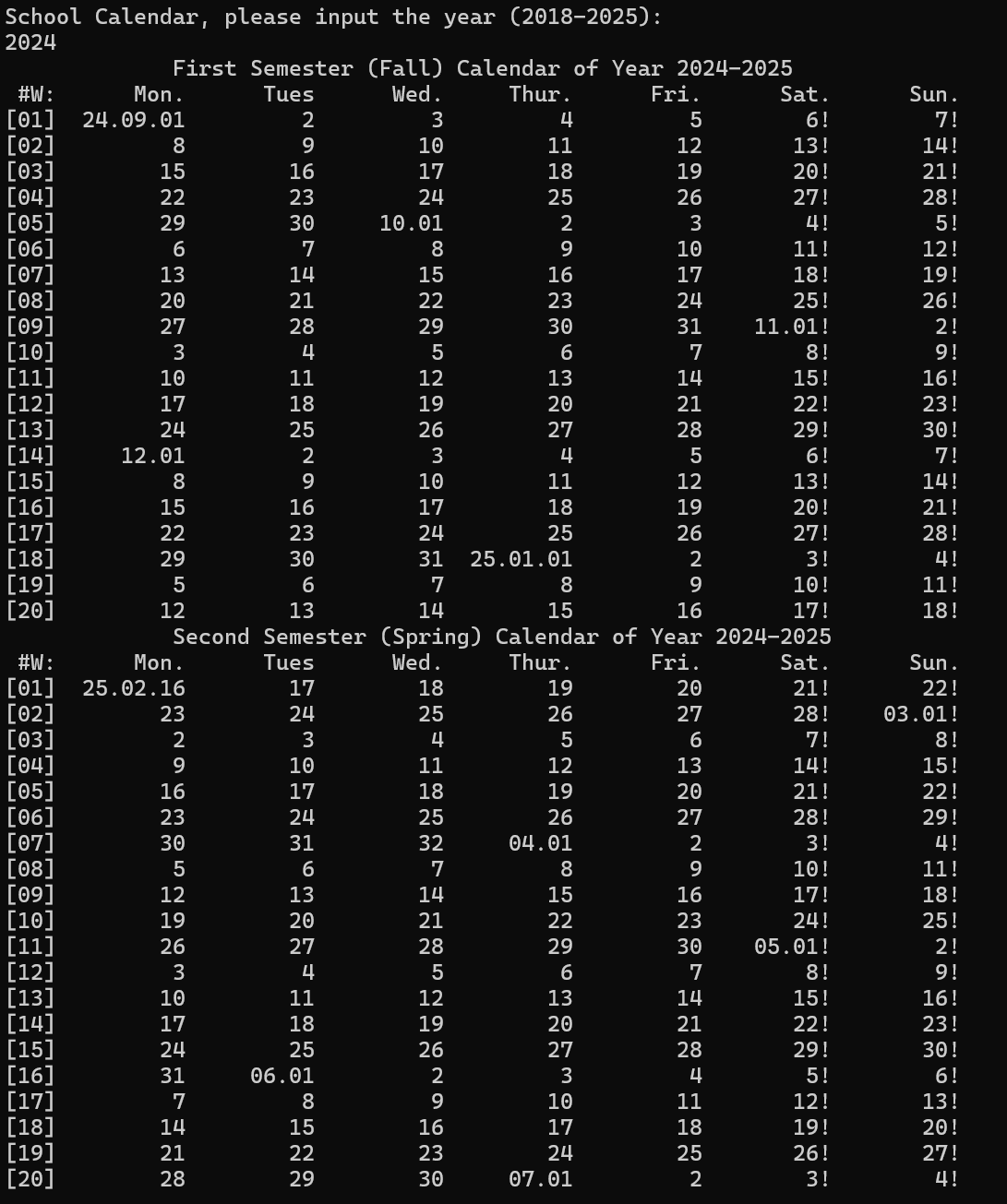
### 测试点2（有效输入）的测试结果

测试步骤：输入有效年份，观察输出情况。

预期测试结果：打印该学年秋季校历和春季校历。

实际测试结果（截图）：





测试结论：正确打印当年秋季校历和春季校历。

## 实验结论

代码达到功能目标。

## 实验总结

* 1. 在写每一个循环步骤的时候，应当在脑中过一遍循环过程，特别是循环的开头和结尾。如在main.c的打印秋季校历部分中，第二个for循环的起始值应为week = 2，因为第一个for循环的结尾已经打印了次年1月1日所在周。如果在第二个for循环对week初始赋值week=1，会出现重复打印的情况。
  2. 主函数的打印过程略显重复，如改进，可以用函数封装打印过程。

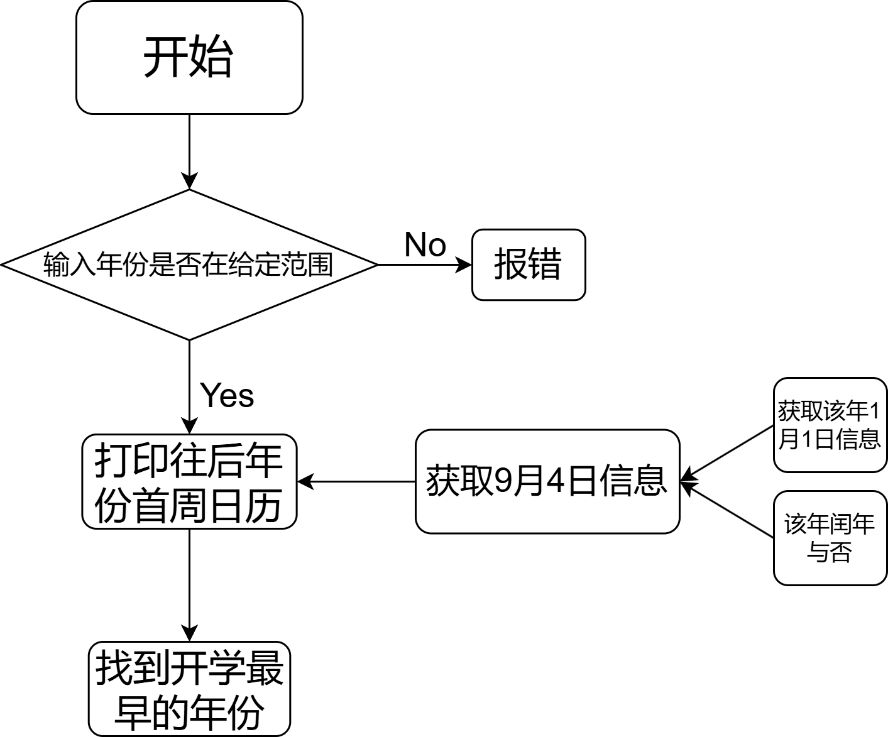
# 实验八：不同年份秋季校历首周对比（Calendar08）

## 实验任务

1.比较多个年份的秋季校历首周信息和首日信息，寻找开学最早的那一年；

2.约定秋季学期从当年的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；

## 实验步骤



主函数部分：

1. int main()
2. {
3. int inputYear;
4. printf("Autumn Calendar Comparision in %d years\
5. ,plesase input the first year(%d-%d):\n",YEAR\_NUM,YEAR\_MIN,YEAR\_MAX - YEAR\_NUM);
6. scanf("%d",&inputYear);
7. if(inputYear < YEAR\_MIN || inputYear + YEAR\_NUM > YEAR\_MAX){
8. printf("the input year is not supported.\n");
9. return 1;
10. }
11. int i;
12. int Years[YEAR\_NUM] = {0};
13. int startDays[YEAR\_NUM] = {0};
14. int year;
15. year = inputYear;
16. for(i = 0; i < YEAR\_NUM;i ++){
17. Years[i] = year;
18. year ++;
19. printf("\n%s%s%d\n"," ", "First week in autumn Calendar of Year ", Years[i]);
20. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.","Tues.","Wed.","Thur.","Fri.","Sat.","Sun." );
21. printOneWeek(Years[i], getDaySeqOfWeek( Years[i], 9, 4),1);
22. startDays[i]= getThisMonday( Years[i], getDaySeq( Years[i], 9, 4));
23. }
24. int min = 0;
25. for(i = 1; i < YEAR\_NUM;i ++){
26. min = ( startDays[i] < startDays[min])? i : min;
27. }
28. printf("\n%s%s%d\n"," ","Earliest Autumn Semester is in Year ",Years[min]);
29. return 0;
30. }

打印首周周历函数部分：

1. void printOneWeek(int year, int daySeqofWeek, int weekseqshow)
2. {
3. int simpleYear;
4. int month =  9;
5. int day = 4;
6. int dayfirst,monthfirst;
7. if(daySeqofWeek <= 4){
8. monthfirst = month;
9. dayfirst = day - daySeqofWeek + 1;
10. }else{
11. monthfirst = month - 1;
12. dayfirst = 31 - (daySeqofWeek - day) + 1;
13. }*//计算周一在几月几日*
14. simpleYear = year % 100;
15. printf("[01] %02d.%02d.%02d",simpleYear,monthfirst,dayfirst);
16. int i;
17. for(i = 0; i < 6; i ++){
18. dayfirst ++;
19. if(dayfirst == 32){
20. dayfirst = 1;
21. printf("%5s0%d.0%d"," ",month,dayfirst);
22. i ++;
23. }
24. printf("%10d",dayfirst);
25. }*//打印周一至周日*
26. printf("\n");
27. }

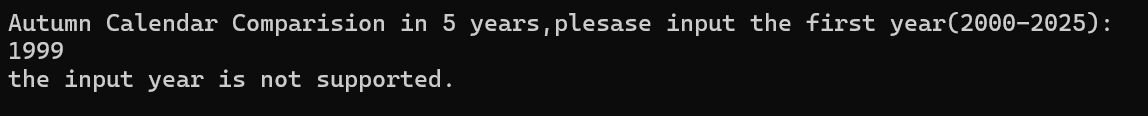
## 代码测试

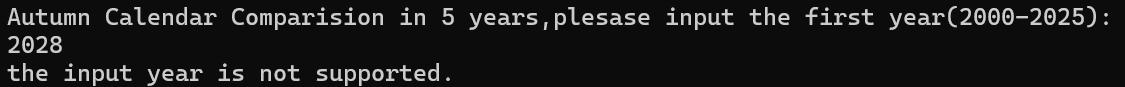
### 3.3.1测试点1（无效输入）的测试结果

测试步骤：输入给定范围外的年份，观察输出情况。

预期输出：预期得到报错信息“the input year is not supported.”

实际测试结果（截图）：





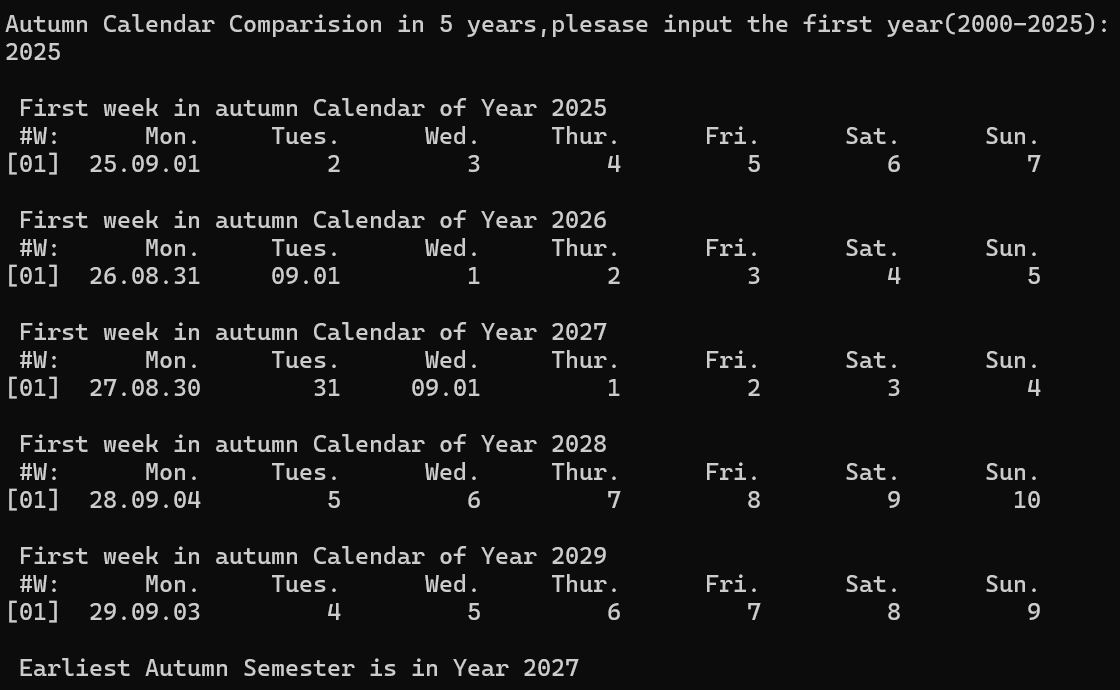
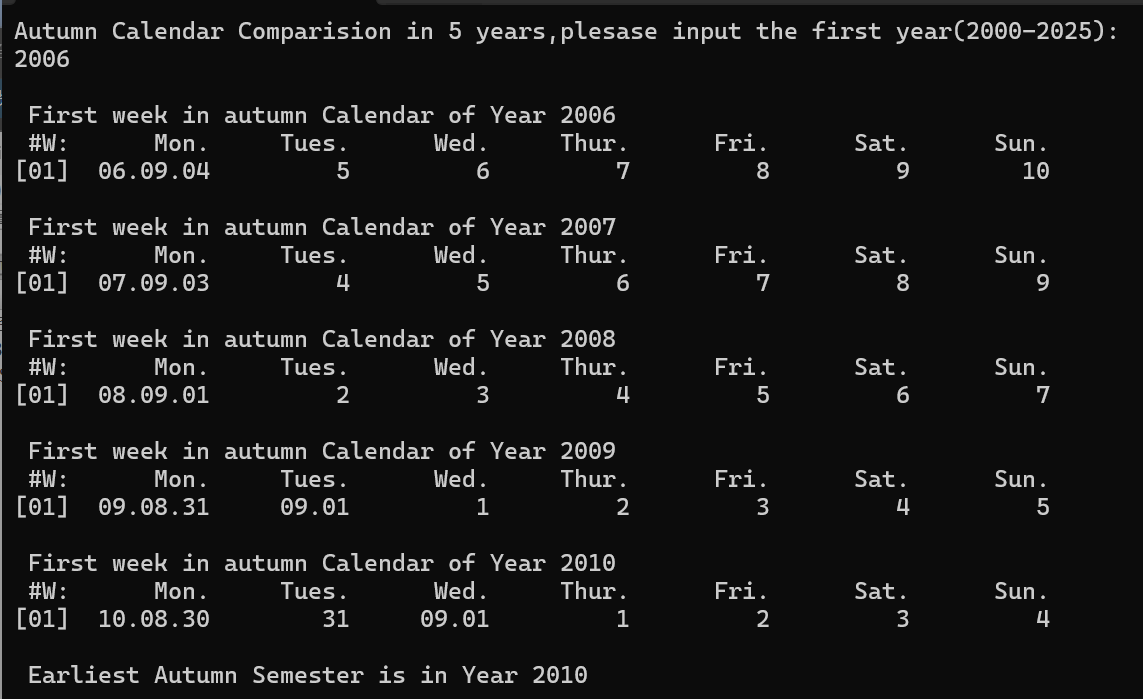
测试结论：在输入给定范围外的年份时，正确提示输入错误。

### 3.3.2测试点2（正常输入）的测试结果

测试步骤：输入给定范围内的年份，观察输情况。

预期输出：打印给定年份后若干年秋季学期首周周历，并在最后输出这若干年中开学最早的一年。

实际测试结果（截图）：



测试结论：正确打印这若干年秋季开学首周周历，并正确找到其中开学最早的年份。

## 实验结论

代码达到功能目标。能正确打印若干年秋季学期首周周历并找出开学最早的年份。

## 实验总结

1. 对月份数组构造和使用的分析

在构造月份数组时，我声明为13个元素的数组并将初值赋为0，让偏移量（月份）对应的数值为该月份对应天数。但在输入二月长度的时候，考虑到闰年的影响，我选择定义两个不同的数组（分别为28天、29天）而不是定义一个通用数组。定义通用数组虽然在定义的时候会比定义两个数组简洁，但也意味着在后续判断闰年的函数中对2月对应天数进行修改，而我为了精简代码过程，防止多个函数中都出现对月份数组重复定义，用const int进行全局变量的定义。因此定义一个通用数组就不再合适。

但是，我对getDaySeq函数的定义还是有些复杂。我认为可以对getDaySeq函数进行改进：新声明定义一个selectMonthArray函数用于生成当年月份数组，再将该月份数组名作为参数传进getDaySeq函数，避免getDaySeq函数中if-else语句中的重复循环。

1. 对主函数设计的反思

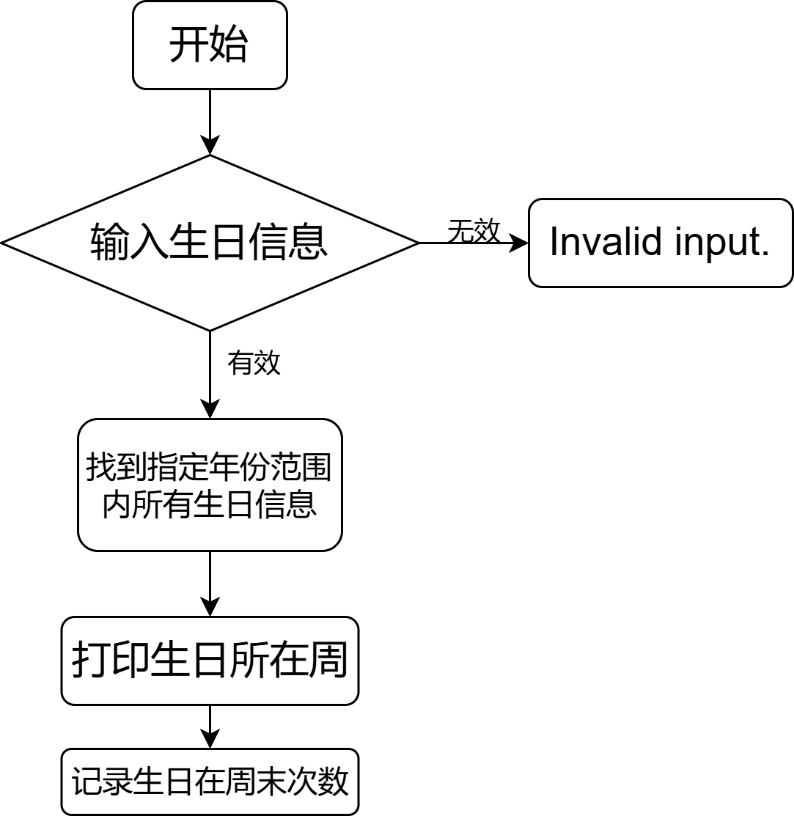
主函数通常反映程序主体思路，而我在编写代码过程中把Years[]数组生成过程和寻找开学最早年份的过程写进了主函数。因此为了使主函数更能体现代码编写思路，可以设计函数setYearArray封装Years[]数组的生成过程，设计findEarliestDay函数封装寻找开学最早年份的过程。

# 实验九：寻找生日【多维数组】(Calendar09)

## 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，计算其周末过生日的次数。

## 实验步骤



主函数部分：

1. int main()
2. {
3. int inputMonth, inputDay, Years[YEAR\_NUM];
4. int Days[YEAR\_NUM][366][4] = {0};
5. printf("Finding Birthday in year (%d - %d), please input the month and day : \n", YEAR\_MIN, YEAR\_MAX);
6. scanf("%d-%d", &inputMonth, &inputDay);
7. if (! isNormal(inputMonth, inputDay)) {
8. printf("Invalid input.\n");
9. return -1;
10. }
11. setYearArray(Years, YEAR\_NUM, YEAR\_MIN);
12. initialDays(Years, Days, YEAR\_NUM);
13. int totalNum = 0, weekendNum = 0;
14. for (int i = 0; i < YEAR\_NUM; i++) {
15. int year = Years[i];
16. int yearLength = isLeapYear(year)? 366 : 365;
17. printf("\nBirthday in Year %d\n", year);
18. printf(" #M:%10s%10s%10s%10s%10s%10s%10s\n", "MON.", "Tues.", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
19. if (inputMonth == 2 && inputDay == 29 && !isLeapYear(year)) {
20. printf("Not Found.\n");
21. }
22. else {
23. for (int j = 0; j < (isLeapYear(year)? 366 : 365); j++) {
24. if (Days[i][j][0] == inputMonth && Days[i][j][1] == inputDay) {
25. totalNum++;
26. if (Days[i][j][3] == 0 || Days[i][j][3] == 6) {
27. weekendNum++;
28. }
29. }
30. }
31. printWeeks(year, getWeekSeqOfYear(year, inputMonth, inputDay), inputMonth, inputDay);
32. }
33. }
34. printf("Total %d birthdays are found, %d of them are in weekends.\n", totalNum, weekendNum);
35. return 0;
36. }

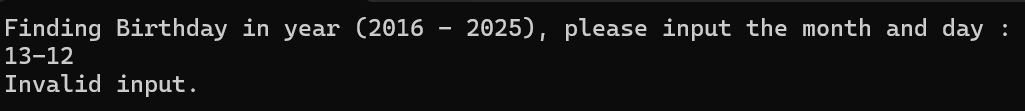
## 代码测试

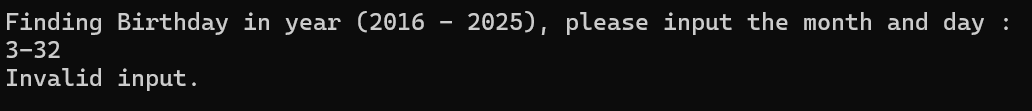
### 测试点1（无效输入）的测试结果

测试步骤：输入无效日期，观察输出情况。

预期测试结果：输出“Invalid input.”

实际测试结果（截图）：





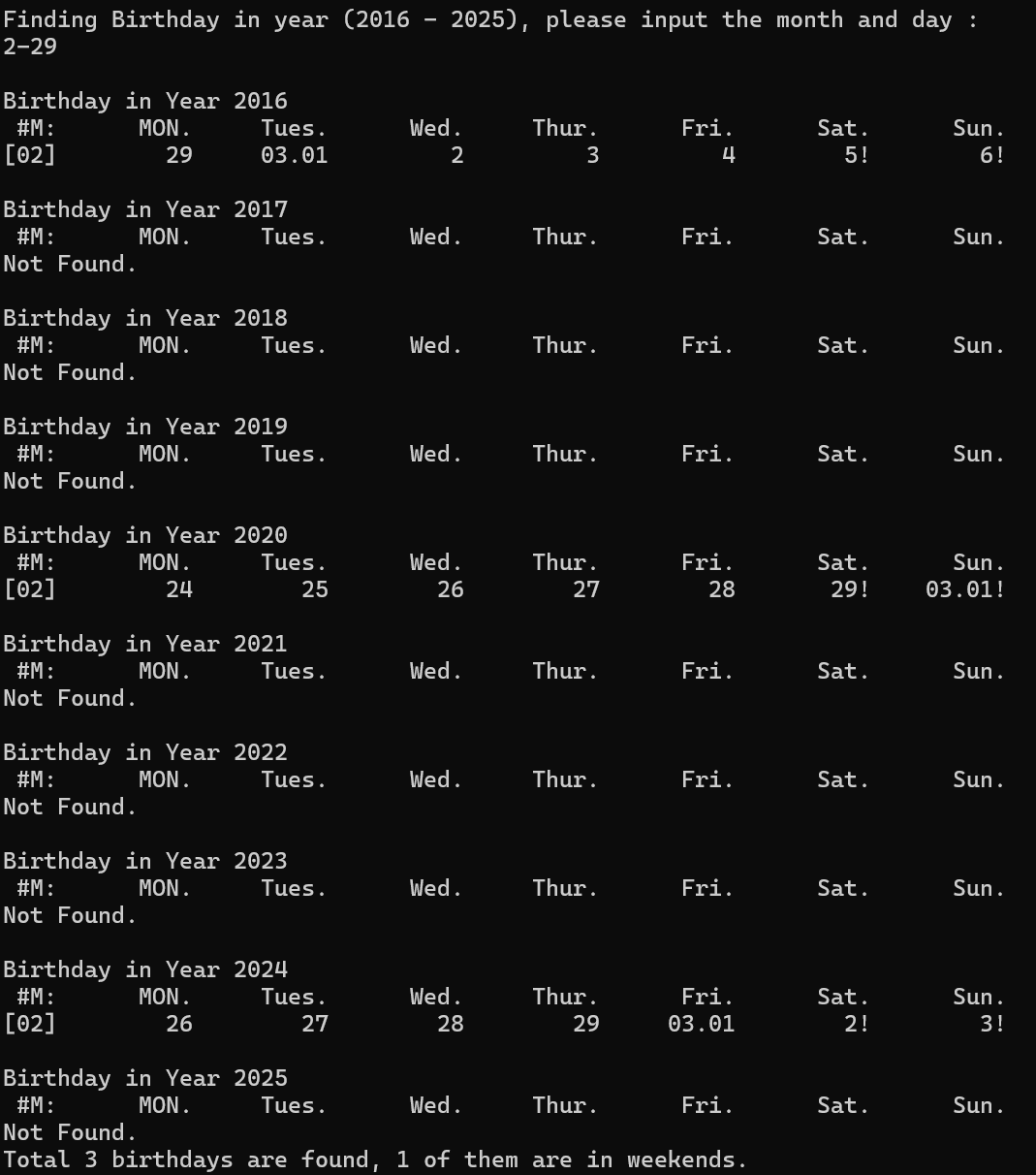
测试结论：正确提示报错信息。

### 测试点1（特殊日期输入）的测试结果

测试步骤：输入特殊日期（2-29），观察输出情况。

预期测试结果：在闰年正确打印2-29所在周周历，在非闰年提示Not found.最后提示找到生日数和在周末次数。

实际测试结果（截图）：



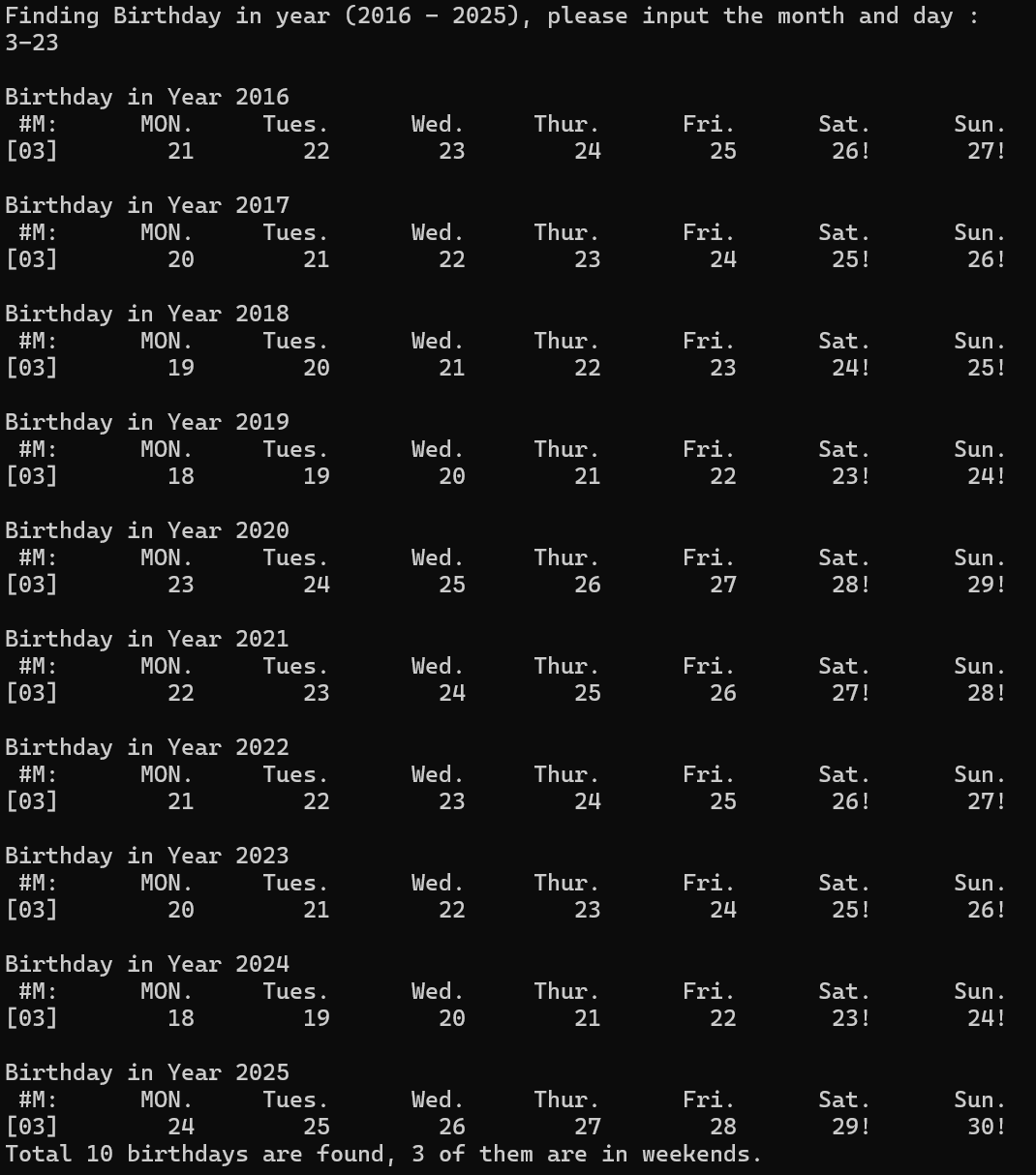
测试结论：与预期测试结果相符。

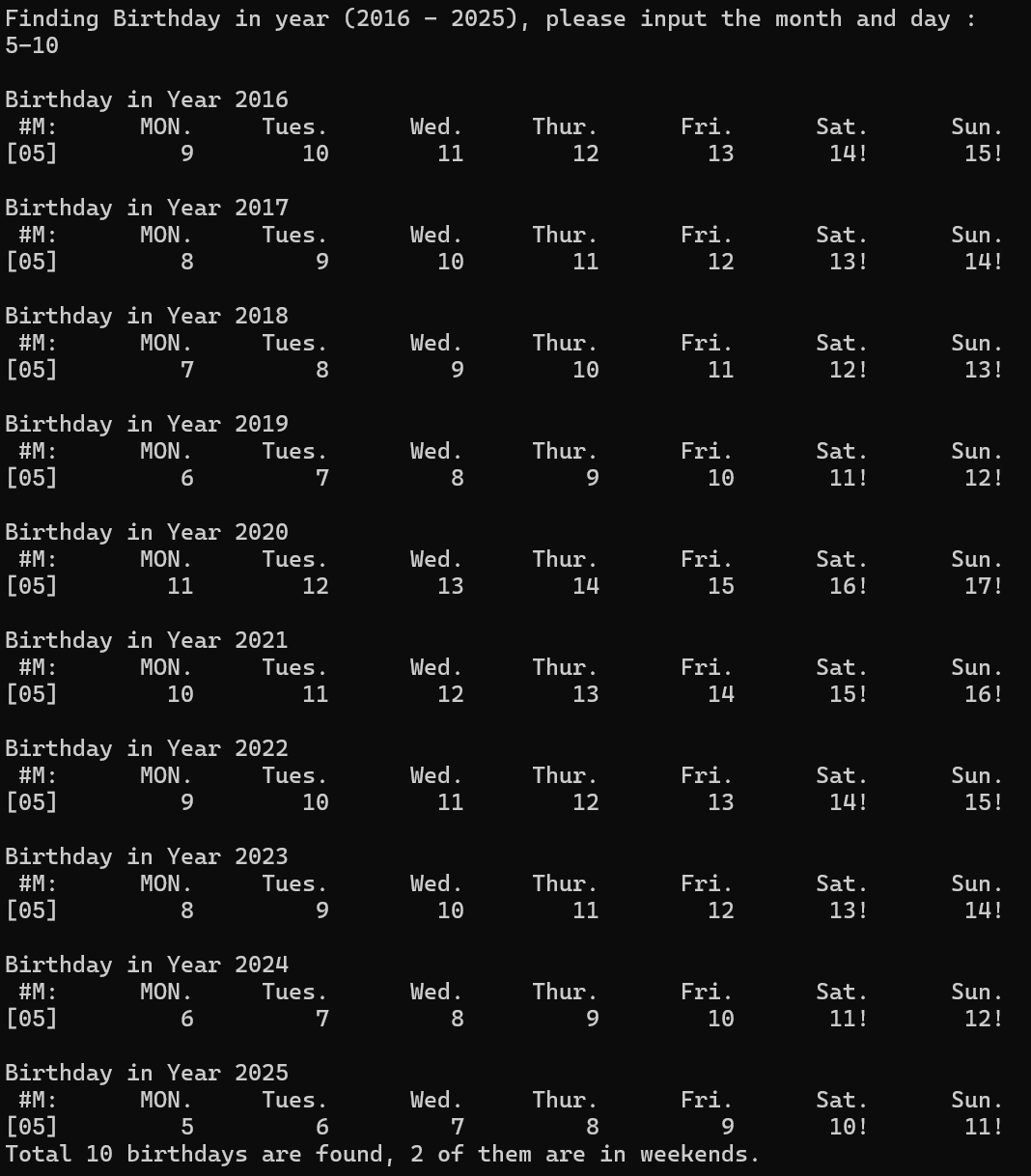
### 测试点1（寻常日期输入）的测试结果

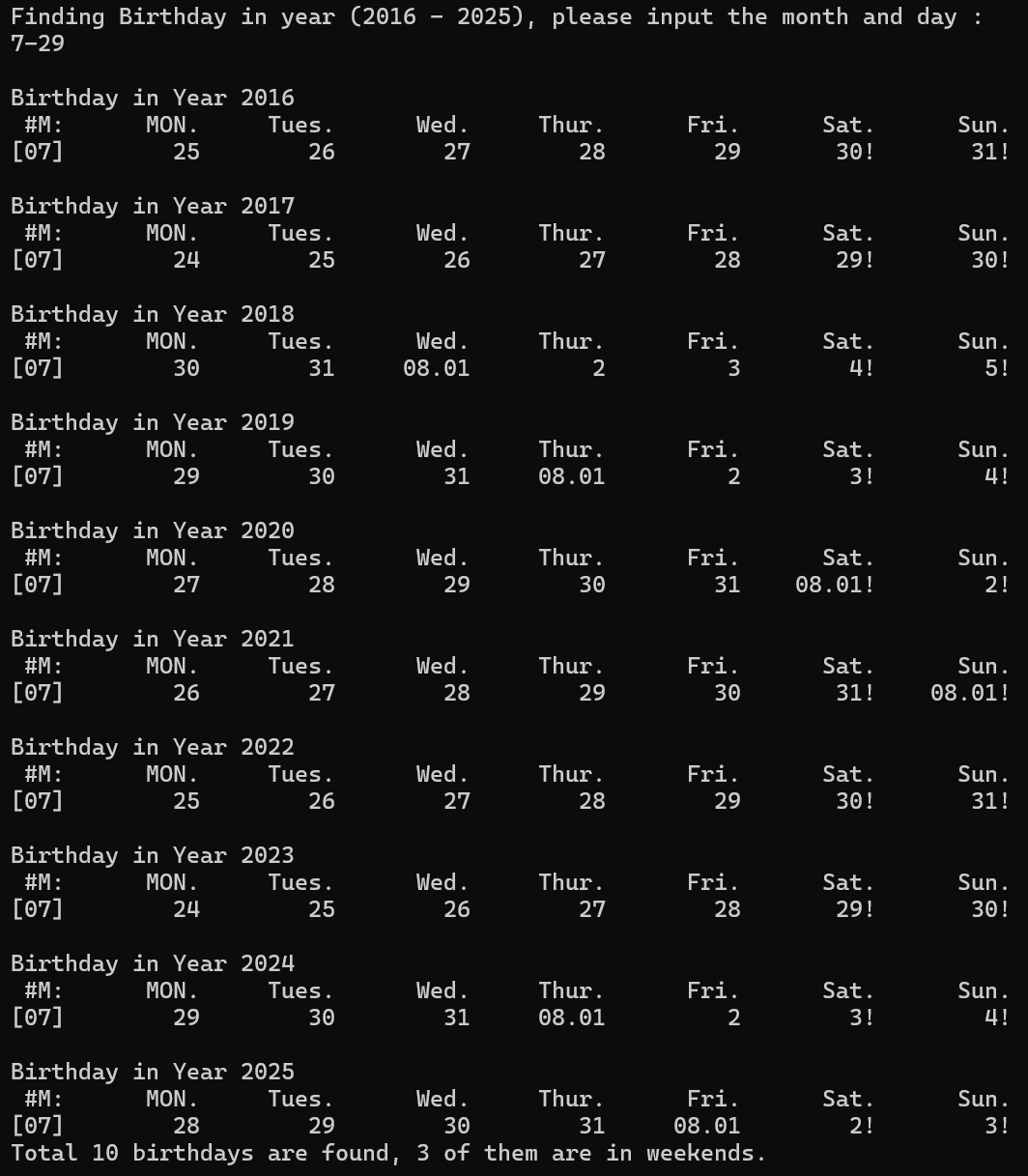
测试步骤：输入寻常日期，观察输出情况。

预期测试结果：打印给定年份范围内该生日所在周周历，并汇报找到的生日数和在周末数。

实际测试结果（截图）：







## 实验结论

代码达到功能目标

## 实验总结

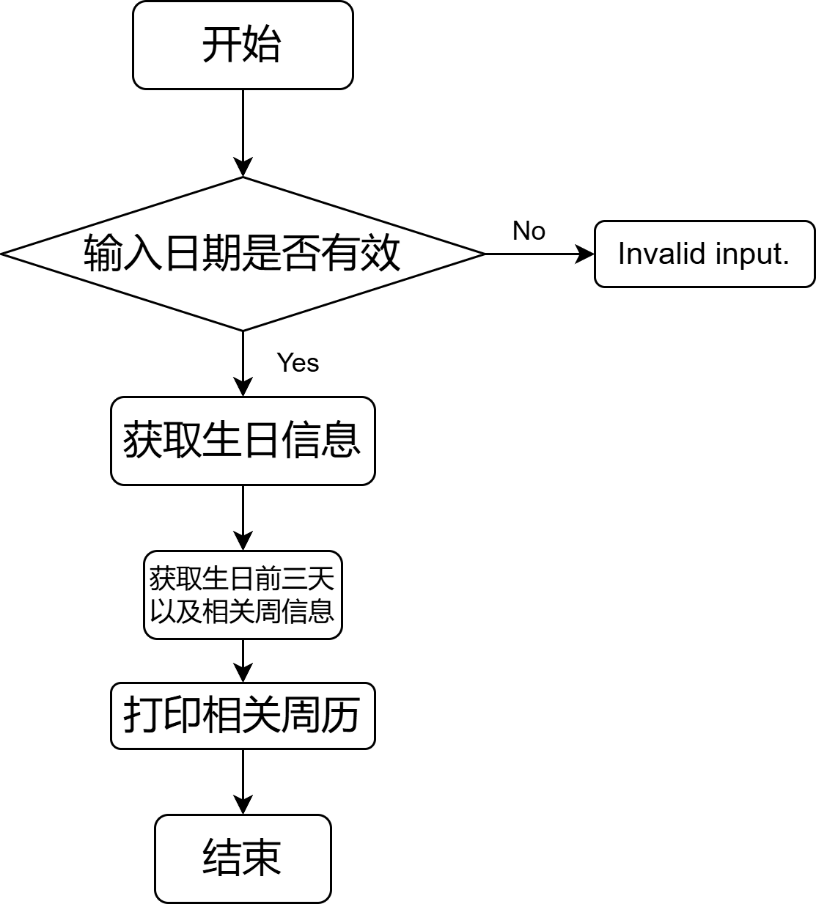
* 1. 最初思考设计三维数组储存信息的时候，不同于之前的Calendar实验，daySeqOfYear通过计算得出，而是储存在数组的第二维中。这就使得之前系列的函数不能完全植入该实验（强行进行该操作就需要将数组传进各个子函数，降低了数组的安全性），因此对各个计算日期的函数就需要重新思考设计。当然，我最后选择了另一种方法，根据三维数组第三维的信息新建一个函数计算daySeqOfYear。
  2. 在编写printWeeks函数时，仅靠year、weekSeq、month三个函数就将周历打印出来，对我来说强行写会导致函数过长，因此我新添加一个参数day（当月第几天），并新设计一个函数printOneDay来进行简化。

# 实验十：寻找生日【结构体】(Calendar10)

## 实验任务

1. 假定某人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；
2. 用日期结构体记录单一日期的所有数据属性；

## 实验步骤



主函数部分：

1. int main()
2. {
3. int inputMonth, inputDay;
4. printf("Finding Birthday in year(%d-%d), please input month and day (mm-dd):\n", YEAR\_START, YEAR\_END);
5. scanf("%d-%d", &inputMonth, &inputDay);
7. int Years[YEAR\_NUM]; *// 2018-2027*
8. setYearArray(Years, YEAR\_NUM, YEAR\_START);
9. int i;
10. for(i = 0; i < YEAR\_NUM; i ++){
11. printf("\n%s%s%d\n", " ", "Birthday in Year ", Years[i]);
12. int ifPrint = isDay(Years[i], inputMonth, inputDay);
13. if(ifPrint == 0){
14. printf(" Not found.\n");
15. continue;
16. }else if(ifPrint == 2){
17. printf("Invalid input.\n");
18. continue;
19. }
20. Day birthDay, prepareDay, printStartDay;
21. int printDayRange = 3;
22. birthDay = setDay(Years[i], inputMonth, inputDay); *// get the birthday*
23. prepareDay = getDayBefore(birthDay, printDayRange );      *// get the 3 day before the birthday*
24. printStartDay = getDayBefore(prepareDay, prepareDay.weekDay - 1);*// get the first day of the week*
25. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tue.", "Wed.", "Thu.", "Fri.", "Sat.", "Sun.");
26. int displayFormat = 0;
27. if(birthDay.weekDay < 4){
28. printf(" [%02d]", birthDay.weekSeq - 1);
29. }else{
30. printf(" [%02d]", birthDay.weekSeq);
31. }
32. Day currentDay;
33. for( currentDay = printStartDay; getTwoDaysInterval(currentDay, birthDay) <= 0; currentDay = getDayAfter(currentDay, 1)){
34. if( getTwoDaysInterval(currentDay, prepareDay) < 0){
35. displayFormat = DATE\_NOSHOW;
36. }else if( getTwoDaysInterval(currentDay, prepareDay) == 0){
37. displayFormat = DATE\_INFO\_FULL;
38. }else if( getTwoDaysInterval(currentDay, birthDay) < 0){
39. displayFormat = DATE\_INFO\_BRIEF;
40. }else if( getTwoDaysInterval(currentDay, birthDay) == 0){
41. displayFormat = DATE\_STAR;
42. }
43. printDay(currentDay, displayFormat);
44. if( currentDay.weekDay == 7 && getTwoDaysInterval(currentDay, birthDay) != 0){
45. printf("\n [%02d]", birthDay.weekSeq);
46. }
47. }
48. }
49. return 0;
50. }

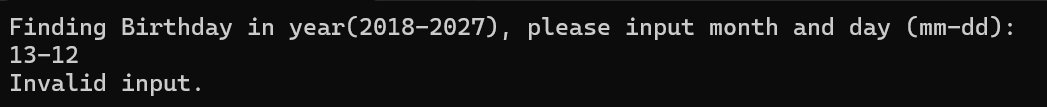
## 代码测试

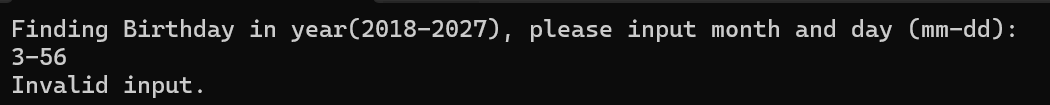
### 测试点1（输入无效日期）的测试结果

测试步骤：输入无效日期，观察输出情况。

预期测试结果：输出“Invalid input.”

实际测试结果（截图）：





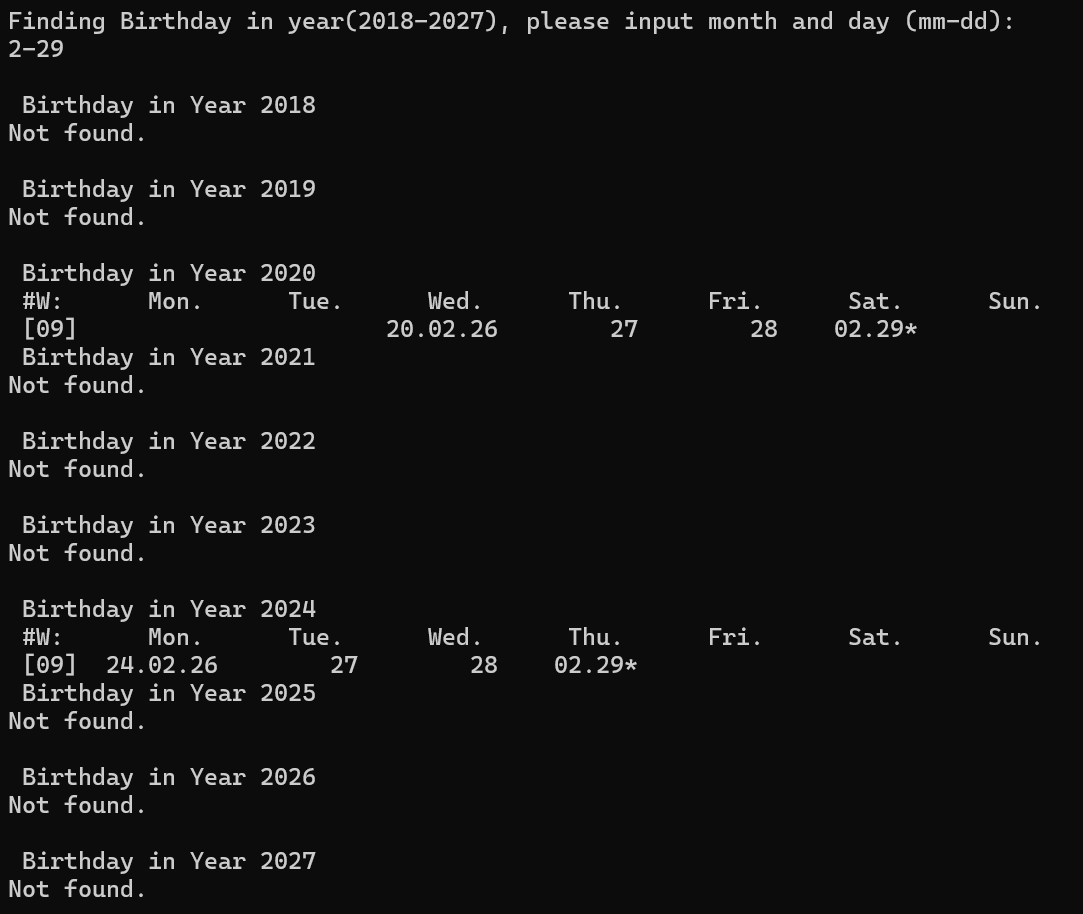
测试结论：正确提示无效输出。

### 5.3.2测试点1（输入特殊日期）的测试结果

测试步骤：输入特殊日期2月29日，观察输出情况。

预期测试结果：在非闰年输出“Not found.”，在闰年正常打印相关周历。

实际测试结果（截图）：



实际测试结果：

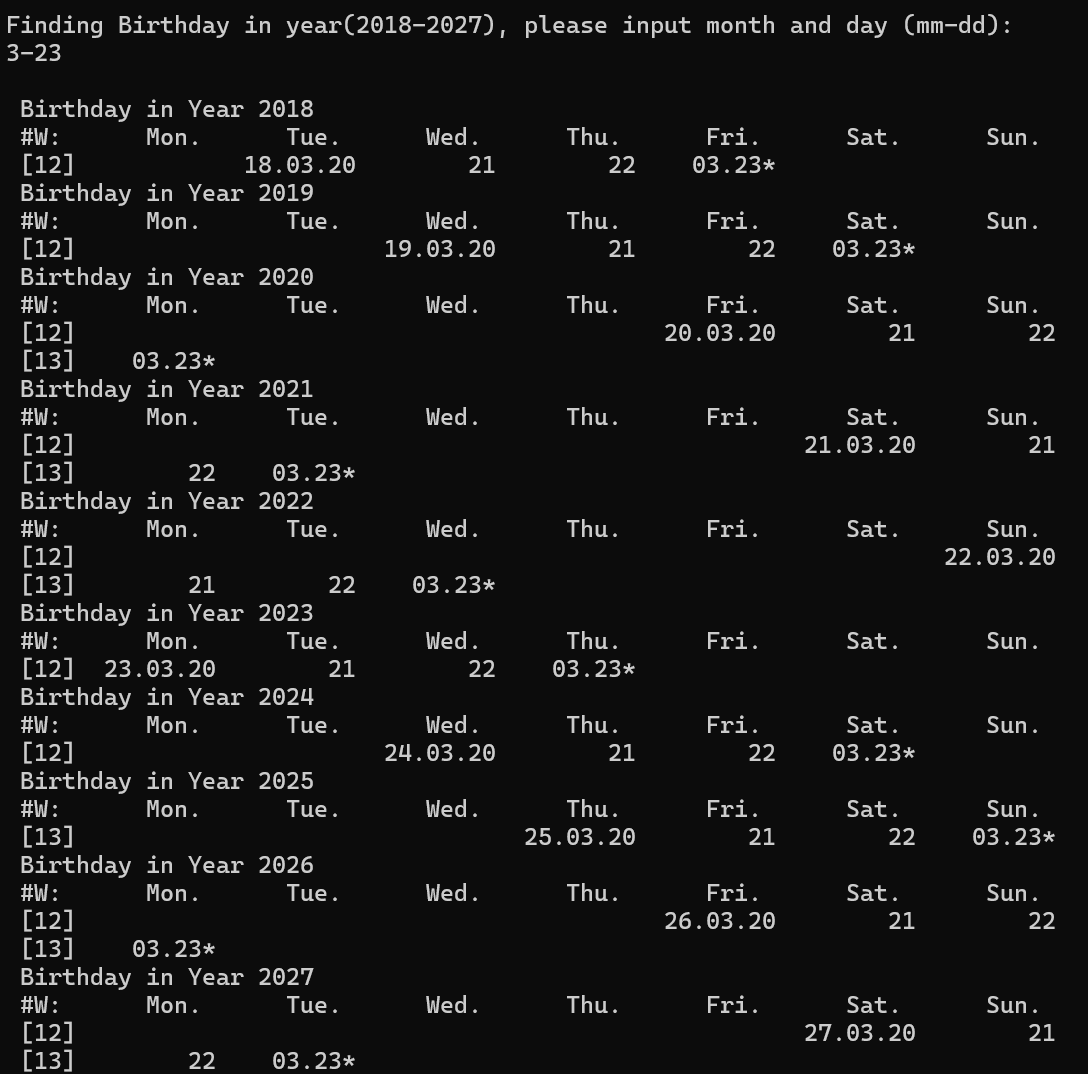
测试正常输出非闰年和闰年相关信息。

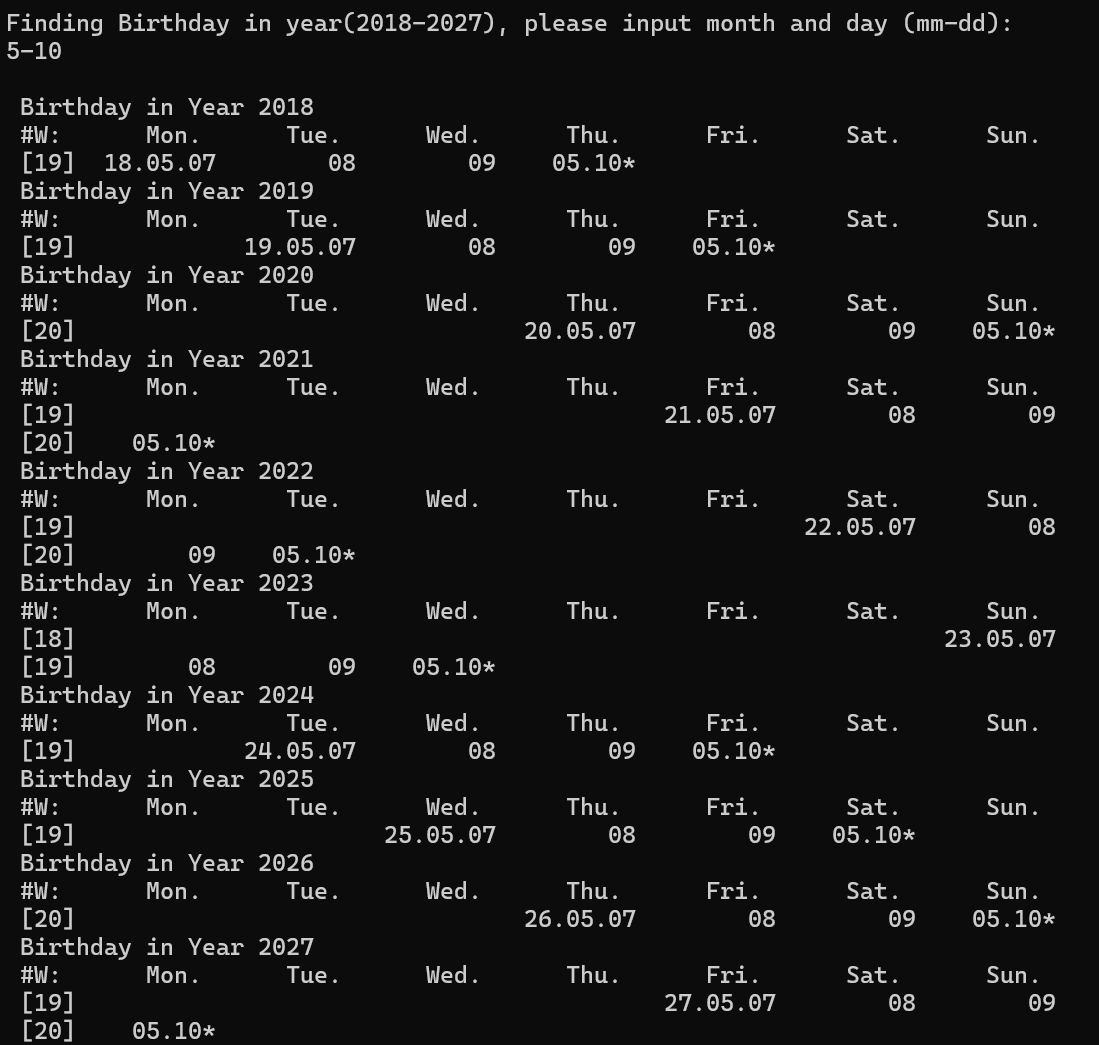
### 5.3.3测试点1（输入有效日期）的测试结果

测试步骤：输入正常日期，观察输出情况。

预期测试结果：正确打印生日相关周周历。

实际测试结果（截图）：





测试结果：正常打印生日趴相关周周历。

## 实验结论

代码达到功能目标

## 实验总结

1. 在实验八中，打印日期前繁琐的日期转换计算影响了代码的可读性。因此在本实验中，选择构造自定义的日期结构体类型，保存关于日期相关的所有信息，就无需再设立多个状态变量。比如在主函数最后打印相关周历信息的循环的时候，用相关函数封装，使循环中遍历的变量也是自定义的日期类型，该类型可以通过相关函数比较大小、偏移天数，最后一股脑全打印出来。
2. 当然，虽然定义了新的日期类型，但在对变量赋值的时候仍然需要相关计算，这也就不可避免地需要构造月份数组，源代码中没有定义全局数组变量，这使得月份数组代码在多函数中都出现，变得重复。可以改进：定义两个不同的全局数组变量（闰年、非闰年）进行简化。

# 本课程学习总结

最初学习C语言时，我并不懂什么“#include”，亦或是什么“int main（）”看着这些字母和符号分开来每一个都认识，拼在一起就不懂的组合，我只是一味的背下来这些细节、结构……更别提什么机器语言汇编语言之类的东西……

但随着课程的学习以及课下大量的时间投入，我发现C语言其实并没有那么高级，很多东西了解一下就可以掌握，比如printf和scanf两个函数看上去就是一副“生人勿近”的面孔，但实际上只是stdio头文件里面封装好的输出输入函数。最开始，我并没有意识到C语言并不难，焦虑于每周的作业，苦恼于长时间debug找不到错误点，也自嘲于无数次次scanf输出参数时忘记加取地址符&、数组和指针使用时越界导致的segmentation fault。

说到指针，对于其万物皆可指的精髓我虽明白，但还是经常混淆所指向之内容，也经常苦思它与多维数组之间的关联。虽然较为晦涩难懂，在空指针、野指针的苦海中遨游，但也架不住我隔三岔五就打开老师课上讲的课件进行复习。这也算是我进入大学后学到的第一个学习技巧：用好互联网，用好线上资源。

当然，在C语言的学习过程中，我也深刻领会到了不求甚解的学习理念。在学习前面的内容时，难免碰到没学到的后期内容，我们不需要马上就掌握这些后期学习内容，只需了解一个大概，在后面学到的时候进行适当回顾，这样既可以温习前期的代码思路、代码格式，又可以加深对后期学习内容的理解。比如在讲scanf函数时，老师讲到取地址符&，最初我并不明白&存在的意义，上网搜索也只是一知半解，直到后面学习指针的时候才明白。还有深刻一点：在函数引用参数时，引用单个数据变量，在函数中的形式参数并不会影响被引用的单个变量；但是将数组名传进函数，在函数中对其进行修改就会改变主函数中数组的值。初学时我并不明白这样的差异是由什么导致的，直到学完指针才得以弄清楚，更加深化了对指针、数组的理解。老师在讲链表入门时，我更深刻地理解了指针必须要注意的：不能无所指！在编写Calendar系列实验的时候，我也注意到在较大工程中，函数的运用并没有那么简单，引用参数时稍不注意范围就会导致程序的崩溃，比如在打印秋季校历的时候，我没有注意到getMonth函数对month==12的情况，因此在后期调用输出最终结果时，只能打印到11.30。

虽然每次出现隐性错误时，debug的过程总是烦闷又长，但经过一学期的磨练，也算是摸到了一点点门道，比最初的自己能更快找到每次代码出现问题所在。这些东西无法通过别人传授获得，绝知此事要躬行！

我也学到了C语言之外的编程知识。比如多文件工程，在写Calendar07打印全年校历时，我切身体会到多文件工程对main.c文件的简化，极大地增加了代码的可读性。

经过老师一学期的教导，我学到的不仅是C语言本身的知识，更是将来编程良好的习惯。当看到其他班同学声明全局数组全局变量不加const时，我一看就不顺眼，认为他的习惯不好，再比如看到其他同学写代码不注意缩进不注意空格时，我切身体会到代码难看到不想读的样子，也更深刻理解老师对我们的习惯要求，便更加严格要求自己的代码编写规范。

“斗争，失败，在斗争，再失败，再斗争，直至胜利。”总之，C语言带我踏进了电子之门，让我更深刻地理解了软硬件之间的结合，过程是辛苦的，但结果是收获。C语言的学习过程也让我认识到任何新知识的学习都无法一蹴而就，甚至可能会出现投入大量时间却收不到可视化结果的情况，但只有坚持到底，坚持“一万小时”，才有可能从入门到精通！

# 附录

## 实验一：计算星期几（Calendar01）

main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define current\_year 2024
4. #define weekday\_JAN1st 1
5. int main()
6. {
7. int month1,day,dayseq,daynum;
8. printf("This year is 2024,please input your month(1-12) and day(1-31) and the DaySeqOfWeek will be presented:\n");
9. scanf("%d%d",&month1,&day);
10. if(month1 == 1){
11. daynum = day;
12. } else if(month1 == 2){
13. daynum = day + 31;
14. } else if(month1 == 3){
15. daynum = 31 + 29 + day;
16. } else if(month1 == 4){
17. daynum = 31 + 29 + 31 + day;
18. } else if(month1 == 5){
19. daynum = 31 + 29 + 31 + 30 + day;
20. } else if(month1 == 6){
21. daynum = 31 + 29 + 31 + 30 + 31 + day;
22. } else if(month1 == 7){
23. daynum = 31 + 29 + 31 + 30 + 31 + 30 + day;
24. } else if(month1 == 8){
25. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + day;
26. } else if(month1 == 9){
27. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + day;
28. } else if(month1 == 10){
29. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + day;
30. } else if(month1 == 11){
31. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + day;
32. } else if(month1 == 12){
33. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 +30 + day;
34. }
35. daynum = daynum % 7;
36. if(daynum == 0){
37. daynum = 7;
38. }
39. dayseq = weekday\_JAN1st + daynum - 1;
41. printf("%d-%d-%d is No.%d day in that week ",current\_year,month1,day,dayseq);
42. return 0;
43. }

## 实验二：打印月历（Calendar02）

main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define current\_year 2024
4. #define weekday\_Jan1st 1
5. int main()
6. {
7. int month;
8. printf("please input a month in 2024:");
9. scanf("%d",&month);
10. if(month > 12){
11. printf("please input a proper month!");
12. }else{
13. int day,dayseq1,daynum;
14. day = 1;
15. if(month == 1){
16. daynum = day;
17. } else if(month == 2){
18. daynum = day + 31;
19. } else if(month == 3){
20. daynum = 31 + 29 + day;
21. } else if(month == 4){
22. daynum = 31 + 29 + 31 + day;
23. } else if(month == 5){
24. daynum = 31 + 29 + 31 + 30 + day;
25. } else if(month == 6){
26. daynum = 31 + 29 + 31 + 30 + 31 + day;
27. } else if(month == 7){
28. daynum = 31 + 29 + 31 + 30 + 31 + 30 + day;
29. } else if(month == 8){
30. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + day;
31. } else if(month == 9){
32. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + day;
33. } else if(month == 10){
34. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + day;
35. } else if(month == 11){
36. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + day;
37. } else if(month == 12){
38. daynum = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 +30 + day;
39. }
40. daynum = daynum % 7;
41. if(daynum == 0){
42. daynum = 7;
43. }
44. dayseq1 = weekday\_Jan1st + daynum - 1;
45. int monlength = 30;
46. if(month == 1||month == 3||month == 5||month == 7||month == 8||month == 10||month == 12){
47. monlength = 31;
48. } else if(month == 2){
49. monlength = 29;
50. }
51. printf("\n%d-%d has total %d days.\n",current\_year,month,monlength);
52. printf("%d-%d-1 is No.%d day in that week.\n",current\_year,month,dayseq1);
53. printf("\n%10s%10s%10s%10s%10s%10s%10s\n",
54. "Mon.","Tues.","wedn.","Thur.","Fri.","Sat.","Sun.");
55. int spacenum = (dayseq1 -1) \* 10;
56. int space = 0;
57. for(space = 0;space < spacenum;space++){
58. printf(" ");
59. }
60. int date = 1;
61. int dayseq2;
62. while(date <= monlength){
63. if(date % 7 == 0){
64. dayseq2 = 7;
65. } else{
66. dayseq2 = date % 7;
67. }
68. if(spacenum + (dayseq2) \* 10 == 70){
69. printf("%10d\n",date);
70. } else {
71. printf("%10d",date);
72. }
73. date ++;
74. }
75. }
76. return 0;
77. }

## 实验三：打印周历（Calendar03）

main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define CURRENT\_YEAR 2021
4. int main()
5. {
6. int WEEKDAY\_JAN1;
7. WEEKDAY\_JAN1 = (CURRENT\_YEAR + (CURRENT\_YEAR - 1) / 4 - (CURRENT\_YEAR - 1) / 100 + (CURRENT\_YEAR - 1) / 400) % 7;
8. int IS\_LEAPYEAR;
9. IS\_LEAPYEAR = CURRENT\_YEAR % 4;
10. if(IS\_LEAPYEAR != 0){
11. IS\_LEAPYEAR = 1;
12. }
13. int lengthJan = 31;
14. int lengthFeb = (IS\_LEAPYEAR == 1)? lengthJan + 28 : lengthJan + 29;
15. int lengthMar = lengthFeb + 31;
16. int lengthApr = lengthMar + 30;
17. int lengthMay = lengthApr + 31;
18. int lengthJun = lengthMay + 30;
19. int lengthJul = lengthJun + 31;
20. int lengthAug = lengthJul + 31;
21. int lengthSep = lengthAug + 30;
22. int lengthOct = lengthSep + 31;
23. int lengthNov = lengthOct + 30;
24. int lengthDec = lengthNov + 31;
25. int weekSeqOfYear,month,day;
26. printf("Week Calandar of Year %d,please input week sequence(1-53): ",CURRENT\_YEAR);
27. scanf("%d",&weekSeqOfYear);
28. int dayfirst;
29. dayfirst = (weekSeqOfYear == 1)? 1 : (7 - WEEKDAY\_JAN1 + 1) + 1 + 7 \*(weekSeqOfYear - 2);
30. if(dayfirst <= lengthJan){
31. month = 1;
32. if(weekSeqOfYear == 1){
33. day = 1;
34. }else{
35. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1;
36. }
37. }else if(dayfirst <= lengthFeb){
38. month = 2;
39. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthJan;
40. }else if(dayfirst <= lengthMar){
41. month = 3;
42. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthFeb;
43. }else if(dayfirst <= lengthApr){
44. month = 4;
45. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthMar;
46. }else if(dayfirst <= lengthMay){
47. month = 5;
48. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthApr;
49. }else if(dayfirst <= lengthJun){
50. month = 6;
51. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthMay;
52. }else if(dayfirst <= lengthJul){
53. month = 7;
54. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthJun;
55. }else if(dayfirst <= lengthAug){
56. month = 8;
57. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthJul;
58. }else if(dayfirst <= lengthSep){
59. month = 9;
60. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthAug;
61. }else if(dayfirst <= lengthOct){
62. month = 10;
63. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthSep;
64. }else if(dayfirst <= lengthNov){
65. month = 11;
66. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthOct;
67. }else if(dayfirst <= lengthDec){
68. month = 12;
69. day = 1 + 7 \* (weekSeqOfYear - 1) - WEEKDAY\_JAN1 + 1 - lengthNov;
70. }else{
71. printf("input error!\n");
72. return -1;
73. }
74. printf("%10s%10s%10s%10s%10s%10s%10s\n","Mon.","Tues.","Wedn.","Thur.","Fri.","Sat.","Sun.");
75. int spacenum,count;
76. spacenum = (WEEKDAY\_JAN1 - 1) \* 10;
77. if(weekSeqOfYear == 1){
78. for(count = 1; count < spacenum ;count ++){
79. printf(" ");
80. }
81. for(spacenum ; spacenum < 70 ; spacenum = spacenum + 10){
82. printf("%7d.%2d",month,day);
83. day ++;
84. }
85. }
86. int Feb = (IS\_LEAPYEAR == 1)? 28 : 29;
87. int monthday[] = {0,31,Feb,31,30,31,30,31,31,30,31,30,31};
88. if(weekSeqOfYear > 1){
89. for(count = 1;count <= 7;count ++){
90. if(day == monthday[month] + 1){
91. month ++;
92. day = 1;
93. }
94. printf("%7d.%2d",month,day);
95. day ++;
96. if( month == 12 && day == 32){
97. break;
98. }
99. }
100. }
101. return 0;
102. }

## 实验四：打印春季校历（Calendar04）

main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define CURRENT\_YEAR 2021
4. int main()
5. {
6. int WEEKDAY\_JAN1;
7. WEEKDAY\_JAN1 = (CURRENT\_YEAR + (CURRENT\_YEAR - 1) / 4 - (CURRENT\_YEAR - 1) / 100 + (CURRENT\_YEAR - 1) / 400) % 7;
8. int IS\_LEAPYEAR;
9. IS\_LEAPYEAR = CURRENT\_YEAR % 4;
10. if(IS\_LEAPYEAR != 0){
11. IS\_LEAPYEAR = 1;
12. }
14. int lengthJan = 31;
15. int lengthFeb = (IS\_LEAPYEAR == 1)? lengthJan + 28 : lengthJan + 29;
16. int lengthMar = lengthFeb + 31;
17. int lengthApr = lengthMar + 30;
18. int lengthMay = lengthApr + 31;
19. int lengthJun = lengthMay + 30;
20. int lengthJul = lengthJun + 31;
21. int lengthAug = lengthJul + 31;
22. int lengthSep = lengthAug + 30;
23. int lengthOct = lengthSep + 31;
24. int lengthNov = lengthOct + 30;
25. int lengthDec = lengthNov + 31;
26. int referenceDay,sumday,dayseq,day,month;
27. month = 2;
28. printf("Spring Semester Clandar of Year %d,\
29. please input reference day in Feb(1-%d):\n",CURRENT\_YEAR,lengthFeb - lengthJan);
30. scanf("%d",&referenceDay);
31. if(referenceDay < 1 || referenceDay > lengthFeb - lengthJan){
32. printf("Invalid Input!\n");
33. return 0;
34. }
35. printf("%18s%s%d\n"," ","Spring Semester Calandar of Year ",CURRENT\_YEAR);
36. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n","Mon.","Tues.","Wed.","Thur.","Fri.","Sat.","Sun.");
37. sumday = lengthJan + referenceDay;
38. dayseq = (sumday % 7 + WEEKDAY\_JAN1) % 7 ;
39. dayseq = (dayseq == 0)? 7 : dayseq;
40. day = referenceDay;
41. for(dayseq ; dayseq <= 7 ; dayseq ++){
42. day ++;
43. if(day >= lengthFeb - lengthJan){
44. day = 1;
45. month ++;
46. }
47. }
48. int length = lengthJun;
49. int endday = 0;
50. length = (length % 7 + WEEKDAY\_JAN1) % 7 ;
51. while(length <= 7){
52. length ++;
53. endday ++;
54. }
55. int Feb = (IS\_LEAPYEAR == 1)? 28 : 29;
56. int monthday[] = {0,31,Feb,31,30,31,30,31,31,30,31,30,31};
57. int count = 0;
58. int i = 0;
59. for(day; day <= monthday[month]; day ++){
60. if(count % 7 == 0){
61. i ++;
62. if(i < 10){
63. printf("\n[0%d]",i);
64. }else if(!(month == 7 && day == endday)){
65. printf("\n[%2d]",i);
66. }
67. }
68. count ++;
69. if(day < 10){
70. printf("%7d.0%d",month,day);
71. }else{
72. printf("%7d.%2d",month,day);
73. }
74. if(day == monthday[month]){
75. day = 0;
76. month ++;
77. }
78. if(month == 7 && day == endday){
79. break;
80. }
81. }
82. return 0;
83. }

## 实验五：打印春季校历【函数版】（Calendar05）

main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define CURRENT\_YEAR 2021
4. #define JAN\_1ST 4
5. *//Functions about month and day*
6. int getMonthLength(int month);
7. int getDaySeq(int month, int day);
8. *//Functions for properties of one day*
9. int getMonth(int daySeqOfYear);
10. int getDay(int daySeqOfYear);
11. int getDaySeqOfWeek(int daySeqOfYear);
12. *//Functions for day movement calculations*
13. int getNextMonday(int daySeqOfYear);
14. int getThisSunday(int daySeqOfYear);
15. *//Functions support school calendar display*
16. void printOneDay(int daySeqOfYear);
17. int main()
18. {
19. int referenceDay;
20. printf("Spring Semester Calendar of Year 2021, please input referenceday in Feb(1-28):");
21. scanf("%d", &referenceDay);
22. if(referenceDay < 1 || referenceDay > 28){
23. printf("Invalid input!");
24. return -1;
25. }
26. int sStartSeqOfYear = getNextMonday( getDaySeq(2, referenceDay) );
27. int sEndSeqOfYear = getThisSunday( getDaySeq(7, 1) );
28. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
29. int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;
30. for( daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0,weekSeqOfSemester = 1;
31. daySeqOfYear <= sEndSeqOfYear;
32. daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7)
33. {
34. if( daySeqOfWeek == 0){
35. printf("[%02d]", weekSeqOfSemester );
36. }
37. printOneDay( daySeqOfYear );
38. if(daySeqOfWeek == 6){
39. printf("\n");
40. weekSeqOfSemester ++;
41. }
42. }
43. return 0;
44. }
45. int getMonthLength(int month)
46. {
47. int monthLength = 0;
48. if(month == 1){
49. monthLength = 0;
50. } else if(month == 2){
51. monthLength = 31;
52. } else if(month == 3){
53. monthLength = 31 + 29;
54. } else if(month == 4){
55. monthLength = 31 + 29 + 31;
56. } else if(month == 5){
57. monthLength = 31 + 29 + 31 + 30;
58. } else if(month == 6){
59. monthLength = 31 + 29 + 31 + 30 + 31;
60. } else if(month == 7){
61. monthLength = 31 + 29 + 31 + 30 + 31 + 30;
62. } else if(month == 8){
63. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31;
64. } else if(month == 9){
65. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31;
66. } else if(month == 10){
67. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30;
68. } else if(month == 11){
69. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31;
70. } else if(month == 12){
71. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 +30;
72. }
73. return monthLength;
74. }
75. int getDaySeq(int month, int day)
76. {
77. int daySeq = day;
78. daySeq += getMonthLength( month );
79. return daySeq;
80. }
81. int getMonth(int daySeqOfYear)
82. {
83. int month = 1;
84. while(daySeqOfYear > getMonthLength(month + 1)){
85. month ++;
86. }
87. return month;
88. }
89. int getDay(int daySeqOfYear)
90. {
91. int day, month;
92. month = getMonth( daySeqOfYear );
93. day = daySeqOfYear - getMonthLength( month );
94. return day;
95. }
96. int getDaySeqOfWeek(int daySeqOfYear)
97. {
98. int daySeqOfWeek;
99. daySeqOfWeek = (daySeqOfYear + JAN\_1ST - 1) % 7;
100. return daySeqOfWeek;
101. }
102. int getNextMonday(int daySeqOfYear)
103. {
104. int daySeqOfWeek = getDaySeqOfWeek( daySeqOfYear );
105. return daySeqOfYear + (7 - daySeqOfWeek) + 1;
106. }
107. int getThisSunday(int daySeqOfYear)
108. {
109. int daySeqOfWeek = getDaySeqOfWeek( daySeqOfYear );
110. return daySeqOfYear + 7 - daySeqOfWeek;
111. }
112. void printOneDay(int daySeqOfYear)
113. {
114. int day, month, daySeqOfWeek;
115. day = getDay( daySeqOfYear );
116. month = getMonth( daySeqOfYear );
117. daySeqOfWeek = getDaySeqOfWeek( daySeqOfYear );
118. if(day == 1 && (daySeqOfWeek == 0 || daySeqOfWeek == 6) ){
119. printf("%4s%02d.%02d!", " ", month, day);
120. }else if(day == 1){
121. printf("%5s%02d.%02d", " ", month, day);
122. }else if(day != 1 && (daySeqOfWeek == 0 || daySeqOfWeek == 6) ){
123. printf("%9d!", day);
124. }else{
125. printf("%10d", day);
126. }
127. }

## 实验六：打印秋季校历【函数版】（Calendar06）

main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. int isLeapYear( int year );*//1 for leapYear, 0 for not*
4. int getDaySeqOnJan1( int year );
5. int getMonthLength( int year, int month );
6. int getDaySeq( int year, int month, int day );
7. int getMonth(int year, int daySeqOfYear );
8. int getDay(int year, int daySeqOfYear );
9. int getDaySeqOfWeek(int year, int daySeqOfYear );
10. int getNextMonday(int year, int daySeqOfYear );
11. int getThisMonday(int year, int daySeqOfYear );
12. int getThisSunday(int year, int daySeqOfYear );
13. void printOneDay(int year, int daySeqOfYear );
14. int main()
15. {
16. int year, referenceDay;
17. printf("Autumn Semester Calendar, please input referenceYear(2000- 2030): and referenceday in Jan(1-31) eg.[YYYY-DD] :");
18. scanf("%d-%d", &year, &referenceDay);
19. if( year < 2000 || year > 2030 || referenceDay < 1 || referenceDay > 31 ){
20. printf("Invalid input.\n");
21. return 0;
22. }
23. printf("Autumn Semester Calendar of %d\n", year);
24. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
25. int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;
26. int sStartSeqOfYear = getThisMonday( year, getDaySeq(year, 9, 1) );
27. int sEndSeqOfYear = getThisSunday( year + 1, getDaySeq(year + 1, 1, referenceDay) );
28. for(daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0, weekSeqOfSemester = 1;
29. daySeqOfYear <= getDaySeq(year, 12, 31);
30. daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7)
31. {
32. if( daySeqOfWeek == 0 ){
33. printf("[%02d]", weekSeqOfSemester);
34. }
35. printOneDay(year, daySeqOfYear);
36. if( daySeqOfWeek == 6 ){
37. printf("\n");
38. weekSeqOfSemester ++;
39. }
40. }
41. for(daySeqOfYear = 1, daySeqOfWeek, weekSeqOfSemester;
42. daySeqOfYear <= sEndSeqOfYear;
43. daySeqOfYear ++, daySeqOfWeek ++, daySeqOfWeek %= 7)
44. {
45. if( daySeqOfWeek == 0){
46. printf("[%02d]", weekSeqOfSemester);
47. }
48. if(weekSeqOfSemester == 13){
49. ;
50. }
51. printOneDay(year + 1, daySeqOfYear);
52. if( daySeqOfWeek == 6 ){
53. printf("\n");
54. weekSeqOfSemester ++;
55. getThisMonday( year, getDaySeq(year, 9, 1) );
56. }
57. }
58. return 0;
59. }
60. int isLeapYear( int year )
61. {
62. int leap = 0;
63. if( year % 4 == 0 && year % 100 != 0 || year % 400 == 0 ){
64. leap = 1;
65. }
66. return leap;
67. }
68. int getDaySeqOnJan1( int year )
69. {
70. return (year + (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400) % 7;
71. }
72. int getMonthLength( int year, int month )
73. {
74. int monthLength = 0;
75. if(month == 1){
76. monthLength = 0;
77. } else if(month == 2){
78. monthLength = 31;
79. } else if(month == 3){
80. if(isLeapYear(year)){
81. monthLength = 31 + 29;
82. } else {
83. monthLength = 31 + 28;
84. }
85. } else if(month == 4){
86. monthLength = 31 + 29 + 31;
87. } else if(month == 5){
88. monthLength = 31 + 29 + 31 + 30;
89. } else if(month == 6){
90. monthLength = 31 + 29 + 31 + 30 + 31;
91. } else if(month == 7){
92. monthLength = 31 + 29 + 31 + 30 + 31 + 30;
93. } else if(month == 8){
94. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31;
95. } else if(month == 9){
96. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31;
97. } else if(month == 10){
98. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30;
99. } else if(month == 11){
100. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31;
101. } else if(month == 12){
102. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 +30;
103. }
104. return monthLength;
105. }
106. int getDaySeq( int year, int month, int day )
107. {
108. return getMonthLength(year, month) + day;
109. }
110. int getMonth(int year, int daySeqOfYear )
111. {
112. int month = 1;
113. while(daySeqOfYear > getMonthLength(year, month + 1)){
114. month ++;
115. if(month == 12){
116. break;
117. }
118. }
119. return month;
120. }
121. int getDay(int year, int daySeqOfYear )
122. {
123. return daySeqOfYear - getMonthLength(year, getMonth(year, daySeqOfYear));
124. }
125. int getDaySeqOfWeek(int year, int daySeqOfYear )
126. {
127. return (getDaySeqOnJan1(year) + daySeqOfYear - 1) % 7;*//0:Sunday, 1:Monday, 2:Tuesday, 3:Wednesday, 4:Thursday, 5:Friday, 6:Saturday*
128. }
129. int getNextMonday(int year, int daySeqOfYear )
130. {
131. int daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
132. if(daySeqOfWeek == 0){
133. return daySeqOfYear + 1;
134. } else {
135. return daySeqOfYear + (7 - daySeqOfWeek) + 1;
136. }
137. }
138. int getThisMonday(int year, int daySeqOfYear )
139. {
140. int daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
141. if(daySeqOfWeek == 0){
142. return daySeqOfYear - 6;
143. } else {
144. return daySeqOfYear - daySeqOfWeek + 1;
145. }
146. }
147. int getThisSunday(int year, int daySeqOfYear )
148. {
149. int daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
150. if(daySeqOfWeek == 0){
151. return daySeqOfYear;
152. }else{
153. return daySeqOfYear + (7 - daySeqOfWeek);
154. }
155. }
156. void printOneDay(int year, int daySeqOfYear )
157. {
158. int day, month, daySeqOfWeek;
159. day = getDay(year, daySeqOfYear);
160. month = getMonth(year, daySeqOfYear);
161. daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
162. if(day == 1 && (daySeqOfWeek == 0 || daySeqOfWeek == 6) ){
163. printf("%4s%02d.%02d!", " ", month, day);
164. }else if(day == 1){
165. printf("%5s%02d.%02d", " ", month, day);
166. }else if(day != 1 && (daySeqOfWeek == 0 || daySeqOfWeek == 6) ){
167. printf("%9d!", day);
168. }else{
169. printf("%10d", day);
170. }
171. }

## 实验七：打印全学年校历【函数版】（Calendar07）

### main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include "date.h"
4. #include "funs.h"
5. int main()
6. {
7. int inputYear;
8. printf("School Calendar, please input the year (%d-%d):\n", YEAR\_MIN, YEAR\_MAX);
9. scanf("%d", &inputYear);
10. if(inputYear < YEAR\_MIN || inputYear > YEAR\_MAX){
11. printf("Invalid input.");
12. return -1;
13. }
14. int startWeek, endWeek, maxWeek, week, i;
15. printf("%13s%s%d%s%d\n", " ", "First Semester (Fall) Calendar of Year ", inputYear, "-", inputYear + 1);
16. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
17. startWeek = getWeekSeqOfYear(inputYear, 9, 4);
18. maxWeek = getWeekSeqOfYear(inputYear, 12, 31);
19. endWeek = getWeekSeqOfYear(inputYear + 1, 1, 20) - 1;
20. for(week = startWeek, i = 1; week <= maxWeek; week ++, i ++){
21. printOneWeek(inputYear, week, i);
22. }
23. for(week = 2 ; week <= endWeek; week ++, i ++){
24. printOneWeek(inputYear + 1, week, i);
25. }
26. inputYear ++;
27. printf("%13s%s%d%s%d\n", " ", "Second Semester (Spring) Calendar of Year ", inputYear - 1, "-", inputYear);
28. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
29. startWeek = getWeekSeqOfYear(inputYear, 2, 15) + 1;
30. endWeek = getWeekSeqOfYear(inputYear, 7, 1) ;
31. for(week = startWeek, i = 1; week <= endWeek; week ++, i ++){
32. printOneWeek(inputYear, week, i);
33. }
34. return 0;
35. }

### date.h

1. #ifndef DATE\_H\_INCLUDED
2. #define DATE\_H\_INCLUDED
3. #define YEAR\_MIN 2018
4. #define YEAR\_MAX 2025
5. *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**
6. *//functjions used in this program*
7. *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**
8. *//*
9. *//functions on different year*
10. int isLeapYear(int year);
11. int getDaySeqOnJan1(int year);
12. *//*
13. *//functions about month and day*
14. int getMonthLength(int year, int month);
15. int getDaySeq( int year, int month, int day );
16. int getWeekSeqOfYear(int year, int month, int day);
17. *//*
18. *//functions for properties of one day*
19. int getMonth(int year, int daySeqOfYear );
20. int getDay(int year, int daySeqOfYear );
21. int getDaySeqOfWeek(int year, int daySeqOfYear );
22. *//*
23. *//functions for day movement calculation*
24. int getNextMonday(int year, int daySeqOfYear );
25. int getThisMonday(int year, int daySeqOfYear );
26. int getThisSunday(int year, int daySeqOfYear );
27. #endif *//DATE\_H\_INCLUDED*

### date.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include "date.h"
4. #include "funs.h"
5. int isLeapYear(int year)
6. {
7. int leap = 0;
8. if( year % 4 == 0 && year % 100 != 0 || year % 400 == 0 ){
9. leap = 1;
10. }
11. return leap;
12. }
13. int getDaySeqOnJan1(int year)
14. {
15. return (year + (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400) % 7;
16. }
17. int getMonthLength(int year, int month)
18. {
19. int monthLength = 0;
20. if(month == 1){
21. monthLength = 0;
22. } else if(month == 2){
23. monthLength = 31;
24. } else if(month == 3){
25. if(isLeapYear(year)){
26. monthLength = 31 + 29;
27. } else {
28. monthLength = 31 + 28;
29. }
30. } else if(month == 4){
31. monthLength = 31 + 29 + 31;
32. } else if(month == 5){
33. monthLength = 31 + 29 + 31 + 30;
34. } else if(month == 6){
35. monthLength = 31 + 29 + 31 + 30 + 31;
36. } else if(month == 7){
37. monthLength = 31 + 29 + 31 + 30 + 31 + 30;
38. } else if(month == 8){
39. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31;
40. } else if(month == 9){
41. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31;
42. } else if(month == 10){
43. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30;
44. } else if(month == 11){
45. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31;
46. } else if(month == 12){
47. monthLength = 31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 +30;
48. }
49. return monthLength;
50. }
51. int getDaySeq( int year, int month, int day )
52. {
53. return getMonthLength(year, month) + day;
54. }
55. int getWeekSeqOfYear(int year, int month, int day )
56. {
57. int daySeqOfYear = getDaySeq( year, month, day );
58. return (daySeqOfYear - ( 7 - getDaySeqOnJan1(year) + 1)) / 7 + 2;
59. }
60. int getMonth(int year, int daySeqOfYear )
61. {
62. int month = 1;
63. while(daySeqOfYear > getMonthLength(year, month + 1)){
64. month ++;
65. if(month == 12){
66. break;
67. }
68. }
69. return month;
70. }
71. int getDay(int year, int daySeqOfYear )
72. {
73. return daySeqOfYear - getMonthLength(year, getMonth(year, daySeqOfYear));
74. }
75. int getDaySeqOfWeek(int year, int daySeqOfYear )
76. {
77. return (getDaySeqOnJan1(year) + daySeqOfYear - 1) % 7;*//0:Sunday, 1:Monday, 2:Tuesday, 3:Wednesday, 4:Thursday, 5:Friday, 6:Saturday*
78. }
79. int getNextMonday(int year, int daySeqOfYear )
80. {
81. int daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
82. if(daySeqOfWeek == 0){
83. return daySeqOfYear + 1;
84. } else {
85. return daySeqOfYear + (7 - daySeqOfWeek) + 1;
86. }
87. }
88. int getThisMonday(int year, int daySeqOfYear )
89. {
90. int daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
91. if(daySeqOfWeek == 0){
92. return daySeqOfYear - 6;
93. } else {
94. return daySeqOfYear - daySeqOfWeek + 1;
95. }
96. }
97. int getThisSunday(int year, int daySeqOfYear )
98. {
99. int daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
100. if(daySeqOfWeek == 0){
101. return daySeqOfYear;
102. }else{
103. return daySeqOfYear + (7 - daySeqOfWeek);
104. }
105. }

### funs.h

1. #ifndef FUNS\_H\_INCLUDED
2. #define FUNS\_H\_INCLUDED
3. *//*
4. *//functions support school calendar displau*
5. void printOneDay(int year, int daySeqOfYear, int formatType);
6. void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow);
7. #endif *// FUNS\_H\_INCLUDED*

### funs.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include "date.h"
4. #include "funs.h"
5. #define DATE\_INFO\_BRIEF 1
6. #define DATE\_INFO\_FULL 2
7. void printOneDay(int year, int daySeqOfYear, int formatType)
8. {
9. int day, month, daySeqOfWeek;
10. day = getDay(year, daySeqOfYear);
11. month = getMonth(year, daySeqOfYear);
12. daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
13. if(formatType == 2){
14. if(daySeqOfWeek == 5 || daySeqOfWeek == 6){
15. printf(" %02d.%02d.%02d!", year % 100, month, day);
16. }else{
17. printf("  %02d.%02d.%02d", year % 100, month, day);
18. }
19. }else if(formatType == 1){
20. if(day == 1 && (daySeqOfWeek == 5 || daySeqOfWeek == 6) ){
21. printf("%4s%02d.%02d!", " ", month, day);
22. }else if(day == 1){
23. printf("%5s%02d.%02d", " ", month, day);
24. }else if(day != 1 && (daySeqOfWeek == 5 || daySeqOfWeek == 6) ){
25. printf("%9d!", day);
26. }else{
27. printf("%10d", day);
28. }
29. }
30. }
31. void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow)
32. {
33. printf("[%02d]", weekSeqShow);
34. int sStartDeqOfYear = 7 \* (weekSeqOfYear - 1) - getDaySeqOnJan1(year) + 1;
35. int sEndSeqOfYear = sStartDeqOfYear + 6;
36. int currentYearLength = (isLeapYear( year ))? 366 : 365;
37. int daySeqOfYear =  sStartDeqOfYear;
38. if( weekSeqShow == 1){
39. printOneDay(year, daySeqOfYear, DATE\_INFO\_FULL);
40. daySeqOfYear ++;
41. }
42. for( ; daySeqOfYear <= sEndSeqOfYear; daySeqOfYear ++){
43. if( daySeqOfYear <= currentYearLength ){
44. printOneDay(year, daySeqOfYear, DATE\_INFO\_BRIEF);
45. }else{
46. if(daySeqOfYear - currentYearLength == 1){
47. printOneDay(year + 1, daySeqOfYear - currentYearLength, DATE\_INFO\_FULL);
48. }else{
49. printOneDay(year + 1, daySeqOfYear - currentYearLength, DATE\_INFO\_BRIEF);
50. }
51. }
52. }
53. printf("\n");
54. }

## 实验八：不同年份秋季校历首周对比（Calendar08）

main.c

1. #include <stdio.h>
2. #define YEAR\_MIN 2000
3. #define YEAR\_MAX 2030
4. #define YEAR\_NUM 5
5. int monthLength[] = {0,31,0,31,30,31,30,31,31,30,31,30,31};
6. int isLeapYear(int year );
7. int getDaySeqOnJan1(int year);
8. int getDaySeq(int year, int month, int day);
9. int getDaySeqOfWeek(int year, int month, int day);
10. int getThisMonday(int year, int dayseqofYear);
11. void printOneWeek(int year, int weekSeqofYear, int weekseqshow);
12. int main()
13. {
14. int inputYear;
15. printf("Autumn Calendar Comparision in %d years\
16. ,plesase input the first year(%d-%d):\n",YEAR\_NUM,YEAR\_MIN,YEAR\_MAX - YEAR\_NUM);
17. scanf("%d",&inputYear);
18. if(inputYear < YEAR\_MIN || inputYear + YEAR\_NUM > YEAR\_MAX){
19. printf("the input year is not supported.\n");
20. return 1;
21. }
22. int i;
23. int Years[YEAR\_NUM] = {0};
24. int startDays[YEAR\_NUM] = {0};
25. int year;
26. year = inputYear;
27. for(i = 0; i < YEAR\_NUM;i ++){
28. Years[i] = year;
29. year ++;
30. printf("\n%s%s%d\n"," ", "First week in autumn Calendar of Year ", Years[i]);
31. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.","Tues.","Wed.","Thur.","Fri.","Sat.","Sun." );
32. printOneWeek(Years[i], getDaySeqOfWeek( Years[i], 9, 4),1);
33. startDays[i]= getThisMonday( Years[i], getDaySeq( Years[i], 9, 4));
34. }
35. int min = 0;
36. for(i = 1; i < YEAR\_NUM;i ++){
37. min = ( startDays[i] < startDays[min])? i : min;
38. }
39. printf("\n%s%s%d\n"," ","Earliest Spring Semester is in Year ",Years[min]);
40. return 0;
41. }
42. int isLeapYear(int year )
43. {
44. int IS\_LEAPYEAR;
45. IS\_LEAPYEAR = year % 4;
46. if((year % 4 == 0 && year % 100!= 0) || year % 400 == 0){
47. IS\_LEAPYEAR = 1;
48. return 1;
49. }else{
50. return 0;
51. }
52. }
53. int getDaySeqOnJan1(int year)
54. {
55. int WEEKDAY\_JAN1;
56. WEEKDAY\_JAN1 = (year + (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400) % 7;
57. return WEEKDAY\_JAN1;
58. }
59. int getDaySeq(int year, int month, int day)
60. {
61. if(isLeapYear(year)){
62. monthLength[2] = 29;
63. }else{
64. monthLength[2] = 28;
65. }
66. int i = 0;
67. int daySeq = 0;
68. for(i = 0;i < month;i ++){
69. daySeq += monthLength[i];
70. }
71. daySeq += day;
72. return daySeq;
73. }
74. int getDaySeqOfWeek(int year, int month, int day)
75. {
76. int weekSeq;
77. int daySeq;
78. int Jan1;
79. daySeq = getDaySeq(year,month,day);
80. Jan1 = getDaySeqOnJan1(year);
81. weekSeq = (Jan1 + daySeq - 1) % 7;
82. if(weekSeq == 0){
83. weekSeq = 7;
84. }
85. return weekSeq;
86. }
87. int getThisMonday(int year, int dayseqofYear)
88. {
89. int MondaySeqOfYear;
90. int daySeq;
91. int month = 9;
92. int day = 4;
93. daySeq = getDaySeqOfWeek(year,month,day);
94. MondaySeqOfYear = dayseqofYear - daySeq + 1;
95. return MondaySeqOfYear;
96. }
97. void printOneWeek(int year, int weekSeqofYear, int weekseqshow)
98. {
99. int simpleYear;
100. int month =  9;
101. int day = 4;
102. int dayfirst,monthfirst;
103. if(weekSeqofYear <= 4){
104. monthfirst = month;
105. dayfirst = day - weekSeqofYear + 1;
106. }else{
107. monthfirst = month - 1;
108. dayfirst = 31 - (weekSeqofYear - day) + 1;
109. }
110. simpleYear = year % 100;
111. printf("[01]%4d.0%d.%2d",simpleYear,monthfirst,dayfirst);
112. int i;
113. for(i = 0; i < 6; i ++){
114. dayfirst ++;
115. if(dayfirst == 32){
116. dayfirst = 1;
117. printf("%5s0%d.0%d"," ",month,dayfirst);
118. i ++;
119. }
120. printf("%10d",dayfirst);
121. }
122. printf("\n");
123. }

## 实验九：寻找生日【多维数组】(Calendar09)

main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define YEAR\_MIN 2016
4. #define YEAR\_MAX 2025
5. #define YEAR\_NUM 10
6. int isNormal(int inputMonth,int  inputDay);
7. void setYearArray(int Years[], int yearNum, int yearMin);
8. void initialDays(int Years[], int Days[][366][4], int yearNum);
9. int isLeapYear(int year);
10. int getDaySeqOnJan1(int year);
11. int getMonth(int year, int daySeq);
12. int getDay(int year, int daySeq);
13. int getDaySeq( int year, int month, int day );
14. int getWeekSeqOfYear(int year, int month, int day);
15. int getDaySeqOfWeek(int year, int daySeq);
16. void printWeeks(int year, int weekSeq, int month, int day);
17. void printOneDay(int year, int daySeqOfYear );
18. int main()
19. {
20. int inputMonth, inputDay, Years[YEAR\_NUM];
21. int Days[YEAR\_NUM][366][4] = {0};
22. printf("Finding Birthday in year (%d - %d), please input the month and day : \n", YEAR\_MIN, YEAR\_MAX);
23. scanf("%d-%d", &inputMonth, &inputDay);
24. if (! isNormal(inputMonth, inputDay)) {
25. printf("Invalid input.\n");
26. return -1;
27. }
28. setYearArray(Years, YEAR\_NUM, YEAR\_MIN);
29. initialDays(Years, Days, YEAR\_NUM);
30. int totalNum = 0, weekendNum = 0;
31. int i, j;
32. for (i = 0; i < YEAR\_NUM; i++) {
33. int year = Years[i];
34. int yearLength = isLeapYear(year)? 366 : 365;
35. printf("\nBirthday in Year %d\n", year);
36. printf(" #M:%10s%10s%10s%10s%10s%10s%10s\n", "MON.", "Tues.", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
37. if (inputMonth == 2 && inputDay == 29 && !isLeapYear(year)) {
38. printf("Not Found.\n");
39. }
40. else {
41. for (j = 0; j < (isLeapYear(year)? 366 : 365); j++) {
42. if (Days[i][j][0] == inputMonth && Days[i][j][1] == inputDay) {
43. totalNum++;
44. if (Days[i][j][3] == 0 || Days[i][j][3] == 6) {
45. weekendNum++;
46. }
47. }
48. }
49. printWeeks(year, getWeekSeqOfYear(year, inputMonth, inputDay), inputMonth, inputDay);
50. }
51. }
52. printf("Total %d birthdays are found, %d of them are in weekends.\n", totalNum, weekendNum);
53. return 0;
54. }
55. int isNormal(int inputMonth,int  inputDay)
56. {
57. int monthLength[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
58. if( inputMonth < 1 || inputMonth > 12 || inputDay < 1 || inputDay > monthLength[inputMonth - 1] ) {
59. return 0;
60. }
61. return 1;
62. }
63. void setYearArray(int Years[], int yearNum, int yearMin)
64. {
65. int i;
66. for (i = 0; i < yearNum; i++) {
67. Years[i] = yearMin++;
68. }
69. }
70. int isLeapYear(int year)
71. {
72. int leap = 0;
73. if( year % 4 == 0 && year % 100 != 0 || year % 400 == 0 ){
74. leap = 1;
75. }
76. return leap;
77. }
78. int getDaySeqOnJan1(int year)
79. {
80. return (year + (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400) % 7;
81. }
82. void initialDays(int Years[], int Days[][366][4], int yearNum)
83. {
84. int i, j;
85. for (i = 0; i < yearNum; i++) {
86. int year = Years[i];
87. int yearLength = isLeapYear(year)? 366 : 365;
88. for (j = 0; j < yearLength; j++) {
89. Days[i][j][0] = getMonth(year, j + 1);
90. Days[i][j][1] = getDay(year, j + 1);
91. Days[i][j][2] = getWeekSeqOfYear(year, Days[i][j][0], Days[i][j][1]);
92. Days[i][j][3] = getDaySeqOfWeek( year, j + 1 );
93. }
94. }
95. return;
96. }
97. int getMonth(int year, int daySeq)
98. {
99. int daysInMonth[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
100. if (isLeapYear(year)) {
101. daysInMonth[1] = 29;
102. }
103. int month = 0;
104. int day = daySeq;
105. while (day > daysInMonth[month]) {
106. day -= daysInMonth[month];
107. month++;
108. }
109. return month + 1;
110. }
111. int getDay(int year, int daySeq)
112. {
113. int daysInMonth[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
114. if (isLeapYear(year)) {
115. daysInMonth[1] = 29;
116. }
117. int month = 0;
118. int day = daySeq;
119. while (day > daysInMonth[month]) {
120. day -= daysInMonth[month];
121. month++;
122. }
123. return day;
124. }
125. int getDaySeq( int year, int month, int day )
126. {
127. int daysInMonth[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
128. if (isLeapYear(year)) {
129. daysInMonth[1] = 29;
130. }
131. int daySeq = 0;
132. int i;
133. for (i = 0; i < month - 1; i++) {
134. daySeq += daysInMonth[i];
135. }
136. daySeq += day;
137. return daySeq;
138. }
139. int getWeekSeqOfYear(int year, int month, int day)
140. {
141. int daySeqOfYear = getDaySeq( year, month, day );
142. return (daySeqOfYear - ( 7 - getDaySeqOnJan1(year) + 1)) / 7 + 2;
143. }
144. int getDaySeqOfWeek(int year, int daySeq)
145. {
146. int daySeqOfJan1 = (year + (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400) % 7;
147. return (daySeqOfJan1 + daySeq - 1) % 7;
148. }
149. void printWeeks(int year, int weekSeq, int month, int day)
150. {
151. printf("[%02d]", month);
152. int i;
153. int daySeqOfWeek = getDaySeqOfWeek(year, getDaySeq(year, month, day));
154. for (i = 0; i < 7; i++) {
155. printOneDay(year, getDaySeq(year, month, day - daySeqOfWeek + i + 1));
156. }
157. printf("\n");
158. }
159. void printOneDay(int year, int daySeqOfYear)
160. {
161. int day, month, daySeqOfWeek;
162. day = getDay(year, daySeqOfYear);
163. month = getMonth(year, daySeqOfYear);
164. daySeqOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
165. if(day == 1 && (daySeqOfWeek == 0 || daySeqOfWeek == 6) ){
166. printf("%4s%02d.%02d!", " ", month, day);
167. }else if(day == 1){
168. printf("%5s%02d.%02d", " ", month, day);
169. }else if(day != 1 && (daySeqOfWeek == 0 || daySeqOfWeek == 6) ){
170. printf("%9d!", day);
171. }else{
172. printf("%10d", day);
173. }
174. }

## 实验十：寻找生日【结构体】(Calendar10)

main.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define DATE\_NOSHOW 0
4. #define DATE\_INFO\_FULL 1
5. #define DATE\_INFO\_BRIEF 2
6. #define DATE\_STAR 3
7. #define YEAR\_START 2018
8. #define YEAR\_NUM 10
9. #define YEAR\_END (YEAR\_START + YEAR\_NUM - 1)
10. typedef struct{
11. int year;
12. int daySeq;
13. int month;
14. int day;
15. int weekSeq;
16. int weekDay;
17. } Day;
18. int isLeapYear(int year);*//leap year to return 1, else return 0*
19. int getDaySeqOnJan1(int year);
20. int getDaySeq(int year, int month, int day);
21. int getWeekSeqOfYear(int year, int month, int day);
22. int getMonth(int year, int daySeqOfYear);
23. int getDay(int year, int daySeqOfYear);
24. int getDaySeqOfWeek(int year, int daySeqOfYear);*//0 for Sunday, 1 for Monday, 6 for Saturday*
25. void setYearArray(int yearArray[], int yearNum, int yearStart);
26. int isDay(int year, int month, int day);
27. Day setDay(int year, int month, int day);
28. Day getDayBefore(Day currentDay, int interval);
29. Day getDayAfter(Day currentDay, int interval);
30. int getTwoDaysInterval(Day startDay, Day endDay);
31. void printDay(Day currentDay, int displayFormat);
32. int main()
33. {
34. Day currentDay;
35. int inputMonth, inputDay;
36. printf("Finding Birthday in year(%d-%d), please input month and day:\n", YEAR\_START, YEAR\_END);
37. scanf("%d-%d", &inputMonth, &inputDay);
38. int Years[YEAR\_NUM]; *// 2018-2027*
39. setYearArray(Years, YEAR\_NUM, YEAR\_START);
40. int i;
41. for(i = 0; i < YEAR\_NUM; i ++){
42. printf("\n%s%s%d\n", " ", "Birthday in Year ", Years[i]);
43. int ifPrint = isDay(Years[i], inputMonth, inputDay);
44. if(ifPrint != 1){
45. printf(" Not found.\n");
46. continue;
47. }
48. Day birthDay, prepareDay, printStartDay;
49. int printDayRange = 3;
50. birthDay = setDay(Years[i], inputMonth, inputDay); *// get the birthday*
51. prepareDay = getDayBefore(birthDay, printDayRange );      *// get the 3 day before the birthday*
52. printStartDay = getDayBefore(prepareDay, prepareDay.weekDay - 1);*// get the first day of the week*
53. printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tue.", "Wed.", "Thu.", "Fri.", "Sat.", "Sun.");
54. int displayFormat = 0;
55. if(birthDay.weekDay < 4){
56. printf(" [%02d]", birthDay.weekSeq - 1);
57. }else{
58. printf(" [%02d]", birthDay.weekSeq);
59. }
60. for( currentDay = printStartDay; getTwoDaysInterval(currentDay, birthDay) <= 0; currentDay = getDayAfter(currentDay, 1)){
61. if( getTwoDaysInterval(currentDay, prepareDay) < 0){
62. displayFormat = DATE\_NOSHOW;
63. }else if( getTwoDaysInterval(currentDay, prepareDay) == 0){
64. displayFormat = DATE\_INFO\_FULL;
65. }else if( getTwoDaysInterval(currentDay, birthDay) < 0){
66. displayFormat = DATE\_INFO\_BRIEF;
67. }else if( getTwoDaysInterval(currentDay, birthDay) == 0){
68. displayFormat = DATE\_STAR;
69. }
70. printDay(currentDay, displayFormat);
71. if( currentDay.weekDay == 7 && getTwoDaysInterval(currentDay, birthDay) != 0){
72. printf("\n [%02d]", birthDay.weekSeq);
73. }
74. }
75. }
76. return 0;
77. }
78. void setYearArray(int yearArray[], int yearNum, int yearStart){
79. int i;
80. for(i = 0; i < yearNum; i ++){
81. yearArray[i] = yearStart + i;
82. }
83. }
84. int isLeapYear(int year){
85. if ((year % 4 == 0 && year % 100 != 0) || year % 400 == 0){
86. return 1;
87. }
88. return 0;
89. }
90. int getDaySeqOnJan1(int year){
91. int daySeq;
92. daySeq = (year + (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400) % 7;
93. return daySeq;
94. }
95. int getDaySeq(int year, int month, int day){
96. int daySeq = 0;
97. int monthLength[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
98. int k = isLeapYear(year);
99. monthLength[2] += k;
100. int i;
101. for(i = 1; i < month; i ++){
102. daySeq += monthLength[i];
103. }
104. daySeq += day;
105. return daySeq;
106. }
107. int getWeekSeqOfYear(int year, int month, int day){
108. int daySeq = getDaySeq(year, month, day);
109. int daySeqOnJan1 = getDaySeqOnJan1(year);
110. int weekSeq;
111. if(day >= 1 && day <= 8 - daySeqOnJan1){
112. weekSeq = 1;
113. }
114. weekSeq = (daySeq + daySeqOnJan1 - 8) / 7 + 2;
115. return weekSeq;
116. }
117. int getMonth(int year, int daySeqOfYear){
118. int monthLength[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
119. monthLength[2] += isLeapYear(year);
120. int month;
121. for(month = 1; month <= 12; month ++){
122. if(daySeqOfYear <= monthLength[month]){
123. return month;
124. }
125. daySeqOfYear -= monthLength[month];
126. }
127. return 0;
128. }
129. int getDay(int year, int daySeqOfYear){
130. int monthLength[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
131. monthLength[2] += isLeapYear(year);
132. int month;
133. for(month = 1; month <= 12; month ++){
134. if(daySeqOfYear <= monthLength[month]){
135. return daySeqOfYear;
136. }
137. daySeqOfYear -= monthLength[month];
138. }
139. return 0;
140. }
141. int getDaySeqOfWeek(int year, int daySeqOfYear)
142. {
143. int daySeqOnJan1 = getDaySeqOnJan1(year);
144. int daySeqOfWeek = (daySeqOfYear + daySeqOnJan1 - 1) % 7  ;
145. if(daySeqOfWeek == 0){
146. daySeqOfWeek = 7;
147. }
148. return daySeqOfWeek;
149. }
150. int isDay(int year, int month, int day)
151. {
152. int monthLength[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
153. monthLength[2] += isLeapYear(year);
154. if(year < 1 || month < 1 || month > 12 || day < 1 || day > monthLength[month]){
155. return 0;
156. }
157. if(!isLeapYear(year) && month == 2 && day > 28){
158. return 0;
159. }
160. return 1;
161. }
162. Day setDay(int year, int month, int day)
163. {
164. Day day1;
165. day1.year = year;
166. day1.month = month;
167. day1.day = day;
168. day1.daySeq = getDaySeq(year, month, day);
169. day1.weekSeq = getWeekSeqOfYear(year, month, day);
170. day1.weekDay = getDaySeqOfWeek(year, day1.daySeq);
171. return day1;
172. }
173. Day getDayBefore(Day day, int interval)
174. {
175. int year = day.year;
176. int monthLength[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
177. monthLength[2] += isLeapYear(year);
178. int month = day.month;
179. int day1 = day.day;
181. day1 -= interval;
182. if(day1 < 1){
183. month --;
184. day1 += monthLength[month];
185. }
186. if(month < 1){   *//year - 1*
187. year --;
188. month = 12;
189. }
190. Day ReturnDay;
191. ReturnDay = setDay(year, month, day1);
193. return ReturnDay;
194. }
195. Day getDayAfter(Day day, int interval)
196. {
197. int year = day.year;
198. int monthLength[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
199. monthLength[2] += isLeapYear(year);
200. int month = day.month;
201. int day1 = day.day;
203. day1 += interval;
204. if(day1 > monthLength[month]){
205. day1 -= monthLength[month];
206. month ++;
207. }
208. if(month > 12){   *//year + 1*
209. year ++;
210. month = 1;
211. }
212. Day ReturnDay;
213. ReturnDay = setDay(year, month, day1);
214. return ReturnDay;
215. }
216. int getTwoDaysInterval(Day startDay, Day endDay)
217. {
218. return startDay.daySeq - endDay.daySeq;
219. }
220. void printDay(Day currentDay, int displayFormat)
221. {
223. if(displayFormat == 0){
224. printf("%10s", " ");
225. }else if(displayFormat == 1){
226. printf("  %02d.%02d.%02d", currentDay.year % 1000, currentDay.month, currentDay.day);
227. }else if(displayFormat == 2){
228. printf("%8s%02d", " ", currentDay.day);
229. }else if(displayFormat == 3){
230. printf("%4s%02d.%02d\*", " ", currentDay.month, currentDay.day);
231. }
232. }