華中科技大學

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　韦竞翔　　学号　U202413713

日期　2025.1.13　　地点　华中科技大学

成绩　　教师　刘威

1. 实验目的

完成日历系列代码。

2. 实验环境

操作系统：Windows 11

编程工具：Dev-C++

3. 实验一

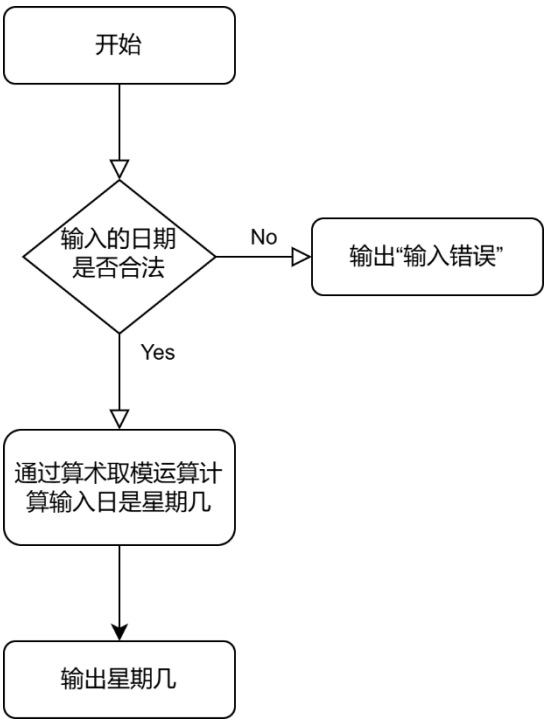
3.1 实验任务

1.运用所学到的知识，编写一个计算星期几的程序，计算 2024 年某月某日是星期几

2.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

3.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

3.2 实验步骤



3.3 代码测试

测试点 1：输入“1 1”，预期输出“1”

实际测试结果：

```
请输入月份与日期：1 1
1
```

测试结论：达到目标

测试点 2：输入“7 11”，预期输出“4”

实际测试结果：

```
请输入月份与日期：7 11
4
```

测试结论：达到目标

测试点 3：输入“2 30”，预期输出“输入错误”

实际测试结果：
```
请输入月份与日期：2 30
输入错误
```

测试结论：达到目标

测试点 4：输入“13 4”，预期输出“输入错误”

实际测试结果：
```
请输入月份与日期：13 4
输入错误
```

测试结论：达到目标

3.4 实验结论

代码达到功能目标

3.5 实验总结

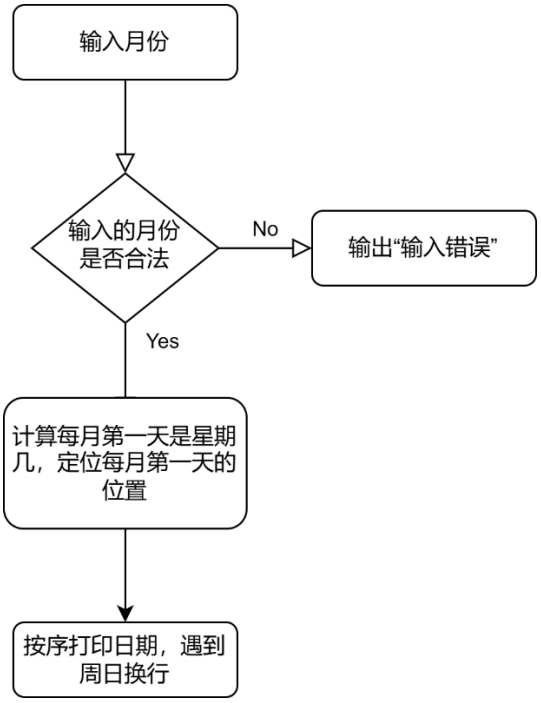
学会了基础的 C 语言语法和计算星期几的方法

4. 实验二

4.1 实验任务

1.运用所学到的知识，编写打印月历的程序，打印 2024 年 1 到 12 月的某个月的月历
2.约定每个星期从周一开始
3.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

4.2 实验步骤



4.3 代码测试

测试点 1：输入“1”，预期输出

```
Mo Tu We Th Fr Sa Su
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

实际测试结果：

```
请输入月份：1
Mo Tu We Th Fr Sa Su
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

测试结论：达到目标

测试点 2：输入“12”，预期输出

```
Mo Tu We Th Fr Sa Su
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31
```

实际测试结果：

```
请输入月份：12
Mo Tu We Th Fr Sa Su
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31
```

测试结论：达到目标

测试点 3：输入“13”，预期输出“输入错误”

实际测试结果：

```
请输入月份：13
输入错误
```

测试结论：达到目标

测试点 4：输入“0”，预期输出“输入错误”

实际测试结果：

```
请输入月份：0
输入错误
```

测试结论：达到目标

4.4 实验结论

代码达到功能目标

4.5 实验总结

学会了基础的 C 语言语法和打印月历的方法

5. 实验三

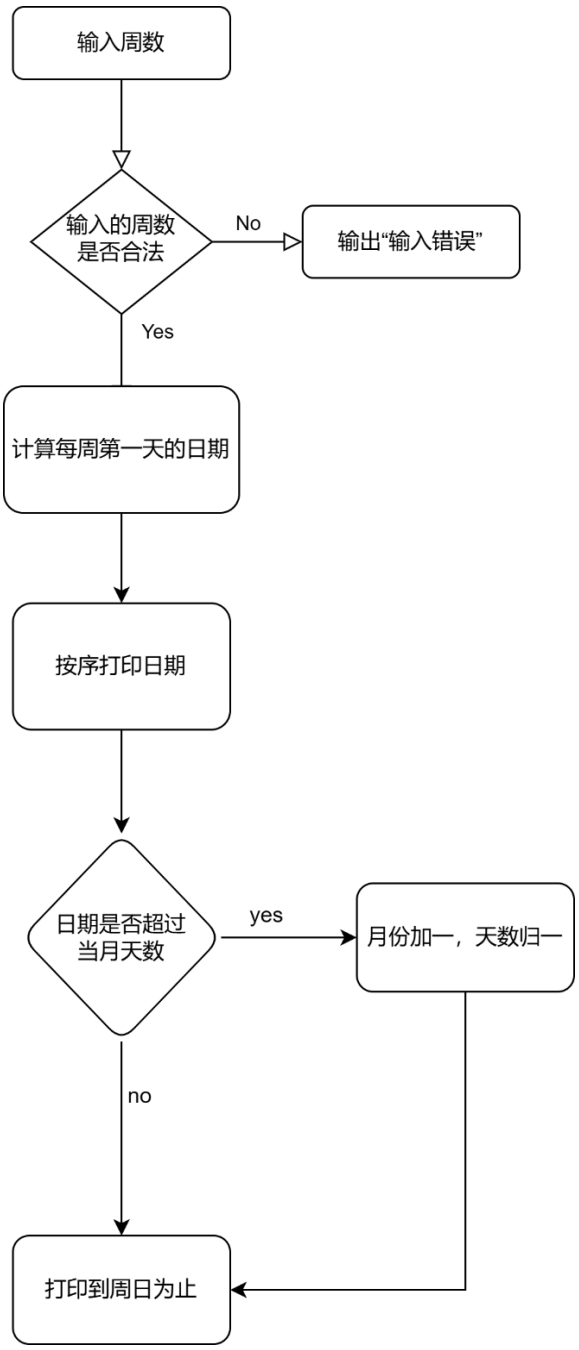
5.1 实验任务

1.运用所学到的知识，编写一个打印周历的程序，打印 2024 年某一周的周历
2.约定每个星期从周一开始

3.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

4.第 1 周和第 53 周中，仅打印 2024 年的日期

5.2 实验步骤



5.3 代码测试

测试点 1：输入“-1”，预期输出“输入错误”

实际测试结果：

```
请输入周数：-1
输入错误
```

测试结论：达到目标

测试点 2：输入“54”，预期输出“输入错误”

实际测试结果：
```
请输入周数：54
输入错误
```

测试结论：达到目标

测试点 3：输入“1”，预期输出
```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
01: 01.01 01.02 01.03 01.04 01.05 01.06 01.07
```

实际测试结果：
```
请输入周数：1
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
01: 01.01 01.02 01.03 01.04 01.05 01.06 01.07
```

测试结论：达到目标

测试点 4：输入“5”，预期输出
```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
05: 01.29 01.30 01.31 02.01 02.02 02.03 02.04
```

实际测试结果：
```
请输入周数：5
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
05: 01.29 01.30 01.31 02.01 02.02 02.03 02.04
```

测试结论：达到目标

测试点 5：输入“53”，预期输出
```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
53: 12.30 12.31
```

实际测试结果：
```
请输入周数：53
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
53: 12.30 12.31
```

测试结论：达到目标

5.4 实验结论

代码达到功能目标

5.5 实验总结

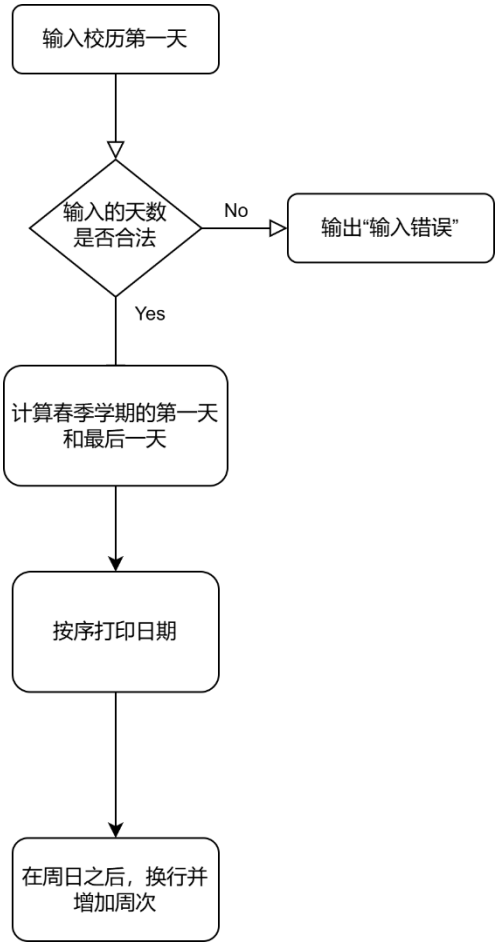
学会了基础的 C 语言语法和打印周历的方法

6. 实验四

6.1 实验任务

1. 打印 2024 年春季学期的校历
2. 春季学期从 2 月份的某一周开始，用户输入 2 月份的某日作为参考日：如果该日就是周一，即从当周开始；如果该日不是周一，即从下周开始。
3. 春季学期到 7 月份的第一个周末结束。
4. 约定校历的每列的宽度为 10 个字符，可以在 printf 语句中用%5s 打印空格、用 %02d 打印月或者日的数字

6.2 实验步骤



6.3 代码测试

测试点 1：输入“1”，预期输出

```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]02.05 02.06 02.07 02.08 02.09 02.10 02.11
[02]02.12 02.13 02.14 02.15 02.16 02.17 02.18
[03]02.19 02.20 02.21 02.22 02.23 02.24 02.25
[04]02.26 02.27 02.28 02.29 03.01 03.02 03.03
[05]03.04 03.05 03.06 03.07 03.08 03.09 03.10
[06]03.11 03.12 03.13 03.14 03.15 03.16 03.17
[07]03.18 03.19 03.20 03.21 03.22 03.23 03.24
[08]03.25 03.26 03.27 03.28 03.29 03.30 03.31
[09]04.01 04.02 04.03 04.04 04.05 04.06 04.07
[10]04.08 04.09 04.10 04.11 04.12 04.13 04.14
[11]04.15 04.16 04.17 04.18 04.19 04.20 04.21
[12]04.22 04.23 04.24 04.25 04.26 04.27 04.28
[13]04.29 04.30 05.01 05.02 05.03 05.04 05.05
[14]05.06 05.07 05.08 05.09 05.10 05.11 05.12
[15]05.13 05.14 05.15 05.16 05.17 05.18 05.19
[16]05.20 05.21 05.22 05.23 05.24 05.25 05.26
[17]05.27 05.28 05.29 05.30 05.31 06.01 06.02
[18]06.03 06.04 06.05 06.06 06.07 06.08 06.09
[19]06.10 06.11 06.12 06.13 06.14 06.15 06.16
[20]06.17 06.18 06.19 06.20 06.21 06.22 06.23
[21]06.24 06.25 06.26 06.27 06.28 06.29 06.30
[22]07.01 07.02 07.03 07.04 07.05 07.06 07.07
```

实际测试结果：

```
请输入校历第一天：1
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]02.05 02.06 02.07 02.08 02.09 02.10 02.11
[02]02.12 02.13 02.14 02.15 02.16 02.17 02.18
[03]02.19 02.20 02.21 02.22 02.23 02.24 02.25
[04]02.26 02.27 02.28 02.29 03.01 03.02 03.03
[05]03.04 03.05 03.06 03.07 03.08 03.09 03.10
[06]03.11 03.12 03.13 03.14 03.15 03.16 03.17
[07]03.18 03.19 03.20 03.21 03.22 03.23 03.24
[08]03.25 03.26 03.27 03.28 03.29 03.30 03.31
[09]04.01 04.02 04.03 04.04 04.05 04.06 04.07
[10]04.08 04.09 04.10 04.11 04.12 04.13 04.14
[11]04.15 04.16 04.17 04.18 04.19 04.20 04.21
[12]04.22 04.23 04.24 04.25 04.26 04.27 04.28
[13]04.29 04.30 05.01 05.02 05.03 05.04 05.05
[14]05.06 05.07 05.08 05.09 05.10 05.11 05.12
[15]05.13 05.14 05.15 05.16 05.17 05.18 05.19
[16]05.20 05.21 05.22 05.23 05.24 05.25 05.26
[17]05.27 05.28 05.29 05.30 05.31 06.01 06.02
[18]06.03 06.04 06.05 06.06 06.07 06.08 06.09
[19]06.10 06.11 06.12 06.13 06.14 06.15 06.16
[20]06.17 06.18 06.19 06.20 06.21 06.22 06.23
[21]06.24 06.25 06.26 06.27 06.28 06.29 06.30
[22]07.01 07.02 07.03 07.04 07.05 07.06 07.07
```

测试结论：达到目标

测试点 2：输入“26”，预期输出

```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]02.26 02.27 02.28 02.29 03.01 03.02 03.03
[02]03.04 03.05 03.06 03.07 03.08 03.09 03.10
[03]03.11 03.12 03.13 03.14 03.15 03.16 03.17
[04]03.18 03.19 03.20 03.21 03.22 03.23 03.24
[05]03.25 03.26 03.27 03.28 03.29 03.30 03.31
[06]04.01 04.02 04.03 04.04 04.05 04.06 04.07
[07]04.08 04.09 04.10 04.11 04.12 04.13 04.14
[08]04.15 04.16 04.17 04.18 04.19 04.20 04.21
[09]04.22 04.23 04.24 04.25 04.26 04.27 04.28
[10]04.29 04.30 05.01 05.02 05.03 05.04 05.05
[11]05.06 05.07 05.08 05.09 05.10 05.11 05.12
[12]05.13 05.14 05.15 05.16 05.17 05.18 05.19
[13]05.20 05.21 05.22 05.23 05.24 05.25 05.26
[14]05.27 05.28 05.29 05.30 05.31 06.01 06.02
[15]06.03 06.04 06.05 06.06 06.07 06.08 06.09
[16]06.10 06.11 06.12 06.13 06.14 06.15 06.16
[17]06.17 06.18 06.19 06.20 06.21 06.22 06.23
[18]06.24 06.25 06.26 06.27 06.28 06.29 06.30
[19]07.01 07.02 07.03 07.04 07.05 07.06 07.07
```

实际测试结果：

```
请输入校历第一天：26
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]02.26 02.27 02.28 02.29 03.01 03.02 03.03
[02]03.04 03.05 03.06 03.07 03.08 03.09 03.10
[03]03.11 03.12 03.13 03.14 03.15 03.16 03.17
[04]03.18 03.19 03.20 03.21 03.22 03.23 03.24
[05]03.25 03.26 03.27 03.28 03.29 03.30 03.31
[06]04.01 04.02 04.03 04.04 04.05 04.06 04.07
[07]04.08 04.09 04.10 04.11 04.12 04.13 04.14
[08]04.15 04.16 04.17 04.18 04.19 04.20 04.21
[09]04.22 04.23 04.24 04.25 04.26 04.27 04.28
[10]04.29 04.30 05.01 05.02 05.03 05.04 05.05
[11]05.06 05.07 05.08 05.09 05.10 05.11 05.12
[12]05.13 05.14 05.15 05.16 05.17 05.18 05.19
[13]05.20 05.21 05.22 05.23 05.24 05.25 05.26
[14]05.27 05.28 05.29 05.30 05.31 06.01 06.02
[15]06.03 06.04 06.05 06.06 06.07 06.08 06.09
[16]06.10 06.11 06.12 06.13 06.14 06.15 06.16
[17]06.17 06.18 06.19 06.20 06.21 06.22 06.23
[18]06.24 06.25 06.26 06.27 06.28 06.29 06.30
[19]07.01 07.02 07.03 07.04 07.05 07.06 07.07
```

测试结论：达到目标

测试点 3：输入“0”，预期输出“输入错误”

实际测试结果：

```
请输入校历第一天：0
输入错误
```

测试结论：达到目标

测试点 4：输入“30”，预期输出“输入错误”

实际测试结果：

```
请输入校历第一天：30
输入错误
```

测试结论：达到目标

6.4 实验结论

代码达到功能目标

6.5 实验总结

学会了函数的使用，体会到模块化编程的好处

7. 实验五

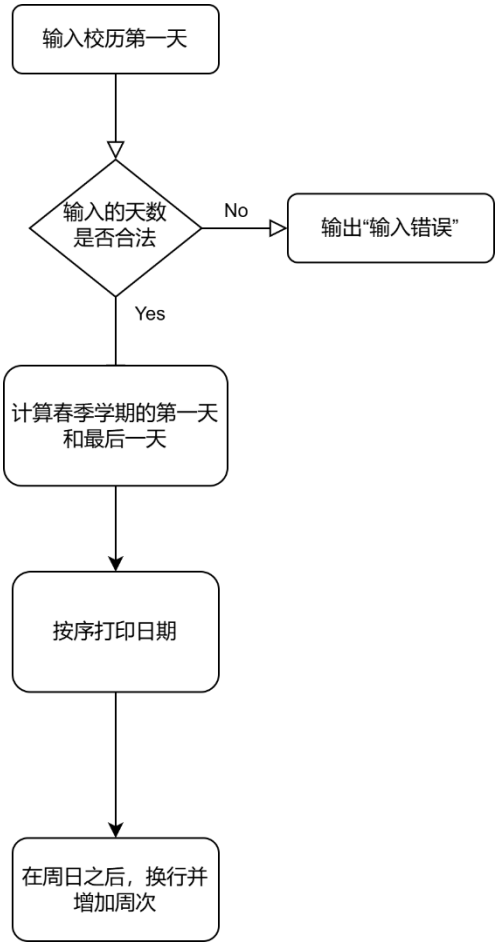
7.1 实验任务

1.用函数改写打印华中科技大学校历的程序 打印 2024 年春季学期的校历

2.开发下列函数：

```
// 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);
```

7.2 实验步骤



7.3 代码测试

测试点 1：输入“1”，预期输出

```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]02.05 02.06 02.07 02.08 02.09 02.10 02.11
[02]02.12 02.13 02.14 02.15 02.16 02.17 02.18
[03]02.19 02.20 02.21 02.22 02.23 02.24 02.25
[04]02.26 02.27 02.28 02.29 03.01 03.02 03.03
[05]03.04 03.05 03.06 03.07 03.08 03.09 03.10
[06]03.11 03.12 03.13 03.14 03.15 03.16 03.17
[07]03.18 03.19 03.20 03.21 03.22 03.23 03.24
[08]03.25 03.26 03.27 03.28 03.29 03.30 03.31
[09]04.01 04.02 04.03 04.04 04.05 04.06 04.07
[10]04.08 04.09 04.10 04.11 04.12 04.13 04.14
[11]04.15 04.16 04.17 04.18 04.19 04.20 04.21
[12]04.22 04.23 04.24 04.25 04.26 04.27 04.28
[13]04.29 04.30 05.01 05.02 05.03 05.04 05.05
[14]05.06 05.07 05.08 05.09 05.10 05.11 05.12
[15]05.13 05.14 05.15 05.16 05.17 05.18 05.19
[16]05.20 05.21 05.22 05.23 05.24 05.25 05.26
[17]05.27 05.28 05.29 05.30 05.31 06.01 06.02
[18]06.03 06.04 06.05 06.06 06.07 06.08 06.09
[19]06.10 06.11 06.12 06.13 06.14 06.15 06.16
[20]06.17 06.18 06.19 06.20 06.21 06.22 06.23
[21]06.24 06.25 06.26 06.27 06.28 06.29 06.30
[22]07.01 07.02 07.03 07.04 07.05 07.06 07.07
```

实际测试结果：

```
请输入校历第一天：1
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]02.05 02.06 02.07 02.08 02.09 02.10 02.11
[02]02.12 02.13 02.14 02.15 02.16 02.17 02.18
[03]02.19 02.20 02.21 02.22 02.23 02.24 02.25
[04]02.26 02.27 02.28 02.29 03.01 03.02 03.03
[05]03.04 03.05 03.06 03.07 03.08 03.09 03.10
[06]03.11 03.12 03.13 03.14 03.15 03.16 03.17
[07]03.18 03.19 03.20 03.21 03.22 03.23 03.24
[08]03.25 03.26 03.27 03.28 03.29 03.30 03.31
[09]04.01 04.02 04.03 04.04 04.05 04.06 04.07
[10]04.08 04.09 04.10 04.11 04.12 04.13 04.14
[11]04.15 04.16 04.17 04.18 04.19 04.20 04.21
[12]04.22 04.23 04.24 04.25 04.26 04.27 04.28
[13]04.29 04.30 05.01 05.02 05.03 05.04 05.05
[14]05.06 05.07 05.08 05.09 05.10 05.11 05.12
[15]05.13 05.14 05.15 05.16 05.17 05.18 05.19
[16]05.20 05.21 05.22 05.23 05.24 05.25 05.26
[17]05.27 05.28 05.29 05.30 05.31 06.01 06.02
[18]06.03 06.04 06.05 06.06 06.07 06.08 06.09
[19]06.10 06.11 06.12 06.13 06.14 06.15 06.16
[20]06.17 06.18 06.19 06.20 06.21 06.22 06.23
[21]06.24 06.25 06.26 06.27 06.28 06.29 06.30
[22]07.01 07.02 07.03 07.04 07.05 07.06 07.07
```

测试结论：达到目标

```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]02.26 02.27 02.28 02.29 03.01 03.02 03.03
[02]03.04 03.05 03.06 03.07 03.08 03.09 03.10
[03]03.11 03.12 03.13 03.14 03.15 03.16 03.17
[04]03.18 03.19 03.20 03.21 03.22 03.23 03.24
[05]03.25 03.26 03.27 03.28 03.29 03.30 03.31
[06]04.01 04.02 04.03 04.04 04.05 04.06 04.07
[07]04.08 04.09 04.10 04.11 04.12 04.13 04.14
[08]04.15 04.16 04.17 04.18 04.19 04.20 04.21
[09]04.22 04.23 04.24 04.25 04.26 04.27 04.28
[10]04.29 04.30 05.01 05.02 05.03 05.04 05.05
[11]05.06 05.07 05.08 05.09 05.10 05.11 05.12
[12]05.13 05.14 05.15 05.16 05.17 05.18 05.19
[13]05.20 05.21 05.22 05.23 05.24 05.25 05.26
[14]05.27 05.28 05.29 05.30 05.31 06.01 06.02
[15]06.03 06.04 06.05 06.06 06.07 06.08 06.09
[16]06.10 06.11 06.12 06.13 06.14 06.15 06.16
[17]06.17 06.18 06.19 06.20 06.21 06.22 06.23
[18]06.24 06.25 06.26 06.27 06.28 06.29 06.30
[19]07.01 07.02 07.03 07.04 07.05 07.06 07.07
```

测试点 2：输入“26”，预期输出

```
请输入校历第一天：26
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]02.26 02.27 02.28 02.29 03.01 03.02 03.03
[02]03.04 03.05 03.06 03.07 03.08 03.09 03.10
[03]03.11 03.12 03.13 03.14 03.15 03.16 03.17
[04]03.18 03.19 03.20 03.21 03.22 03.23 03.24
[05]03.25 03.26 03.27 03.28 03.29 03.30 03.31
[06]04.01 04.02 04.03 04.04 04.05 04.06 04.07
[07]04.08 04.09 04.10 04.11 04.12 04.13 04.14
[08]04.15 04.16 04.17 04.18 04.19 04.20 04.21
[09]04.22 04.23 04.24 04.25 04.26 04.27 04.28
[10]04.29 04.30 05.01 05.02 05.03 05.04 05.05
[11]05.06 05.07 05.08 05.09 05.10 05.11 05.12
[12]05.13 05.14 05.15 05.16 05.17 05.18 05.19
[13]05.20 05.21 05.22 05.23 05.24 05.25 05.26
[14]05.27 05.28 05.29 05.30 05.31 06.01 06.02
[15]06.03 06.04 06.05 06.06 06.07 06.08 06.09
[16]06.10 06.11 06.12 06.13 06.14 06.15 06.16
[17]06.17 06.18 06.19 06.20 06.21 06.22 06.23
[18]06.24 06.25 06.26 06.27 06.28 06.29 06.30
[19]07.01 07.02 07.03 07.04 07.05 07.06 07.07
```

实际测试结果：

测试结论：达到目标

测试点 3：输入“0”，预期输出“输入错误”

```
请输入校历第一天：0
输入错误
```

实际测试结果：

测试结论：达到目标

测试点 4：输入“30”，预期输出“输入错误”

```
请输入校历第一天：30
输入错误
```

实际测试结果：

测试结论：达到目标

7.4 实验结论

代码达到功能目标

7.5 实验总结

学会了函数的使用，体会到模块化编程的好处

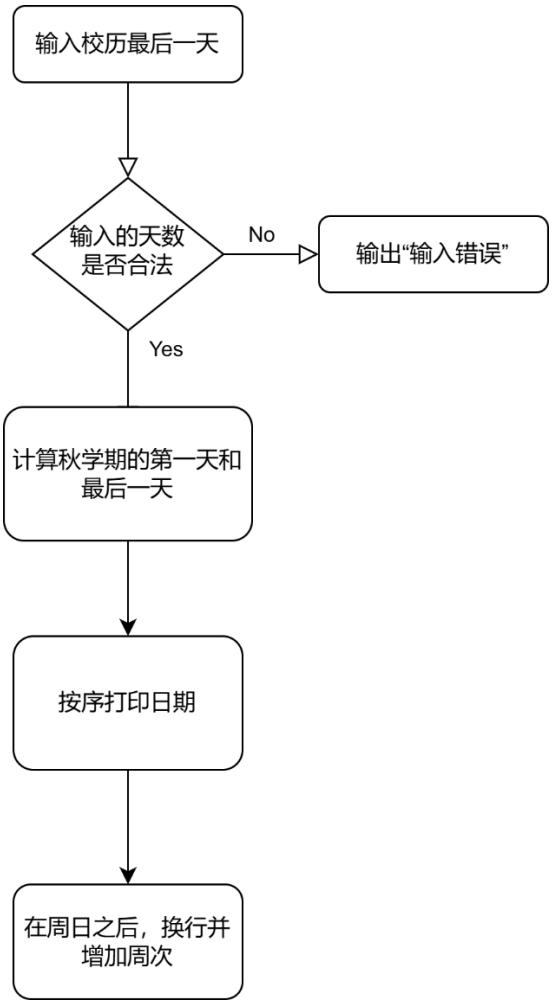
8. 实验六

8.1 实验任务

1.用函数改写打印华中科技大学校历的程序，打印 2024 年秋季学期的校历。
2.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。
3.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。

```
// Functions on different year
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  getNextMonday(int year, int daySeqOfYear);
int  getThisMonday(int year, int daySeqOfYear);
int  getThisSunday(int year, int daySeqOfYear);
//
// Functions support school calendar display
void printOneDay(int year, int daySeqOfYear);
```

8.2 实验步骤



8.3 代码测试

测试点 1：输入“0”，预期输出“输入错误”

实际测试结果：

```
请输入校历最后一天：0
输入错误
```

测试结论：达到目标

测试点 2：输入“32”，预期输出“输入错误”

实际测试结果：

```
请输入校历最后一天：32
输入错误
```

测试结论：达到目标

测试点 3：输入“5”，预期输出

```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]08.26 08.27 08.28 08.29 08.30 08.31 09.01
[02]09.02 09.03 09.04 09.05 09.06 09.07 09.08
[03]09.09 09.10 09.11 09.12 09.13 09.14 09.15
[04]09.16 09.17 09.18 09.19 09.20 09.21 09.22
[05]09.23 09.24 09.25 09.26 09.27 09.28 09.29
[06]09.30 10.01 10.02 10.03 10.04 10.05 10.06
[07]10.07 10.08 10.09 10.10 10.11 10.12 10.13
[08]10.14 10.15 10.16 10.17 10.18 10.19 10.20
[09]10.21 10.22 10.23 10.24 10.25 10.26 10.27
[10]10.28 10.29 10.30 10.31 11.01 11.02 11.03
[11]11.04 11.05 11.06 11.07 11.08 11.09 11.10
[12]11.11 11.12 11.13 11.14 11.15 11.16 11.17
[13]11.18 11.19 11.20 11.21 11.22 11.23 11.24
[14]11.25 11.26 11.27 11.28 11.29 11.30 12.01
[15]12.02 12.03 12.04 12.05 12.06 12.07 12.08
[16]12.09 12.10 12.11 12.12 12.13 12.14 12.15
[17]12.16 12.17 12.18 12.19 12.20 12.21 12.22
[18]12.23 12.24 12.25 12.26 12.27 12.28 12.29
[19]12.30 12.31 01.01 01.02 01.03 01.04 01.05
```

实际测试结果：

```
请输入校历最后一天：5
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]08.26 08.27 08.28 08.29 08.30 08.31 09.01
[02]09.02 09.03 09.04 09.05 09.06 09.07 09.08
[03]09.09 09.10 09.11 09.12 09.13 09.14 09.15
[04]09.16 09.17 09.18 09.19 09.20 09.21 09.22
[05]09.23 09.24 09.25 09.26 09.27 09.28 09.29
[06]09.30 10.01 10.02 10.03 10.04 10.05 10.06
[07]10.07 10.08 10.09 10.10 10.11 10.12 10.13
[08]10.14 10.15 10.16 10.17 10.18 10.19 10.20
[09]10.21 10.22 10.23 10.24 10.25 10.26 10.27
[10]10.28 10.29 10.30 10.31 11.01 11.02 11.03
[11]11.04 11.05 11.06 11.07 11.08 11.09 11.10
[12]11.11 11.12 11.13 11.14 11.15 11.16 11.17
[13]11.18 11.19 11.20 11.21 11.22 11.23 11.24
[14]11.25 11.26 11.27 11.28 11.29 11.30 12.01
[15]12.02 12.03 12.04 12.05 12.06 12.07 12.08
[16]12.09 12.10 12.11 12.12 12.13 12.14 12.15
[17]12.16 12.17 12.18 12.19 12.20 12.21 12.22
[18]12.23 12.24 12.25 12.26 12.27 12.28 12.29
[19]12.30 12.31 01.01 01.02 01.03 01.04 01.05
```

测试结论：达到目标

测试点 4：输入“6”，预期输出

```
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]08.26 08.27 08.28 08.29 08.30 08.31 09.01
[02]09.02 09.03 09.04 09.05 09.06 09.07 09.08
[03]09.09 09.10 09.11 09.12 09.13 09.14 09.15
[04]09.16 09.17 09.18 09.19 09.20 09.21 09.22
[05]09.23 09.24 09.25 09.26 09.27 09.28 09.29
[06]09.30 10.01 10.02 10.03 10.04 10.05 10.06
[07]10.07 10.08 10.09 10.10 10.11 10.12 10.13
[08]10.14 10.15 10.16 10.17 10.18 10.19 10.20
[09]10.21 10.22 10.23 10.24 10.25 10.26 10.27
[10]10.28 10.29 10.30 10.31 11.01 11.02 11.03
[11]11.04 11.05 11.06 11.07 11.08 11.09 11.10
[12]11.11 11.12 11.13 11.14 11.15 11.16 11.17
[13]11.18 11.19 11.20 11.21 11.22 11.23 11.24
[14]11.25 11.26 11.27 11.28 11.29 11.30 12.01
[15]12.02 12.03 12.04 12.05 12.06 12.07 12.08
[16]12.09 12.10 12.11 12.12 12.13 12.14 12.15
[17]12.16 12.17 12.18 12.19 12.20 12.21 12.22
[18]12.23 12.24 12.25 12.26 12.27 12.28 12.29
[19]12.30 12.31 01.01 01.02 01.03 01.04 01.05
[20]01.06 01.07 01.08 01.09 01.10 01.11 01.12
```

实际测试结果：

```
请输入校历最后一天：6
#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.
[01]08.26 08.27 08.28 08.29 08.30 08.31 09.01
[02]09.02 09.03 09.04 09.05 09.06 09.07 09.08
[03]09.09 09.10 09.11 09.12 09.13 09.14 09.15
[04]09.16 09.17 09.18 09.19 09.20 09.21 09.22
[05]09.23 09.24 09.25 09.26 09.27 09.28 09.29
[06]09.30 10.01 10.02 10.03 10.04 10.05 10.06
[07]10.07 10.08 10.09 10.10 10.11 10.12 10.13
[08]10.14 10.15 10.16 10.17 10.18 10.19 10.20
[09]10.21 10.22 10.23 10.24 10.25 10.26 10.27
[10]10.28 10.29 10.30 10.31 11.01 11.02 11.03
[11]11.04 11.05 11.06 11.07 11.08 11.09 11.10
[12]11.11 11.12 11.13 11.14 11.15 11.16 11.17
[13]11.18 11.19 11.20 11.21 11.22 11.23 11.24
[14]11.25 11.26 11.27 11.28 11.29 11.30 12.01
[15]12.02 12.03 12.04 12.05 12.06 12.07 12.08
[16]12.09 12.10 12.11 12.12 12.13 12.14 12.15
[17]12.16 12.17 12.18 12.19 12.20 12.21 12.22
[18]12.23 12.24 12.25 12.26 12.27 12.28 12.29
[19]12.30 12.31 01.01 01.02 01.03 01.04 01.05
[20]01.06 01.07 01.08 01.09 01.10 01.11 01.12
```

测试结论：达到目标

8.4 实验结论

代码达到功能目标

8.5 实验总结

学会了函数的使用，体会到模块化编程的好处

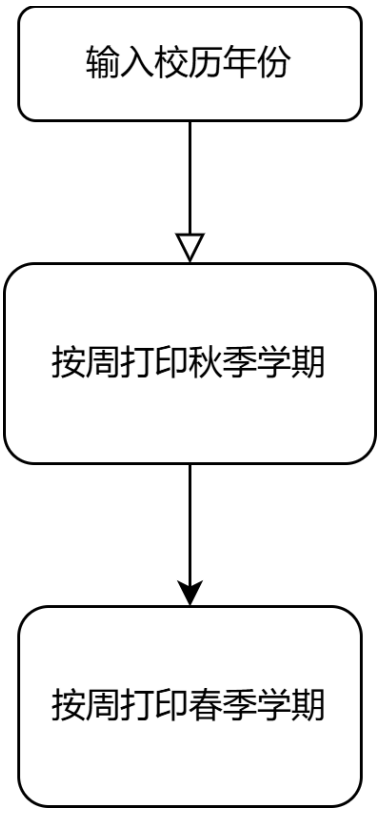
9. 实验七

9.1 实验任务

1. 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
2. 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。
3. 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。

```
//
// Functions support school calendar display
void printOneDay(int year, int daySeqOfYear, int formatType);
void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow);
```

9.2 实验步骤



9.3 代码测试

测试点 1：输入“2024”，预期输出同实际测试结果

实际测试结果：

```
请输入学期年份：2024
        First Semester (Fall) Calendar of Year 2024 - 2025
 #W:      Mon.      Tues.      Wed.      Thur.      Fri.      Sat.      Sun.
[36]     09.02     09.03     09.04     09.05     09.06     09.07     09.08
[37]     09.09     09.10     09.11     09.12     09.13     09.14     09.15
[38]     09.16     09.17     09.18     09.19     09.20     09.21     09.22
[39]     09.23     09.24     09.25     09.26     09.27     09.28     09.29
[40]     09.30     10.01     10.02     10.03     10.04     10.05     10.06
[41]     10.07     10.08     10.09     10.10     10.11     10.12     10.13
[42]     10.14     10.15     10.16     10.17     10.18     10.19     10.20
[43]     10.21     10.22     10.23     10.24     10.25     10.26     10.27
[44]     10.28     10.29     10.30     10.31     11.01     11.02     11.03
[45]     11.04     11.05     11.06     11.07     11.08     11.09     11.10
[46]     11.11     11.12     11.13     11.14     11.15     11.16     11.17
[47]     11.18     11.19     11.20     11.21     11.22     11.23     11.24
[48]     11.25     11.26     11.27     11.28     11.29     11.30     12.01
[49]     12.02     12.03     12.04     12.05     12.06     12.07     12.08
[50]     12.09     12.10     12.11     12.12     12.13     12.14     12.15
[51]     12.16     12.17     12.18     12.19     12.20     12.21     12.22
[52]     12.23     12.24     12.25     12.26     12.27     12.28     12.29
[53]     12.30     12.31     01.01     01.02     01.03     01.04     01.05
[54]     01.06     01.07     01.08     01.09     01.10     01.11     01.12
[55]     01.13     01.14     01.15     01.16     01.17     01.18     01.19
[56]     01.20     01.21     01.22     01.23     01.24     01.25     01.26
```

```
        Second Semester (Spring) Calendar of Year 2024 - 2025
 #W:      Mon.      Tues.      Wed.      Thur.      Fri.      Sat.      Sun.
[06]     02.03     02.04     02.05     02.06     02.07     02.08     02.09
[07]     02.10     02.11     02.12     02.13     02.14     02.15     02.16
[08]     02.17     02.18     02.19     02.20     02.21     02.22     02.23
[09]     02.24     02.25     02.26     02.27     02.28     03.01     03.02
[10]     03.03     03.04     03.05     03.06     03.07     03.08     03.09
[11]     03.10     03.11     03.12     03.13     03.14     03.15     03.16
[12]     03.17     03.18     03.19     03.20     03.21     03.22     03.23
[13]     03.24     03.25     03.26     03.27     03.28     03.29     03.30
[14]     03.31     04.01     04.02     04.03     04.04     04.05     04.06
[15]     04.07     04.08     04.09     04.10     04.11     04.12     04.13
[16]     04.14     04.15     04.16     04.17     04.18     04.19     04.20
[17]     04.21     04.22     04.23     04.24     04.25     04.26     04.27
[18]     04.28     04.29     04.30     05.01     05.02     05.03     05.04
[19]     05.05     05.06     05.07     05.08     05.09     05.10     05.11
[20]     05.12     05.13     05.14     05.15     05.16     05.17     05.18
[21]     05.19     05.20     05.21     05.22     05.23     05.24     05.25
[22]     05.26     05.27     05.28     05.29     05.30     05.31     06.01
[23]     06.02     06.03     06.04     06.05     06.06     06.07     06.08
[24]     06.09     06.10     06.11     06.12     06.13     06.14     06.15
[25]     06.16     06.17     06.18     06.19     06.20     06.21     06.22
[26]     06.23     06.24     06.25     06.26     06.27     06.28     06.29
```

测试结论：达到目标

测试点 2：输入“2020”，预期输出同实际测试结果

实际测试结果：

```
请输入学期年份：2020
        First Semester (Fall) Calendar of Year 2020 - 2021
 #W:      Mon.      Tues.      Wed.      Thur.      Fri.      Sat.      Sun.
[35]     08.24     08.25     08.26     08.27     08.28     08.29     08.30
[36]     08.31     09.01     09.02     09.03     09.04     09.05     09.06
[37]     09.07     09.08     09.09     09.10     09.11     09.12     09.13
[38]     09.14     09.15     09.16     09.17     09.18     09.19     09.20
[39]     09.21     09.22     09.23     09.24     09.25     09.26     09.27
[40]     09.28     09.29     09.30     10.01     10.02     10.03     10.04
[41]     10.05     10.06     10.07     10.08     10.09     10.10     10.11
[42]     10.12     10.13     10.14     10.15     10.16     10.17     10.18
[43]     10.19     10.20     10.21     10.22     10.23     10.24     10.25
[44]     10.26     10.27     10.28     10.29     10.30     10.31     11.01
[45]     11.02     11.03     11.04     11.05     11.06     11.07     11.08
[46]     11.09     11.10     11.11     11.12     11.13     11.14     11.15
[47]     11.16     11.17     11.18     11.19     11.20     11.21     11.22
[48]     11.23     11.24     11.25     11.26     11.27     11.28     11.29
[49]     11.30     12.01     12.02     12.03     12.04     12.05     12.06
[50]     12.07     12.08     12.09     12.10     12.11     12.12     12.13
[51]     12.14     12.15     12.16     12.17     12.18     12.19     12.20
[52]     12.21     12.22     12.23     12.24     12.25     12.26     12.27
[53]     12.28     12.29     12.30     12.31     01.01     01.02     01.03
[54]     01.04     01.05     01.06     01.07     01.08     01.09     01.10
[55]     01.11     01.12     01.13     01.14     01.15     01.16     01.17
```

```
        Second Semester (Spring) Calendar of Year 2020 - 2021
 #W:      Mon.      Tues.      Wed.      Thur.      Fri.      Sat.      Sun.
[07]     02.08     02.09     02.10     02.11     02.12     02.13     02.14
[08]     02.15     02.16     02.17     02.18     02.19     02.20     02.21
[09]     02.22     02.23     02.24     02.25     02.26     02.27     02.28
[10]     03.01     03.02     03.03     03.04     03.05     03.06     03.07
[11]     03.08     03.09     03.10     03.11     03.12     03.13     03.14
[12]     03.15     03.16     03.17     03.18     03.19     03.20     03.21
[13]     03.22     03.23     03.24     03.25     03.26     03.27     03.28
[14]     03.29     03.30     03.31     04.01     04.02     04.03     04.04
[15]     04.05     04.06     04.07     04.08     04.09     04.10     04.11
[16]     04.12     04.13     04.14     04.15     04.16     04.17     04.18
[17]     04.19     04.20     04.21     04.22     04.23     04.24     04.25
[18]     04.26     04.27     04.28     04.29     04.30     05.01     05.02
[19]     05.03     05.04     05.05     05.06     05.07     05.08     05.09
[20]     05.10     05.11     05.12     05.13     05.14     05.15     05.16
[21]     05.17     05.18     05.19     05.20     05.21     05.22     05.23
[22]     05.24     05.25     05.26     05.27     05.28     05.29     05.30
[23]     05.31     06.01     06.02     06.03     06.04     06.05     06.06
[24]     06.07     06.08     06.09     06.10     06.11     06.12     06.13
[25]     06.14     06.15     06.16     06.17     06.18     06.19     06.20
[26]     06.21     06.22     06.23     06.24     06.25     06.26     06.27
```

测试结论：达到目标

测试点 2：输入“2005”，预期输出同实际测试结果

实际测试结果：

```
请输入学期年份：2005
            First Semester (Fall) Calendar of Year 2005 - 2006
 #W:      Mon.      Tues.      Wed.      Thur.      Fri.      Sat.      Sun.
[35]     08.22     08.23     08.24     08.25     08.26     08.27     08.28
[36]     08.29     08.30     08.31     09.01     09.02     09.03     09.04
[37]     09.05     09.06     09.07     09.08     09.09     09.10     09.11
[38]     09.12     09.13     09.14     09.15     09.16     09.17     09.18
[39]     09.19     09.20     09.21     09.22     09.23     09.24     09.25
[40]     09.26     09.27     09.28     09.29     09.30     10.01     10.02
[41]     10.03     10.04     10.05     10.06     10.07     10.08     10.09
[42]     10.10     10.11     10.12     10.13     10.14     10.15     10.16
[43]     10.17     10.18     10.19     10.20     10.21     10.22     10.23
[44]     10.24     10.25     10.26     10.27     10.28     10.29     10.30
[45]     10.31     11.01     11.02     11.03     11.04     11.05     11.06
[46]     11.07     11.08     11.09     11.10     11.11     11.12     11.13
[47]     11.14     11.15     11.16     11.17     11.18     11.19     11.20
[48]     11.21     11.22     11.23     11.24     11.25     11.26     11.27
[49]     11.28     11.29     11.30     12.01     12.02     12.03     12.04
[50]     12.05     12.06     12.07     12.08     12.09     12.10     12.11
[51]     12.12     12.13     12.14     12.15     12.16     12.17     12.18
[52]     12.19     12.20     12.21     12.22     12.23     12.24     12.25
[53]     12.26     12.27     12.28     12.29     12.30     12.31     01.01
[54]     01.02     01.03     01.04     01.05     01.06     01.07     01.08
[55]     01.09     01.10     01.11     01.12     01.13     01.14     01.15
```

```
            Second Semester (Spring) Calendar of Year 2005 - 2006
 #W:      Mon.      Tues.      Wed.      Thur.      Fri.      Sat.      Sun.
[07]     02.06     02.07     02.08     02.09     02.10     02.11     02.12
[08]     02.13     02.14     02.15     02.16     02.17     02.18     02.19
[09]     02.20     02.21     02.22     02.23     02.24     02.25     02.26
[10]     02.27     02.28     03.01     03.02     03.03     03.04     03.05
[11]     03.06     03.07     03.08     03.09     03.10     03.11     03.12
[12]     03.13     03.14     03.15     03.16     03.17     03.18     03.19
[13]     03.20     03.21     03.22     03.23     03.24     03.25     03.26
[14]     03.27     03.28     03.29     03.30     03.31     04.01     04.02
[15]     04.03     04.04     04.05     04.06     04.07     04.08     04.09
[16]     04.10     04.11     04.12     04.13     04.14     04.15     04.16
[17]     04.17     04.18     04.19     04.20     04.21     04.22     04.23
[18]     04.24     04.25     04.26     04.27     04.28     04.29     04.30
[19]     05.01     05.02     05.03     05.04     05.05     05.06     05.07
[20]     05.08     05.09     05.10     05.11     05.12     05.13     05.14
[21]     05.15     05.16     05.17     05.18     05.19     05.20     05.21
[22]     05.22     05.23     05.24     05.25     05.26     05.27     05.28
[23]     05.29     05.30     05.31     06.01     06.02     06.03     06.04
[24]     06.05     06.06     06.07     06.08     06.09     06.10     06.11
[25]     06.12     06.13     06.14     06.15     06.16     06.17     06.18
[26]     06.19     06.20     06.21     06.22     06.23     06.24     06.25
```

测试结论：达到目标

9.4 实验结论

代码达到功能目标

9.5 实验总结

学会了函数的使用，体会到模块化编程的好处

10. 实验八

10.1 实验任务

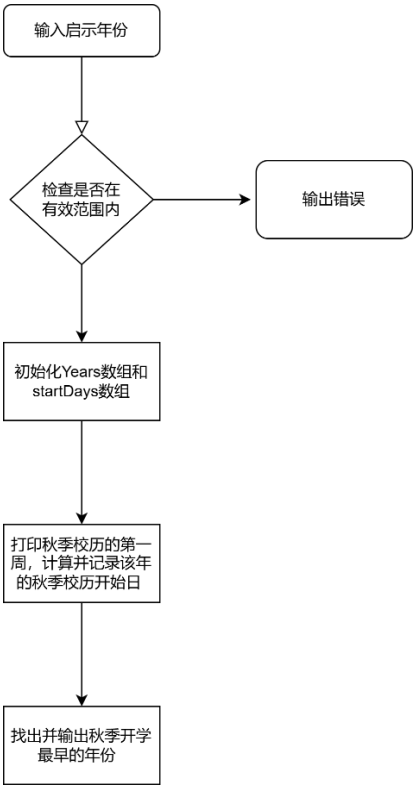
1.用数组改写打印华中科技大学校历的程序，具备打印校历的能力，比较多个年份的校历秋季校历的首日，寻找开学最早的那一年。

2.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。

3.基于数组改进现有的日历程序，改进计算日期的函数

```
//
// 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);
```

10.2 实验步骤



10.3 代码测试

测试点 1：输入“2024”，预期输出如下图

实际测试结果：

```
Fall Calendar Comparison in 5 years, please input the first year (2000-2035):
2024

 First week in spring Calendar of Year 2024
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[36]     09.02     09.03     09.04     09.05     09.06     09.07     09.08

 First week in spring Calendar of Year 2025
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[35]     08.25     08.26     08.27     08.28     08.29     08.30     08.31

 First week in spring Calendar of Year 2026
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[35]     08.24     08.25     08.26     08.27     08.28     08.29     08.30

 First week in spring Calendar of Year 2027
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[35]     08.23     08.24     08.25     08.26     08.27     08.28     08.29

 First week in spring Calendar of Year 2028
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[36]     08.28     08.29     08.30     08.31     09.01     09.02     09.03

 Earliest Spring Semester is in Year 2027
```

测试结论：达到目标

测试点 2：输入“2019”，预期输出如下图

实际测试结果：

```
Fall Calendar Comparison in 5 years, please input the first year (2000-2035):
2019

 First week in spring Calendar of Year 2019
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[36]     09.02     09.03     09.04     09.05     09.06     09.07     09.08

 First week in spring Calendar of Year 2020
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[35]     08.24     08.25     08.26     08.27     08.28     08.29     08.30

 First week in spring Calendar of Year 2021
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[35]     08.23     08.24     08.25     08.26     08.27     08.28     08.29

 First week in spring Calendar of Year 2022
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[35]     08.22     08.23     08.24     08.25     08.26     08.27     08.28

 First week in spring Calendar of Year 2023
 #W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[36]     08.28     08.29     08.30     08.31     09.01     09.02     09.03

 Earliest Spring Semester is in Year 2022
```

测试结论：达到目标

测试点 3：输入“2036”，预期输出“输入错误”

实际测试结果：
```
Fall Calendar Comparison in 5 years, please input the first year (2000-2035):
2036
Sorry, the input year is not supported.
```

测试结论：达到目标

测试点 4：输入“1999”，预期输出“输入错误”

实际测试结果：
```
Fall Calendar Comparison in 5 years, please input the first year (2000-2035):
1999
Sorry, the input year is not supported.
```

测试结论：达到目标

10.4 实验结论

代码达到功能目标

10.5 实验总结

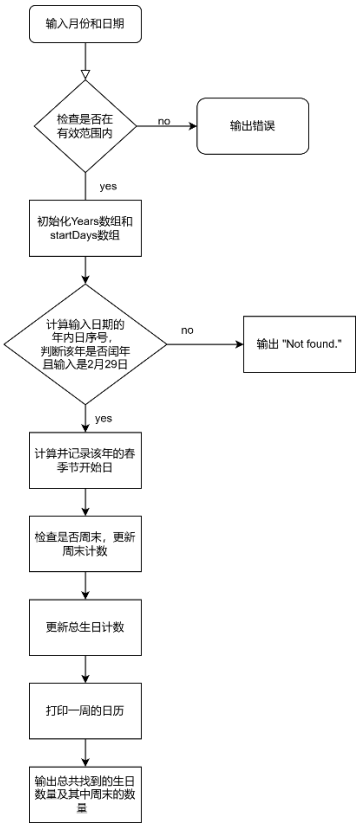
学会了一维数组的使用，体会到数据结构的好处

11. 实验九

11.1 实验任务

1.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

11.2 实验步骤



11.3 代码测试

测试点 1：输入“2 29”，预期输出如图

实际测试结果：

```
Finding Birthday in year (2020-2025), please input the month and day:
2 29

Birthday in Year 2020
#M:     Mon.    Tues.     Wed.    Thur.     Fri.     Sat.     Sun.
[08]    02.24    02.25    02.26    02.27    02.28    02.29    03.01

Birthday in Year 2021
Not found.

Birthday in Year 2022
Not found.

Birthday in Year 2023
Not found.

Birthday in Year 2024
#M:     Mon.    Tues.     Wed.    Thur.     Fri.     Sat.     Sun.
[09]    02.26    02.27    02.28    02.29    03.01    03.02    03.03

Birthday in Year 2025
Not found.

Total 2 birthdays are found, 1 of them are in weekends.
```

测试结论：达到目标

测试点 2：输入“7 11”，预期输出如图

实际测试结果：

```
Finding Birthday in year (2020-2025), please input the month and day:
7 11

Birthday in Year 2020
#M:     Mon.    Tues.     Wed.    Thur.     Fri.     Sat.     Sun.
[27]    07.06    07.07    07.08    07.09    07.10    07.11    07.12

Birthday in Year 2021
#M:     Mon.    Tues.     Wed.    Thur.     Fri.     Sat.     Sun.
[27]    07.05    07.06    07.07    07.08    07.09    07.10    07.11

Birthday in Year 2022
#M:     Mon.    Tues.     Wed.    Thur.     Fri.     Sat.     Sun.
[28]    07.11    07.12    07.13    07.14    07.15    07.16    07.17

Birthday in Year 2023
#M:     Mon.    Tues.     Wed.    Thur.     Fri.     Sat.     Sun.
[28]    07.10    07.11    07.12    07.13    07.14    07.15    07.16

Birthday in Year 2024
#M:     Mon.    Tues.     Wed.    Thur.     Fri.     Sat.     Sun.
[28]    07.08    07.09    07.10    07.11    07.12    07.13    07.14

Birthday in Year 2025
#M:     Mon.    Tues.     Wed.    Thur.     Fri.     Sat.     Sun.
[27]    07.07    07.08    07.09    07.10    07.11    07.12    07.13

Total 6 birthdays are found, 2 of them are in weekends.
```

测试结论：达到目标

测试点 3：输入“2 30”，预期输出“输入错误”

实际测试结果：

```
Finding Birthday in year (2020-2025), please input the month and day:
2 30
Sorry, the input month and day are invalid.
```

测试结论：达到目标

11.4 实验结论

代码达到功能目标

11.5 实验总结

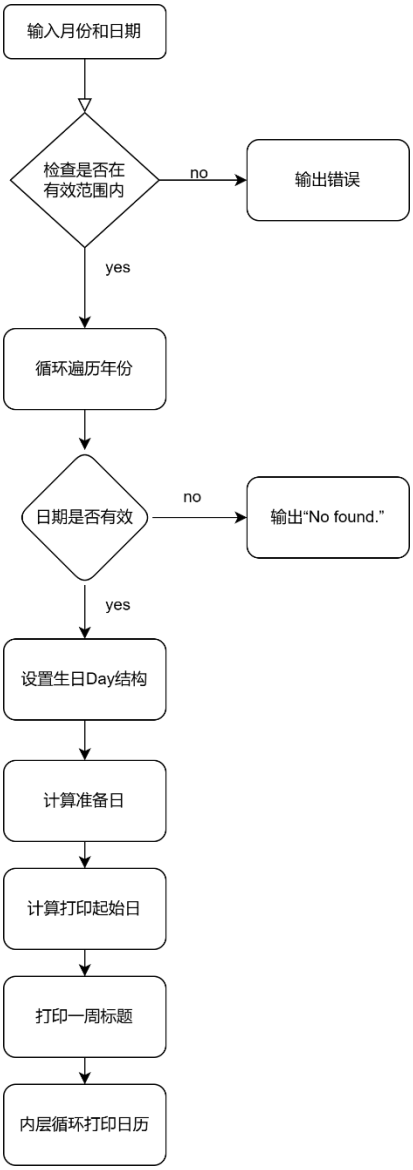
学会了多维数组的使用，体会到数据结构的好处

12. 实验十

12.1 实验任务

1. 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
2. 用日期结构体记录

12.2 实验步骤



12.3 代码测试

测试点 1：输入“2 29”，预期输出如下图

实际测试结果：

```
Finding Birthday in year (2023-2025), please input the month and day:
2 29

 Birthday in Year 2023
Not found.

 Birthday in Year 2024
#W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.
[09]2024.02.26       27       28     02.29*

 Birthday in Year 2025
Not found.
```

测试结论：达到目标

测试点 2：输入“2 30”，预期输出“输入错误”

实际测试结果：

```
Finding Birthday in year (2023-2025), please input the month and day:
2 30
Input wrong.
```

测试结论：达到目标

测试点 3：输入“7 11”，预期输出如下图

实际测试结果：

```
Finding Birthday in year (2023-2025), please input the month and day:
7 11

 Birthday in Year 2023
#W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.

[26]                                                 2023.07.08        9
[27]       10      07.11*

 Birthday in Year 2024
#W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.

[27]2024.07.08        9       10      07.11*

 Birthday in Year 2025
#W:      Mon.     Tues.      Wed.     Thur.      Fri.      Sat.      Sun.

[26]          2025.07.08        9       10      07.11*
```

测试结论：达到目标

测试点 4：输入“13 4”，预期输出“输入错误”

实际测试结果：

```
Finding Birthday in year (2023-2025), please input the month and day:
13 50
Input wrong.
```

测试结论：达到目标

12.4 实验结论

代码达到功能目标

12.5 实验总结

学会了结构体的使用，体会到数据结构的好处

13. 本课程学习总结

我第一次接触 C 语言是在 C 语言课程上，当时看着那些密密麻麻的代码，心里既好奇又忐忑。老师在讲台上滔滔不绝地讲解着变量、数据类型等基础概念，我却像在听天书。课上，我打开编译器，尝试着敲下人生中第一段代码——“Hello, World!”程序。看着屏幕上成功弹出的那行字，内心竟有了一丝小小的成就感，仿佛打开了新世界的大门。

刚开始学习时，我对 C 语言的语法一知半解，常常犯一些低级错误。比如在定义变量时，忘记写分号，导致编译器报错；还有忘记切换中英文导致标点符号出错。从那以后，我养成了仔细检查代码的好习惯，每写一行代码，都会认真检查语法是否正确。

我还尝尝犯逻辑错误。逻辑错误比语法错误更隐蔽，也更难调试。有一次，我写了一个排序程序，代码看起来没有语法错误，也能正常编译运行，但排序的结果却总是不对。我反复检查代码，发现是在循环条件设置上出了问题，导致有些元素没有被正确比较和交换。为了解决这个问题，我在代码中添加了大量的打印语句，逐步跟踪程序的执行过程，最终找到了问题所在。这次经历让我明白，逻辑错误需要仔细分析程序的逻辑流程，通过调试手段来逐步排查。

随着学习的深入，我逐渐意识到代码规范的重要性。良好的代码规范可以让代码更易读、易维护。比如，在命名变量和函数时，我开始采用有意义的命名方式，而不是简单地用 a、b、c 等字母，更不会用中文拼音（除非在考试的时候记不起来英文怎么写）。例如，用 studentScore 来表示学生的成绩，用 calculateAverage 来表示计算平均值的函数，这样一看就能明白变量和函数的用途。

同时，我也注意到了代码缩进和空格的使用。正确的缩进可以让代码的层次结构更加清晰，方便我们理解代码的逻辑。在编写循环和条件语句时，我会严格遵循缩进规范，每进入一层代码块就缩进一定的空格，这样代码看起来整洁美观，也更容易发现逻辑错误。

在调试程序时，我学会了一些实用的技巧。首先是使用打印语句来调试。当程序出现错误时，我会在关键位置添加打印语句，输出变量的值和程序的执行流程，通过观察打印结果来判断程序是否按照预期执行。例如，在一个复杂的算法实现中，我会在每个关键步骤打印出中间变量的值，这样就能清楚地看到程序的运行状态，及时发现逻辑错误。

其次，我还学会了使用调试工具。像 GDB 这样的调试工具，可以让我们更方便地设置断点、单步执行程序、查看变量值等。通过使用调试工具，我可以更精确地定位问题所在，大大提高了调试效率。比如，在调试一个多线程程序时，GDB 的多线程调试功能就帮了我大忙，让我能够清楚地看到每个线程的执行状态和变量值，顺利解决了程序中的同步问题。

在学习初期，我主要通过教材和在线课程来学习 C 语言的基础知识。每天都会花几个小时的时间来阅读教材，从变量、数据类型、运算符等基础概念学起，逐步掌握了 C 语言的基本语法。同时，我还会在在线学习平台上观看一些优质的 C 语言教学视频，通过视频中的实例讲解，加深对知识点的理解。

为了巩固所学知识，我不仅会完成老师布置的头歌在线编程题，还会自己去洛谷等网站找题写，平常也会和同学们讨论一些有趣的知识和题目，这段经历让我受益匪浅，极大提高了我的编程能力。在电工基地，我也经常用 C 语言完成嵌入式项目，这不仅让我领会到 C 语言的魅力，更在实践中加强了我对 C 语言的理解。

学习 C 语言的过程充满了挫折和困难，但正是这些挫折让我更加坚定了学习的决心。每当遇到一个难题，我都会告诉自己不能放弃，要耐心地去分析问题、解决问题。通过一次

次的坚持和努力，我逐渐克服了学习中的困难，也积累了解决问题的经验。我深刻地认识到，学习编程就像爬山，虽然过程艰难，但只要坚持不懈，就一定能看到山顶的风景。

理论知识固然重要，但只有通过实践才能真正掌握 C 语言。在做项目的过程中，我将所学的理论知识应用到实际问题中，通过解决实际问题来加深对知识点的理解。实践让我学会了如何将复杂的现实问题转化为编程问题，如何设计合理的程序来解决问题。同时，实践也让我发现了自己在学习过程中存在的不足，促使我不断改进和提高。

在学习过程中，我养成了总结和反思的好习惯。每当完成一个项目或解决一个难题后，我都会认真总结经验教训，思考自己在学习过程中有哪些做得好的地方，哪些地方还需要改进。通过总结和反思，我能够不断调整学习方法，提高学习效率。

回顾这段学习 C 语言的历程，虽然充满了心酸与挫折，但也让我收获了成长与喜悦。从最初的懵懂无知到现在的基本掌握，C 语言已经成为了我编程道路上的重要基石。在今后的学习和工作中，我将继续努力，不断探索和学习新的编程技术和知识，用 C 语言为我开启更广阔的技术世界。

14. 附录

14.1 实验一“计算星期几”代码

```c
#include <stdio.h>
#include <stdlib.h>

int calculateWeekday(int month, int day);

int main(void) {
    int month, day;
    int weekday;

    printf("请输入月份与日期: ");
    scanf("%d %d", &month, &day);

    if (month == 1 && day >= 1 && day <= 31 ||
        month == 2 && day >= 1 && day <= 29 ||
        month == 3 && day >= 1 && day <= 31 ||
        month == 4 && day >= 1 && day <= 30 ||
        month == 5 && day >= 1 && day <= 31 ||
        month == 6 && day >= 1 && day <= 30 ||
        month == 7 && day >= 1 && day <= 31 ||
        month == 8 && day >= 1 && day <= 31 ||
        month == 9 && day >= 1 && day <= 30 ||
        month == 10 && day >= 1 && day <= 31 ||
        month == 11 && day >= 1 && day <= 30 ||
        month == 12 && day >= 1 && day <= 31) {
        weekday = calculateWeekday(month, day);
        printf("%d", weekday);
    } else {
        printf("输入错误");
    }

    return 0;
}

int calculateWeekday(int month, int day) {
    int daysInMonth[] = {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    int totalDays = 0, weekday;
```

```
    int i;

    //计算这天前的总天数
    for (i = 1; i < month; ++i) {
        totalDays += daysInMonth[i];
    }
    totalDays += day;
    //取模计算星期几
    weekday = (totalDays + 6) % 7 + 1;

    return weekday;
}
```

14.2 实验二“打印月历”代码

```
#include <stdio.h>
#include <stdlib.h>

int calculateWeekday(int month, int day);
void printMonth(int month);

int main(void) {
    int month;

    printf("请输入月份: ");
    scanf("%d",&month);

    if(month >= 1 && month <= 12){
        printMonth(month);
    }else{
        printf("输入错误");
    }

    return 0;
}

//计算每月第一天是星期几
int calculateWeekday(int month, int day) {
    int daysInMonth[] = {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    int totalDays = 0, weekday;
    int i;

    //计算这天前的总天数
    for (i = 1; i < month; ++i) {
        totalDays += daysInMonth[i];
    }
    totalDays += day;
    //取模计算星期几
    weekday = (totalDays + 6) % 7 + 1;

    return weekday;
}

//打印月历
void printMonth(int month) {
    int daysInMonth[] = {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    int i, j, firstDayOfWeek, dayOfWeek, dayOfMonth;

    //计算每月第一天是星期几
    firstDayOfWeek = calculateWeekday(month, 1);

    //打印月历头
```

```
    printf("Mo Tu We Th Fr Sa Su\n");

    //打印每个月日期前空出的空格
    for (i = 1; i < firstDayOfWeek; i++) {
        printf("   ");
    }

    //打印日期
    for (dayOfMonth = 1; dayOfMonth <= daysInMonth[month]; dayOfMonth
++) {
        printf("%2d ", dayOfMonth);

        dayOfWeek = (firstDayOfWeek + dayOfMonth - 1) % 7;
        if (dayOfWeek == 0) {
            printf("\n");
        }
    }
}
```

14.3 实验三 “打印周历”代码

```
#include <stdio.h>
#include <stdlib.h>

int calculateWeekday(int month, int day);
void printWeek(int week);

int main(void) {
    int week;

    printf("请输入周数: ");
    scanf("%d",&week);

    //2024 有 53 周
    if(week > 0 && week <= 53){
        printWeek(week);
    }else{
        printf("输入错误");
    }

    return 0;
}

//计算每周第一天是星期几
int calculateWeekday(int month, int day) {
    int daysInMonth[] = {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 3
0, 31};
    int totalDays = 0, weekday;
    int i;

    //计算这天前的总天数
    for (i = 1; i < month; ++i) {
        totalDays += daysInMonth[i];
    }
    totalDays += day;
    //取模计算星期几
    weekday = (totalDays + 6) % 7 + 1;

    return weekday;
}

//打印周历
void printWeek(int week) {
```

```
    int daysInMonth[] = {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    int month = 1;
    int day = (week - 1) * 7 + 1;

    //计算周一的日期
    while(month <= 12 && day > daysInMonth[month]){
        day -= daysInMonth[month];
        month ++;
    }

    int firstDayOfWeek = calculateWeekday(month, day);

    //打印周历头
    printf("#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.\n");
    printf("%02d: ", week);

    //打印每个周日期前空出的空格
    int i;
    for (i = 1; i < firstDayOfWeek; i++) {
        printf("    ");
    }

    //打印日期
    for (i = 1; i <= 7 - firstDayOfWeek + 1; i++) {
        if(day > daysInMonth[month]){
            month ++;
            day = 1;
        }
        if(month <= 12){
            printf("%02d.%02d ", month, day);
        }
        day ++;
    }

}
```

14.4 实验四“打印校历(2024 春)”代码

```
#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(){
    int referenceDay;
    printf("请输入校历第一天：");
    scanf("%d",&referenceDay);
    if(referenceDay > 0 && referenceDay <= 29){
```

```
25.         // 计算春季学期的第一天和最后一天
26.         int sStartSeqOfYear = getNextMonday(getDaySeq(2, referenceDay
    ));
27.         int sEndSeqOfYear = getThisSunday(getDaySeq(7, 1));
28.
29.         printf("#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.\n");
30.
31.         // 遍历这个学期的每一天
32.         int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;
33.         for (daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0, weekSe
    qOfSemester = 1;
34.              daySeqOfYear <= sEndSeqOfYear;
35.              daySeqOfYear++, daySeqOfWeek++, daySeqOfWeek %= 7){
36.             // 在周一之前，打印周次
37.             if (daySeqOfWeek == 0){
38.                 printf("[%02d]", weekSeqOfSemester);
39.             }
40.
41.             // 打印当前日期
42.             printOneDay(daySeqOfYear);
43.
44.             // 在周日之后，换行并增加周次
45.             if (daySeqOfWeek == 6){
46.                 printf("\n");
47.                 weekSeqOfSemester++;
48.             }
49.         }
50.     }else{
51.         printf("输入错误");
52.     }
53. }
54.
55. int getMonthLength(int month) {
56.     int daysInMonth[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30,
    31};
57.     return daysInMonth[month - 1];
58. }
59.
60. int getDaySeq(int month, int day) {
61.     int totalDays = 0;
62.     int i;
63.     for (i = 1; i < month; ++i) {
64.         totalDays += getMonthLength(i);
65.     }
66.     totalDays += day;
67.     return totalDays;
68. }
69.
70. int getMonth(int daySeqOfYear) {
71.     int month = 1;
72.     while(month <= 12 && daySeqOfYear > getMonthLength(month)){
73.         daySeqOfYear -= getMonthLength(month);
74.         month ++;
75.     }
76.     return month;
77. }
78.
79. int getDay(int daySeqOfYear) {
80.     int month = 1;
81.     while (month <= 12 && daySeqOfYear > getMonthLength(month)){
82.         daySeqOfYear -= getMonthLength(month);
83.         month++;
84.     }
85.     return daySeqOfYear;
86. }
```

```
87.
88. int getDaySeqOfWeek(int daySeqOfYear) {
89.     int daySeqOfWeek = (daySeqOfYear + 6) % 7 + 1;
90.     return daySeqOfWeek;
91. }
92.
93. int getNextMonday(int daySeqOfYear) {
94.     int nextMonday;
95.     int dayOfWeek = getDaySeqOfWeek(daySeqOfYear);
96.     if (dayOfWeek == 1){
97.         nextMonday = daySeqOfYear;
98.     }else{
99.         nextMonday = daySeqOfYear + (8 - dayOfWeek);
100.       }
101.       return nextMonday;
102.   }
103.
104.   int getThisSunday(int daySeqOfYear) {
105.       int thisSunday;
106.       int dayOfWeek = getDaySeqOfWeek(daySeqOfYear);
107.       if (dayOfWeek == 7){
108.           thisSunday = daySeqOfYear;
109.       }else{
110.           thisSunday = daySeqOfYear + (7 - dayOfWeek);
111.       }
112.       return thisSunday;
113.   }
114.
115.   void printOneDay(int daySeqOfYear) {
116.       int month = getMonth(daySeqOfYear);
117.       int day = getDay(daySeqOfYear);
118.       printf("%02d.%02d ", month, day);
119.   }
```

14.5 实验五“春季校历函数版(2024 春)”代码

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  // Functions about month and day
5.  int getMonthLength(int month);
6.  int getDaySeq(int month, int day);
7.
8.  // Functions for properties of one day
9.  int getMonth(int daySeqOfYear);
10. int getDay(int daySeqOfYear);
11. int getDaySeqOfWeek(int daySeqOfYear);
12.
13. // Functions for day movement calculation
14. int getNextMonday(int daySeqOfYear);
15. int getThisSunday(int daySeqOfYear);
16.
17. // Functions support school calendar display
18. void printOneDay(int daySeqOfYear);
19.
20. int main(){
21.     int referenceDay;
22.     printf("请输入校历第一天: ");
23.     scanf("%d",&referenceDay);
24.     if(referenceDay > 0 && referenceDay <= 29){
25.         // 计算春季学期的第一天和最后一天
26.         int sStartSeqOfYear = getNextMonday(getDaySeq(2, referenceDay
    ));
```

```
27.         int sEndSeqOfYear = getThisSunday(getDaySeq(7, 1));
28.
29.         printf("#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.\n");
30.
31.         // 遍历这个学期的每一天
32.         int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;
33.         for (daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0, weekSe
    qOfSemester = 1;
34.              daySeqOfYear <= sEndSeqOfYear;
35.              daySeqOfYear++, daySeqOfWeek++, daySeqOfWeek %= 7){
36.             // 在周一之前，打印周次
37.             if (daySeqOfWeek == 0){
38.                 printf("[%02d]", weekSeqOfSemester);
39.             }
40.
41.             // 打印当前日期
42.             printOneDay(daySeqOfYear);
43.
44.             // 在周日之后，换行并增加周次
45.             if (daySeqOfWeek == 6){
46.                 printf("\n");
47.                 weekSeqOfSemester++;
48.             }
49.         }
50.     }else{
51.         printf("输入错误");
52.     }
53. }
54.
55. int getMonthLength(int month) {
56.     int daysInMonth[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30,
    31};
57.     return daysInMonth[month - 1];
58. }
59.
60. int getDaySeq(int month, int day) {
61.     int totalDays = 0;
62.     int i;
63.     for (i = 1; i < month; ++i) {
64.         totalDays += getMonthLength(i);
65.     }
66.     totalDays += day;
67.     return totalDays;
68. }
69.
70. int getMonth(int daySeqOfYear) {
71.     int month = 1;
72.     while(month <= 12 && daySeqOfYear > getMonthLength(month)){
73.         daySeqOfYear -= getMonthLength(month);
74.         month ++;
75.     }
76.     return month;
77. }
78.
79. int getDay(int daySeqOfYear) {
80.     int month = 1;
81.     while (month <= 12 && daySeqOfYear > getMonthLength(month)){
82.         daySeqOfYear -= getMonthLength(month);
83.         month++;
84.     }
85.     return daySeqOfYear;
86. }
87.
88. int getDaySeqOfWeek(int daySeqOfYear) {
89.     int daySeqOfWeek = (daySeqOfYear + 6) % 7 + 1;
```

```
    return daySeqOfWeek;
}

int getNextMonday(int daySeqOfYear) {
    int nextMonday;
    int dayOfWeek = getDaySeqOfWeek(daySeqOfYear);
    if (dayOfWeek == 1){
        nextMonday = daySeqOfYear;
    }else{
        nextMonday = daySeqOfYear + (8 - dayOfWeek);
      }
      return nextMonday;
  }

  int getThisSunday(int daySeqOfYear) {
      int thisSunday;
      int dayOfWeek = getDaySeqOfWeek(daySeqOfYear);
      if (dayOfWeek == 7){
          thisSunday = daySeqOfYear;
      }else{
          thisSunday = daySeqOfYear + (7 - dayOfWeek);
      }
      return thisSunday;
  }

  void printOneDay(int daySeqOfYear) {
      int month = getMonth(daySeqOfYear);
      int day = getDay(daySeqOfYear);
      printf("%02d.%02d ", month, day);
  }
```

14.6 实验六“秋季校历函数版(2024 秋)”代码

```
#include <stdio.h>
#include <stdlib.h>

// Functions on different year
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 getNextMonday(int year, int daySeqOfYear);
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(){
    int referenceDay;
    printf("请输入校历最后一天：");
    scanf("%d",&referenceDay);
    if(referenceDay > 0 && referenceDay <= 31){
        int currentYearLength, currentYear = 2024;
        if(isLeapYear(currentYear)){
```

```
32.             currentYearLength = 366;
33.         }else{
34.             currentYearLength = 365;
35.         }
36.
37.         // 计算秋季学期的第一天和最后一天
38.         int sStartSeqOfYear = getThisMonday(2024, getDaySeq(2024, 9,
    1));
39.         int sEndSeqOfYear = currentYearLength + getThisSunday(2025, g
    etDaySeq(2025, 1, referenceDay));
40.
41.         printf("#W:  Mon.  Tue.  Wed.  Thu.  Fri.  Sat.  Sun.\n");
42.
43.         // Declare variables to navigate the semester
44.         int daySeqOfYear, daySeqOfWeek, weekSeqOfSemester;
45.
46.         // Print the semester calendar
47.         for (daySeqOfYear = sStartSeqOfYear, daySeqOfWeek = 0, weekSe
    qOfSemester = 1;
48.              daySeqOfYear <= sEndSeqOfYear;
49.              daySeqOfYear++, daySeqOfWeek++, daySeqOfWeek %= 7){
50.             // Before Monday, print the week sequence
51.             if (daySeqOfWeek == 0){
52.                 printf("[%02d]", weekSeqOfSemester);
53.             }
54.
55.             // Print the month and day for the current day
56.             if (daySeqOfYear <= currentYearLength){
57.                 printOneDay(currentYear, daySeqOfYear);
58.             }else{
59.                 printOneDay(currentYear + 1, daySeqOfYear - currentYe
    arLength);
60.             }
61.
62.             // After Sunday, print a new line
63.             if (daySeqOfWeek == 6){
64.                 printf("\n");
65.                 weekSeqOfSemester++;
66.             }
67.         }
68.     }else{
69.         printf("输入错误");
70.     }
71. }
72.
73. int isLeapYear(int year) {
74.     int i;
75.     if((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)){
76.         i = 1;
77.     }else{
78.         i = 0;
79.     }
80.
81.     return i;
82. }
83.
84. int getDaySeqOnJan1(int year) {
85.     int days = 0;
86.     //1990年1月1日是星期一
87.     int i;
88.     for (int i = 1990; i < year; i ++) {
89.         days += isLeapYear(i) ? 366 : 365;
90.     }
91.     return days % 7;
92. }
```

```
93.
94. int getMonthLength(int year, int month) {
95.     int daysInMonth[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30,
    31};
96.     if(isLeapYear(year)){
97.         daysInMonth[1] = 29;
98.     }else{
99.         daysInMonth[1] = 28;
100.       }
101.       return daysInMonth[month - 1];
102.   }
103.
104.   int getDaySeq(int year, int month, int day) {
105.       int totalDays = 0;
106.       int i;
107.       for (i = 1; i < month; i ++) {
108.           totalDays += getMonthLength(year, i);
109.       }
110.       totalDays += day;
111.       return totalDays;
112.   }
113.
114.   int getMonth(int year, int daySeqOfYear) {
115.       int month = 1;
116.       while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
117.           daySeqOfYear -= getMonthLength(year, month);
118.           month ++;
119.       }
120.       return month;
121.   }
122.
123.   int getDay(int year, int daySeqOfYear) {
124.       int month = 1;
125.       while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
126.           daySeqOfYear -= getMonthLength(year, month);
127.           month++;
128.       }
129.       return daySeqOfYear;
130.   }
131.
132.   int getDaySeqOfWeek(int year, int daySeqOfYear) {
133.       int dayOfWeek = getDaySeqOnJan1(year);
134.       int daySeqOfWeek = (dayOfWeek + daySeqOfYear - 1) % 7 + 1;
135.       return daySeqOfWeek;
136.   }
137.
138.   int getNextMonday(int year, int daySeqOfYear) {
139.       int nextMonday;
140.       int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
141.       if (dayOfWeek == 1){
142.           nextMonday = daySeqOfYear;
143.       }else{
144.           nextMonday = daySeqOfYear + (8 - dayOfWeek);
145.       }
146.       return nextMonday;
147.   }
148.
149.   int getThisMonday(int year, int daySeqOfYear) {
150.       int thisMonday;
151.       int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
152.       if (dayOfWeek == 1){
153.           thisMonday = daySeqOfYear;
154.       }else{
155.           thisMonday = daySeqOfYear - dayOfWeek + 1;
```

```
156.        }
157.        return thisMonday;
158.    }
159.
160.    int getThisSunday(int year, int daySeqOfYear) {
161.        int thisSunday;
162.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
163.        if (dayOfWeek == 7){
164.            thisSunday = daySeqOfYear;
165.        }else{
166.            thisSunday = daySeqOfYear + (7 - dayOfWeek);
167.        }
168.        return thisSunday;
169.    }
170.
171.    void printOneDay(int year, int daySeqOfYear) {
172.        int month = getMonth(year, daySeqOfYear);
173.        int day = getDay(year, daySeqOfYear);
174.        printf("%02d.%02d ", month, day);
175.    }
```

14.7 实验七“打印全学年校历(函数版)”代码

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  // Functions on different year
5.  int isLeapYear(int year);
6.  int getDaySeqOnJan1(int year);
7.
8.  // Functions about month and day
9.  int getMonthLength(int year, int month);
10. int getDaySeq(int year, int month, int day);
11. int getWeekSeqOfYear(int year, int month, int day);
12.
13. // Functions for properties of one day
14. int getMonth(int year, int daySeqOfYear);
15. int getDay(int year, int daySeqOfYear);
16. int getDaySeqOfWeek(int year, int daySeqOfYear);
17.
18. // Functions for day movement calculation
19. int getNextMonday(int year, int daySeqOfYear);
20. int getThisMonday(int year, int daySeqOfYear);
21. int getThisSunday(int year, int daySeqOfYear);
22.
23. // Functions support school calendar display
24. void printOneDay(int year, int daySeqOfYear);
25. void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow);
26.
27. int main() {
28.     int inputYear, week;
29.     printf("请输入学期年份：");
30.     scanf("%d",&inputYear);
31.
32.     // Print calendar for fall semester in this year
33.     printf("%13s%s%d%s%d\n", " ", "First Semester (Fall) Calendar of
    Year ", inputYear, " - ", inputYear + 1);
34.     printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues.", "We
    d.", "Thur.", "Fri.", "Sat.", "Sun.");
35.
36.     // Print weeks for the fall semester
37.     for (week = getWeekSeqOfYear(inputYear, 9, 4); week <= getWeekSeq
    OfYear(inputYear, 12, 31) + getWeekSeqOfYear(inputYear + 1, 1, 20); w
    eek ++) {
```

```
38.         printOneWeek(inputYear, week, 1);
39.     }
40.
41.     // Print calendar for spring semester in next year
42.     printf("%13s%s%d%s%d\n", " ", "Second Semester (Spring) Calendar
    of Year ", inputYear, " - ", inputYear + 1);
43.     printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues.", "We
    d.", "Thur.", "Fri.", "Sat.", "Sun.");
44.
45.     for (week = getWeekSeqOfYear(inputYear + 1, 2, 15); week <= getWe
    ekSeqOfYear(inputYear + 1, 7, 1); week ++) {
46.         printOneWeek(inputYear + 1, week, 1);
47.     }
48.
49.     return 0;
50. }
51.
52. int isLeapYear(int year) {
53.     int i;
54.     if((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)){
55.         i = 1;
56.     }else{
57.         i = 0;
58.     }
59.     return i;
60. }
61.
62. int getDaySeqOnJan1(int year) {
63.     int days = 0;
64.     //1990年1月1日是星期一
65.     int i;
66.     for (int i = 1990; i < year; i ++) {
67.         days += isLeapYear(i) ? 366 : 365;
68.     }
69.     return days % 7;
70. }
71.
72. int getMonthLength(int year, int month) {
73.     int daysInMonth[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30,
    31};
74.     if(isLeapYear(year)){
75.         daysInMonth[1] = 29;
76.     }else{
77.         daysInMonth[1] = 28;
78.     }
79.     return daysInMonth[month - 1];
80. }
81.
82. int getDaySeq(int year, int month, int day) {
83.     int totalDays = 0;
84.     int i;
85.     for (i = 1; i < month; i ++) {
86.         totalDays += getMonthLength(year, i);
87.     }
88.     totalDays += day;
89.     return totalDays;
90. }
91.
92. int getWeekSeqOfYear(int year, int month, int day) {
93.     int daySeqOfYear = getDaySeq(year, month, day);
94.     int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
95.     int weekSeqOfYear = (daySeqOfYear - dayOfWeek + 1) / 7 + 1;
96.     return weekSeqOfYear;
97. }
98.
99. int getMonth(int year, int daySeqOfYear) {
```

```
100.        int month = 1;
101.        while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
102.            daySeqOfYear -= getMonthLength(year, month);
103.            month ++;
104.        }
105.        return month;
106.    }
107.
108.    int getDay(int year, int daySeqOfYear) {
109.        int month = 1;
110.        while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
111.            daySeqOfYear -= getMonthLength(year, month);
112.            month++;
113.        }
114.        return daySeqOfYear;
115.    }
116.
117.    int getDaySeqOfWeek(int year, int daySeqOfYear) {
118.        int dayOfWeek = getDaySeqOnJan1(year);
119.        int daySeqOfWeek = (dayOfWeek + daySeqOfYear - 1) % 7 + 1;
120.        return daySeqOfWeek;
121.    }
122.
123.    int getNextMonday(int year, int daySeqOfYear) {
124.        int nextMonday;
125.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
126.        if (dayOfWeek == 1){
127.            nextMonday = daySeqOfYear;
128.        }else{
129.            nextMonday = daySeqOfYear + (8 - dayOfWeek);
130.        }
131.        return nextMonday;
132.    }
133.
134.    int getThisMonday(int year, int daySeqOfYear) {
135.        int thisMonday;
136.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
137.        if (dayOfWeek == 1){
138.            thisMonday = daySeqOfYear;
139.        }else{
140.            thisMonday = daySeqOfYear - dayOfWeek + 1;
141.        }
142.        return thisMonday;
143.    }
144.
145.    int getThisSunday(int year, int daySeqOfYear) {
146.        int thisSunday;
147.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
148.        if (dayOfWeek == 7){
149.            thisSunday = daySeqOfYear;
150.        }else{
151.            thisSunday = daySeqOfYear + (7 - dayOfWeek);
152.        }
153.        return thisSunday;
154.    }
155.
156.    void printOneDay(int year, int daySeqOfYear) {
157.        int month = getMonth(year, daySeqOfYear);
158.        int day = getDay(year, daySeqOfYear);
159.        if(month == 13){
160.            month = 1;
161.        }
162.        printf("    %02d.%02d ", month, day);
163.    }
```

```
164.
165.   void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow) {

166.       int startDaySeqOfYear = (weekSeqOfYear - 1) * 7 + getThisMonda
    y(year, 1);
167.       int endDaySeqOfYear = startDaySeqOfYear + 6;
168.
169.       if (weekSeqShow == 1) {
170.           printf("[%02d] ", weekSeqOfYear);
171.       }
172.
173.       for (int daySeqOfYear = startDaySeqOfYear; daySeqOfYear <= end
    DaySeqOfYear; daySeqOfYear++) {
174.           printOneDay(year, daySeqOfYear);
175.       }
176.       printf("\n");
177.   }
```

14.8 实验八“不同年份秋季校历首周对比”代码

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  // Functions on different year
5.  int isLeapYear(int year);
6.  int getDaySeqOnJan1(int year);
7.
8.  // Functions about month and day
9.  int getMonthLength(int year, int month);
10. int getDaySeq(int year, int month, int day);
11. int getWeekSeqOfYear(int year, int month, int day);
12.
13. // Functions for properties of one day
14. int getMonth(int year, int daySeqOfYear);
15. int getDay(int year, int daySeqOfYear);
16. int getDaySeqOfWeek(int year, int daySeqOfYear);
17.
18. // Functions for day movement calculation
19. int getNextMonday(int year, int daySeqOfYear);
20. int getThisMonday(int year, int daySeqOfYear);
21. int getThisSunday(int year, int daySeqOfYear);
22.
23. // Functions support school calendar display
24. void printOneDay(int year, int daySeqOfYear);
25. void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow);
26.
27. // Functions support multiple year
28. void setYearArray(int yearArray[], int yearNum, int yearStart);
29.
30. #define YEAR_NUM 5
31. #define YEAR_MIN 2000
32. #define YEAR_MAX 2040
33.
34. int main(){
35.     // Declare variables
36.     int inputYear, Years[YEAR_NUM];
37.
38.     // Display the program information
39.     printf("Fall Calendar Comparison in %d years, please input the fi
    rst year (%d-%d): \n",YEAR_NUM, YEAR_MIN, YEAR_MAX - YEAR_NUM);
40.     scanf("%d", &inputYear);
41.
42.     if (inputYear < YEAR_MIN || inputYear + YEAR_NUM > YEAR_MAX) {
43.         printf("Sorry, the input year is not supported.\n");
```

```
44.         return 1;
45.     }
46.
47.     setYearArray(Years, YEAR_NUM, inputYear);
48.
49.     int i;
50.     int startDays[YEAR_NUM] = {0};
51.     for (i = 0; i < YEAR_NUM; i++) {
52.         printf("\n%s%s%d\n", " ", "First week in fall Calendar of Yea
    r ", Years[i]);
53.         printf(" #W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues.",
     "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
54.         printOneWeek(Years[i], getWeekSeqOfYear(Years[i], 9, 4), 1);
55.         startDays[i] = getThisMonday(Years[i], getDaySeq(Years[i], 9,
     4));
56.     }
57.
58.     int min = 0;
59.     for (i = 1; i < YEAR_NUM; i++) {
60.         min = (startDays[i] < startDays[min]) ? i : min;
61.     }
62.
63.     printf("\n%s%s%d\n", " ", "Earliest Fall Semester is in Year ", Y
    ears[min]);
64. }
65.
66. int isLeapYear(int year) {
67.     int i;
68.     if((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)){
69.         i = 1;
70.     }else{
71.         i = 0;
72.     }
73.     return i;
74. }
75.
76. int getDaySeqOnJan1(int year) {
77.     int days = 0;
78.     //1990年1月1日是星期一
79.     int i;
80.     for (int i = 1990; i < year; i ++) {
81.         days += isLeapYear(i) ? 366 : 365;
82.     }
83.     return days % 7;
84. }
85.
86. int getMonthLength(int year, int month) {
87.     int daysInMonth[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30,
    31};
88.     if(isLeapYear(year)){
89.         daysInMonth[1] = 29;
90.     }else{
91.         daysInMonth[1] = 28;
92.     }
93.     return daysInMonth[month - 1];
94. }
95.
96. int getDaySeq(int year, int month, int day) {
97.     int totalDays = 0;
98.     int i;
99.     for (i = 1; i < month; i ++) {
100.         totalDays += getMonthLength(year, i);
101.     }
102.     totalDays += day;
103.     return totalDays;
```

```
104.    }
105.
106.    int getWeekSeqOfYear(int year, int month, int day) {
107.        int daySeqOfYear = getDaySeq(year, month, day);
108.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
109.        int weekSeqOfYear = (daySeqOfYear - dayOfWeek + 1) / 7 + 1;
110.        return weekSeqOfYear;
111.    }
112.
113.    int getMonth(int year, int daySeqOfYear) {
114.        int month = 1;
115.        while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
116.            daySeqOfYear -= getMonthLength(year, month);
117.            month ++;
118.        }
119.        return month;
120.    }
121.
122.    int getDay(int year, int daySeqOfYear) {
123.        int month = 1;
124.        while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
125.            daySeqOfYear -= getMonthLength(year, month);
126.            month++;
127.        }
128.        return daySeqOfYear;
129.    }
130.
131.    int getDaySeqOfWeek(int year, int daySeqOfYear) {
132.        int dayOfWeek = getDaySeqOnJan1(year);
133.        int daySeqOfWeek = (dayOfWeek + daySeqOfYear - 1) % 7 + 1;
134.        return daySeqOfWeek;
135.    }
136.
137.    int getNextMonday(int year, int daySeqOfYear) {
138.        int nextMonday;
139.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
140.        if (dayOfWeek == 1){
141.            nextMonday = daySeqOfYear;
142.        }else{
143.            nextMonday = daySeqOfYear + (8 - dayOfWeek);
144.        }
145.        return nextMonday;
146.    }
147.
148.    int getThisMonday(int year, int daySeqOfYear) {
149.        int thisMonday;
150.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
151.        if (dayOfWeek == 1){
152.            thisMonday = daySeqOfYear;
153.        }else{
154.            thisMonday = daySeqOfYear - dayOfWeek + 1;
155.        }
156.        return thisMonday;
157.    }
158.
159.    int getThisSunday(int year, int daySeqOfYear) {
160.        int thisSunday;
161.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
162.        if (dayOfWeek == 7){
163.            thisSunday = daySeqOfYear;
164.        }else{
165.            thisSunday = daySeqOfYear + (7 - dayOfWeek);
166.        }
167.        return thisSunday;
```

```
168.   }
169.
170.   void printOneDay(int year, int daySeqOfYear) {
171.       int month = getMonth(year, daySeqOfYear);
172.       int day = getDay(year, daySeqOfYear);
173.       if(month == 13){
174.           month = 1;
175.       }
176.       printf("    %02d.%02d ", month, day);
177.   }
178.
179.   void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow) {

180.       int startDaySeqOfYear = (weekSeqOfYear - 1) * 7 + getThisMonda
    y(year, 1);
181.       int endDaySeqOfYear = startDaySeqOfYear + 6;
182.
183.       if (weekSeqShow == 1) {
184.           printf("[%02d] ", weekSeqOfYear);
185.       }
186.
187.       for (int daySeqOfYear = startDaySeqOfYear; daySeqOfYear <= end
    DaySeqOfYear; daySeqOfYear++) {
188.           printOneDay(year, daySeqOfYear);
189.       }
190.       printf("\n");
191.   }
192.
193.   void setYearArray(int yearArray[], int yearNum, int yearStart) {
194.       int i;
195.       for (i = 0; i < yearNum; i++) {
196.           yearArray[i] = yearStart + i;
197.       }
198.   }
```

14.9 实验九“寻找生日(多维数组)”代码

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  // Functions on different year
5.  int isLeapYear(int year);
6.  int getDaySeqOnJan1(int year);
7.
8.  // Functions about month and day
9.  int getMonthLength(int year, int month);
10. int getDaySeq(int year, int month, int day);
11. int getWeekSeqOfYear(int year, int month, int day);
12.
13. // Functions for properties of one day
14. int getMonth(int year, int daySeqOfYear);
15. int getDay(int year, int daySeqOfYear);
16. int getDaySeqOfWeek(int year, int daySeqOfYear);
17.
18. // Functions for day movement calculation
19. int getNextMonday(int year, int daySeqOfYear);
20. int getThisMonday(int year, int daySeqOfYear);
21. int getThisSunday(int year, int daySeqOfYear);
22.
23. // Functions support school calendar display
24. void printOneDay(int year, int daySeqOfYear);
25. void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow);
26.
27. // Functions support multiple year
```

```c
void setYearArray(int yearArray[], int yearNum, int yearStart);
void initialDays(int Years[], int Days[][366][4], int yearNum);

#define YEAR_NUM 6
#define YEAR_MIN 2020
#define YEAR_MAX 2025

int main() {
    int inputMonth, inputDay, Years[YEAR_NUM];

    int Days[YEAR_NUM][366][4] = {0};

    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 > getMonthLength(YEAR_MIN, inputMonth)) {
        printf("Sorry, the input month and day are invalid.\n");
        return 1;
    }

    setYearArray(Years, YEAR_NUM, YEAR_MIN);
    initialDays(Years, Days, YEAR_NUM);

    int totalNum = 0, weekendNum = 0;

    int i;
    for (i = 0; i < YEAR_NUM; i++) {
        int year = Years[i];
        int daySeqOfYear = getDaySeq(year, inputMonth, inputDay);
        if (daySeqOfYear <= (isLeapYear(year) ? 366 : 365)) {
            printf("\nBirthday in Year %d\n", year);
            if(!(isLeapYear(year)) && inputMonth == 2 && inputDay == 29){
                printf("Not found.\n");
            }else{
                int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);

                if (dayOfWeek == 6 || dayOfWeek == 7){
                    weekendNum++;
                }
                totalNum ++;
                printf("#M:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues.", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
                printOneWeek(year, getWeekSeqOfYear(year, inputMonth, inputDay), 1);
            }
        }
    }

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

    return 0;
}

int isLeapYear(int year) {
    int i;
    if((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)){
        i = 1;
    }else{
        i = 0;
    }
    return i;
}
```

```
87.
88. int getDaySeqOnJan1(int year) {
89.     int days = 0;
90.     //1990年1月1日是星期一
91.     int i;
92.     for (int i = 1990; i < year; i ++) {
93.         days += isLeapYear(i) ? 366 : 365;
94.     }
95.     return days % 7;
96. }
97.
98. int getMonthLength(int year, int month) {
99.     int daysInMonth[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30,
    31};
100.        if(isLeapYear(year)){
101.            daysInMonth[1] = 29;
102.        }else{
103.            daysInMonth[1] = 28;
104.        }
105.        return daysInMonth[month - 1];
106.    }
107.
108.    int getDaySeq(int year, int month, int day) {
109.        int totalDays = 0;
110.        int i;
111.        for (i = 1; i < month; i ++) {
112.            totalDays += getMonthLength(year, i);
113.        }
114.        totalDays += day;
115.        return totalDays;
116.    }
117.
118.    int getWeekSeqOfYear(int year, int month, int day) {
119.        int daySeqOfYear = getDaySeq(year, month, day);
120.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
121.        int weekSeqOfYear = (daySeqOfYear - dayOfWeek + 1) / 7 + 1;
122.        return weekSeqOfYear;
123.    }
124.
125.    int getMonth(int year, int daySeqOfYear) {
126.        int month = 1;
127.        while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
128.            daySeqOfYear -= getMonthLength(year, month);
129.            month ++;
130.        }
131.        return month;
132.    }
133.
134.    int getDay(int year, int daySeqOfYear) {
135.        int month = 1;
136.        while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
137.            daySeqOfYear -= getMonthLength(year, month);
138.            month++;
139.        }
140.        return daySeqOfYear;
141.    }
142.
143.    int getDaySeqOfWeek(int year, int daySeqOfYear) {
144.        int dayOfWeek = getDaySeqOnJan1(year);
145.        int daySeqOfWeek = (dayOfWeek + daySeqOfYear - 1) % 7 + 1;
146.        return daySeqOfWeek;
147.    }
148.
149.    int getNextMonday(int year, int daySeqOfYear) {
```

```
150.        int nextMonday;
151.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
152.        if (dayOfWeek == 1){
153.            nextMonday = daySeqOfYear;
154.        }else{
155.            nextMonday = daySeqOfYear + (8 - dayOfWeek);
156.        }
157.        return nextMonday;
158.    }
159.
160.    int getThisMonday(int year, int daySeqOfYear) {
161.        int thisMonday;
162.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
163.        if (dayOfWeek == 1){
164.            thisMonday = daySeqOfYear;
165.        }else{
166.            thisMonday = daySeqOfYear - dayOfWeek + 1;
167.        }
168.        return thisMonday;
169.    }
170.
171.    int getThisSunday(int year, int daySeqOfYear) {
172.        int thisSunday;
173.        int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
174.        if (dayOfWeek == 7){
175.            thisSunday = daySeqOfYear;
176.        }else{
177.            thisSunday = daySeqOfYear + (7 - dayOfWeek);
178.        }
179.        return thisSunday;
180.    }
181.
182.    void printOneDay(int year, int daySeqOfYear) {
183.        int month = getMonth(year, daySeqOfYear);
184.        int day = getDay(year, daySeqOfYear);
185.        if(month == 13){
186.            month = 1;
187.        }
188.        printf("    %02d.%02d ", month, day);
189.    }
190.
191.    void printOneWeek(int year, int weekSeqOfYear, int weekSeqShow) {

192.        int startDaySeqOfYear;
193.        if(getThisMonday(year, 1) == 1){
194.            startDaySeqOfYear = (weekSeqOfYear - 1) * 7 + getThisMonda
    y(year, 1);
195.        }else{
196.            startDaySeqOfYear = (weekSeqOfYear - 1) * 7 + getThisMonda
    y(year, 1) + 7;
197.        }
198.        int endDaySeqOfYear = startDaySeqOfYear + 6;
199.
200.        if (weekSeqShow == 1) {
201.            printf("[%02d] ", weekSeqOfYear);
202.        }
203.
204.        for (int daySeqOfYear = startDaySeqOfYear; daySeqOfYear <= end
    DaySeqOfYear; daySeqOfYear++) {
205.            printOneDay(year, daySeqOfYear);
206.        }
207.        printf("\n");
208.    }
209.
210.    void setYearArray(int yearArray[], int yearNum, int yearStart) {
211.        int i;
```

```
212.       for (i = 0; i < yearNum; i++) {
213.           yearArray[i] = yearStart + i;
214.       }
215.   }
216.
217.   void initialDays(int Years[], int Days[][366][4], int yearNum) {
218.       int year, month, day, yearLength, weekSeq, seqOfWeek;
219.       int i, j;
220.
221.       for (i = 0; i < YEAR_NUM; i++) {
222.           year = Years[i];
223.           yearLength = isLeapYear(year) ? 366 : 365;
224.
225.           for (j = 0; j < yearLength; j++) {
226.               Days[i][j][0] = getMonth(year, j + 1);
227.               Days[i][j][1] = getDay(year, j + 1);
228.               Days[i][j][2] = getWeekSeqOfYear(year, Days[i][j][0],
    Days[i][j][1]);
229.               Days[i][j][3] = getDaySeqOfWeek(year, j + 1);
230.           }
231.       }
232.   }
```

14.10 实验十“寻找生日(结构体)”代码

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  typedef struct {
5.      int year;         // Year
6.      int daySeq;       // Day sequence of year
7.      int month;        // Month
8.      int day;          // Day
9.      int weekSeq;      // Week sequence of year
10.     int weekDay;      // Day sequence of week
11. } Day;
12.
13. #define YEAR_NUM 3
14. #define YEAR_MIN 2023
15. #define YEAR_MAX 2025
16. #define DATE_NOSHOW 0
17. #define DATE_INFO_FULL 1
18. #define DATE_INFO_BRIEF 2
19. #define DATE_STAR 3
20.
21. // Functions on different year
22. int isLeapYear(int year);
23. int getDaySeqOnJan1(int year);
24.
25. // Functions about month and day
26. int getMonthLength(int year, int month);
27. int getDaySeq(int year, int month, int day);
28. int getWeekSeqOfYear(int year, int month, int day);
29.
30. // Functions for properties of one day
31. int getMonth(int year, int daySeqOfYear);
32. int getDay(int year, int daySeqOfYear);
33. int getDaySeqOfWeek(int year, int daySeqOfYear);
34.
35. // Functions support multiple year
36. void setYearArray(int yearArray[], int yearNum, int yearStart);
37.
38. // Functions support struct Day
39. int isDay(int year, int month, int day);
```

```
40. Day setDay(int year, int month, int day);
41. Day getDayBefore(Day currentDay, int interval);
42. Day getDayAfter(Day currentDay, int interval);
43. int getTwoDaysInterval(Day startDay, Day endDay);
44. void printDay(Day currentDay, int displayFormat);
45. 
46. int main() {
47.     int Years[YEAR_NUM] = {YEAR_MIN, YEAR_MIN + 1, YEAR_MIN + 2};
48.     int inputMonth, inputDay;
49.     printf("Finding Birthday in year (%d-%d), please input the month
    and day: \n", YEAR_MIN, YEAR_MAX);
50.     scanf("%d %d", &inputMonth, &inputDay);
51. 
52.     if(inputMonth >= 1 && inputMonth <= 12 && inputDay >= 1 && inputD
    ay <= getDay(YEAR_MIN + 1, getDaySeq(YEAR_MIN + 1, inputMonth, inputD
    ay))){
53.         int i;
54.         for (i = 0; i < YEAR_NUM; i ++) {
55.             printf("\n%s%s%d\n", " ", "Birthday in Year ", Years[i]);

56.             if (!(isDay(Years[i], inputMonth, inputDay))) {
57.                 printf("Not found.\n");
58.                 continue;
59.             }
60.             Day birthDay = setDay(Years[i], inputMonth, inputDay);
61.             Day prepareDay = getDayBefore(birthDay, 3);
62.             Day printStartDay = getDayBefore(prepareDay, prepareDay.w
    eekDay);
63.             printf("#W:%10s%10s%10s%10s%10s%10s%10s\n", "Mon.", "Tues
    .", "Wed.", "Thur.", "Fri.", "Sat.", "Sun.");
64. 
65.             for (Day currentDay = printStartDay; getTwoDaysInterval(c
    urrentDay, birthDay) <= 0; currentDay = getDayAfter(currentDay, 1)) {

66.                 int displayFormat;
67.                 if (getTwoDaysInterval(currentDay, prepareDay) < 0) {

68.                     displayFormat = DATE_NOSHOW;
69.                 } else if (getTwoDaysInterval(currentDay, prepareDay)
     == 0) {
70.                     displayFormat = DATE_INFO_FULL;
71.                 } else if (getTwoDaysInterval(currentDay, birthDay) <
     0) {
72.                     displayFormat = DATE_INFO_BRIEF;
73.                 } else if (getTwoDaysInterval(currentDay, birthDay) =
    = 0) {
74.                     displayFormat = DATE_STAR;
75.                 }
76.                 printDay(currentDay, displayFormat);
77.                 if (currentDay.weekDay == 7) {
78.                     printf("\n");
79.                     if (getTwoDaysInterval(currentDay, birthDay) >= 0
    ) {
80.                         break;
81.                     }
82.                     printf("[%02d]", getWeekSeqOfYear(currentDay.year
    , currentDay.month, currentDay.day));
83.                 }
84.             }
85.             printf("\n");
86.         }
87.     }else{
88.         printf("Input wrong.");
89.     }
90. 
91.     return 0;
```

```
92. }
93.
94. int isLeapYear(int year) {
95.     int i;
96.     if((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)){
97.         i = 1;
98.     }else{
99.         i = 0;
100.       }
101.       return i;
102.   }
103.
104.   int getDaySeqOnJan1(int year) {
105.       int days = 0;
106.       //1990年1月1日是星期一
107.       int i;
108.       for (int i = 1990; i < year; i ++) {
109.           days += isLeapYear(i) ? 366 : 365;
110.       }
111.       return days % 7;
112.   }
113.
114.   int getMonthLength(int year, int month) {
115.       int daysInMonth[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 3
    0, 31};
116.       if(isLeapYear(year)){
117.           daysInMonth[1] = 29;
118.       }else{
119.           daysInMonth[1] = 28;
120.       }
121.       return daysInMonth[month - 1];
122.   }
123.
124.   int getDaySeq(int year, int month, int day) {
125.       int totalDays = 0;
126.       int i;
127.       for (i = 1; i < month; i ++) {
128.           totalDays += getMonthLength(year, i);
129.       }
130.       totalDays += day;
131.       return totalDays;
132.   }
133.
134.   int getWeekSeqOfYear(int year, int month, int day) {
135.       int daySeqOfYear = getDaySeq(year, month, day);
136.       int dayOfWeek = getDaySeqOfWeek(year, daySeqOfYear);
137.       int weekSeqOfYear = (daySeqOfYear - dayOfWeek + 1) / 7 + 1;
138.       return weekSeqOfYear;
139.   }
140.
141.   int getMonth(int year, int daySeqOfYear) {
142.       int month = 1;
143.       while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
144.           daySeqOfYear -= getMonthLength(year, month);
145.           month ++;
146.       }
147.       return month;
148.   }
149.
150.   int getDay(int year, int daySeqOfYear) {
151.       int month = 1;
152.       while(month <= 12 && daySeqOfYear > getMonthLength(year, month
    )){
153.           daySeqOfYear -= getMonthLength(year, month);
154.           month++;
```

```
155.        }
156.        return daySeqOfYear;
157.    }
158.
159.    int getDaySeqOfWeek(int year, int daySeqOfYear) {
160.        int dayOfWeek = getDaySeqOnJan1(year);
161.        int daySeqOfWeek = (dayOfWeek + daySeqOfYear - 1) % 7 + 1;
162.        return daySeqOfWeek;
163.    }
164.
165.    void setYearArray(int yearArray[], int yearNum, int yearStart) {
166.        int i;
167.        for (i = 0; i < yearNum; i++) {
168.            yearArray[i] = yearStart + i;
169.        }
170.    }
171.
172.    int isDay(int year, int month, int day) {
173.        int i;
174.        if(day >= 1 && day <= getMonthLength(year, month)){
175.            i = 1;
176.        }else{
177.            i = 0;
178.        }
179.        return i;
180.    }
181.
182.    Day setDay(int year, int month, int day) {
183.        Day dayStruct;
184.        dayStruct.year = year;
185.        dayStruct.month = month;
186.        dayStruct.day = day;
187.        dayStruct.daySeq = getDaySeq(year, month, day);
188.        dayStruct.weekDay = getDaySeqOfWeek(year, dayStruct.daySeq);
189.        dayStruct.weekSeq = getWeekSeqOfYear(year, month, day);
190.        return dayStruct;
191.    }
192.
193.    Day getDayBefore(Day currentDay, int interval) {
194.        Day prevDay = currentDay;
195.        int i = isLeapYear(prevDay.year)? 366:365;
196.        prevDay.daySeq -= interval;
197.        if(prevDay.daySeq <= 0){
198.            prevDay.year --;
199.            prevDay.daySeq += i;
200.        }
201.        prevDay.day = getDay(prevDay.year, prevDay.daySeq);
202.        prevDay.month = getMonth(prevDay.year, prevDay.daySeq);
203.        prevDay.weekDay = (currentDay.weekDay - interval + 6) % 7 + 1;

204.        return prevDay;
205.    }
206.
207.    Day getDayAfter(Day currentDay, int interval) {
208.        Day nextDay = currentDay;
209.        int i = isLeapYear(nextDay.year)? 366:365;
210.        nextDay.daySeq += interval;
211.        if(nextDay.daySeq > i){
212.            nextDay.year ++;
213.            nextDay.daySeq -= i;
214.        }
215.        nextDay.day = getDay(nextDay.year, nextDay.daySeq);
216.        nextDay.month = getMonth(nextDay.year, nextDay.daySeq);
217.        nextDay.weekDay = (currentDay.weekDay + interval - 1) % 7 + 1;

218.        return nextDay;
```

```
219.   }
220.
221.   int getTwoDaysInterval(Day startDay, Day endDay) {
222.       return startDay.daySeq - endDay.daySeq;
223.   }
224.
225.   void printDay(Day currentDay, int displayFormat) {
226.       switch (displayFormat) {
227.           case DATE_NOSHOW:
228.               printf("          ");
229.               break;
230.           case DATE_INFO_FULL:
231.               printf("%02d.%02d.%02d", currentDay.year, currentDay.m
    onth, currentDay.day);
232.               break;
233.           case DATE_INFO_BRIEF:
234.               printf("%9d", currentDay.day);
235.               break;
236.           case DATE_STAR:
237.               printf("      %02d.%02d*", currentDay.month, currentDa
    y.day);
238.               break;
239.       }
240.   }
```