日期处理

1、日期加法：

题目描述

给定一个日期 DAY 和一个正整数 n，求日期 DAY 加上 n 天后的日期。

输入描述

第一行为给定的日期DAY （格式为YYYY-MM-DD，范围为1900-01-01≤DAY ≤2199-12-31），数据保证一定合法；

第二行为需要增加的天数n（1≤n≤10000）。

输出描述

以YYYY-MM-DD的格式输出增加了n天后的日期。

核心代码

```
void DayAdd(int year, int month, int day, int n){
    int sum = 0, sum_days=0;
    int month_days[13]={0,31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
     //365天
    int sum_day[13]={0,31,59,90,120,151,181,212,243,273,304,334, 365};
     //非闰年
    bool flag = false;
    sum_days = sum_day[12];

    if(year%400==0 || (year%4==0 && year%100!=0)){
     //闰年修改2月天数
        flag = true;
    }
    for(int i=0;i<month;i++){
        if(flag) sum = sum_day[i]++;
        else sum = sum_day[i];
    }
    sum = sum + day + n;

    if(flag){
        sum_days++;
        while(sum > sum_days){
            sum = sum - sum_days;
            if(flag){
```

```
                sum_days--;                                         // 恢复成正常年份的总天数
                flag = false;
            }
            year++;
            if(year%400==0 || (year%4==0 && year%100!=0)){      // 闰年
                sum_days++;
                flag = true;
            }
        }
    }
    else{
        while(sum > sum_days){
            sum = sum - sum_days;
            if(flag){
                sum_days--;                                         // 恢复成正常年份的总天数
                flag = false;
            }
            year++;
            if(year%400==0 || (year%4==0 && year%100!=0)){      // 闰年
                sum_days++;
                flag = true;
            }
        }
    }
    for(int i=1;i<13;i++){
        if(year%400==0 || (year%4==0 && year%100!=0)){      // 闰年
           if(i>=2) sum_day[i]++;
        }
        if(sum <= sum_day[i]){
            month = i;
            day = sum - sum_day[i-1];
            break;
        }
    }
    printf("%04d-%02d-%02d", year, month, day);
}
```

代码优化

```
// 每个月的天数
int dayOfMonth[2][13] = {
    {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
    {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
};
// 是否是闰年
bool isLeapYear(int year) {
    return year % 400 == 0 || (year % 4 == 0 && year % 100 != 0);
}
// 给当前日期加 1天（注意参数都用了引用&，这样对参数的修改可以同步到函数外）
```

```
void addOneDay(int &year, int &month, int &day) {
    int run = 0;
    day++;                                  // 让day加1
    if(isLeapYear(year)) run = 1;           // 判断是否是闰年
    if (day > dayOfMonth[run][month]) {     // 如果超过当前月的天数
        month++;                            // 让month加1
        day = 1;                            // 重置day为1号
    }
    if (month > 12) {                       // 如果月份大于12
        year++;                             // 让year加1
        month = 1;                          // 重置month为1月
    }
}

int main() {
    int year, month, day, n;
    scanf("%d-%d-%d", &year, &month, &day);       // 按格式输入年月日
    scanf("%d", &n);                              // 输入需要增加的天数
    for(int i = 0; i < n; i++) {                  // 遍历 n次，每次加 1天
        addOneDay(year, month, day);
    }
    printf("%04d-%02d-%02d", year, month, day);   // 按格式输出年月日
    return 0;
}
```

2、周几

题目描述

给定一个日期 DAY，求它是周几。

输入描述

第一行为给定的日期DAY（格式为YYYY-MM-DD，范围为1900-01-01≤DAY≤2199-12-31），数据保证一定合法。

输出描述

输出一个整数，表示周几。其中周一到周六分别用1-6表示，周天用0表示。

核心代码

```
// 每个月的天数
int dayOfMonth[2][13] = {
    {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
    {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
};
// 是否是闰年
bool isLeapYear(int year) {
    return year % 400 == 0 || (year % 4 == 0 && year % 100 != 0);
}
```

```
//
void DateTOWeek(int year, int month, int day){
    int sumday=0,weekday=1;
    int run=isLeapYear(year);
    for(int i=1900;i<year;i++){
        if(isLeapYear(i)) sumday += 366;
        else sumday += 365;
    }
    for(int i=1;i<month;i++){
        sumday += dayOfMonth[run][i];
    }
    sumday += day;
    //printf("%d\n", sumday);
    weekday = sumday%7;
    printf("%d", weekday);
}
```