

C++方向编程题答案

第六周

day31

1、题目ID: 861 美国节日

链接: <https://www.nowcoder.com/questionTerminal/d95d98a2f96e49078cd7df84ba0c9d79>

【题目解析】:

题目表述很明白, 难点在于我们要求一个月第N个星期W。那么面对这个问题, 我们拆解的思路是, 首先, 我们要想找到一个月第N个星期W, 一定需要一个参照物, 最好的目标当然是这个月的第一天。拿到参照物后, 我要能得到参照物的星期数, 然后就能得到结果了。所以这个题有两个难点: 判断某个月的1号到底是周几, 然后根据这个星期数得到这个月第N个星期W。这两个功能写成函数, 即可通过反复调用拿到结果。

【解题思路】:

第一个难点是如何判断一天到底是周几。那么为了判断一天到底是周几, 我们有以下两种手段:

- 1、找个参照日, 写一个日期计算器, 算出两个天数之间的差值后对7取余数即可知道目标日期是星期几。
- 2、通过蔡勒公式计算星期数:

$$w = ([\frac{c}{4}] - 2c + y + [\frac{y}{4}] + [\frac{13 \times (m + 1)}{5}] + d - 1) \text{MOD} 7$$

公式中w就是计算出的星期; c是世纪, 其值为真实的世纪数-1, 也就是y / 100的结果; y是这是这个世纪的第几年, 也就是y % 100; m是月, 但是1月和2月要看做13月和14月计算; d是日。根据泰勒公式, 我们只需要知道年月日, 就可以算出星期。然而因为现行的格里历, 所以在1582年之前的日期, 我们要采用另一个公式, 本题中用不到, 可以参考百度百科: <https://baike.baidu.com/item/%E8%94%A1%E5%8B%92%E5%85%AC%E5%BC%8F>

两个方法比较而言, 肯定是方法2的代码更好写, 所以我们就用泰勒公式就好。

第一个难点解决了, 第二个难点, 如何通过这个参照点拿到一个月的第N个星期W。

那么, 我们假设要拿到一个月的第一个周五, 我们要怎么做呢? 一个很简单的思路就是, 先看看这个月的1号是周几, 然后往后数就行了, 假如1号是周四, 那么2号就是第一个周五, 假如1号是周六, 那么7号就是第一个周五。那么怎么拿到这个向后的天数呢?

我们发现, 如果所求星期数比1号星期数大, 那么直接相减后+1就是那一天了, 例如1号周三, 我要周五, 那么(5-3)+1即可求出第一个周五是3号。那么反过来是所求星期数小, 例如1号周三, 我要周一, 那么显然要先把周一看成周八才行。也就是(8-3)+1。第一个周一是6号。但是这样要判断, 所以干脆统统都让它加7以后减, 减完后的结果再mod一下7, 就能得到结果了。也就是: (所求星期数 + 7 - 1号星期数) % 7 + 1。这样我们就拿到了求第一个周几公式。随后, 我们只需要在这个公式上, 加上7 * (n - 1), 即刻求出第n个周几。

而面对某个月的最后一个周几, 我们要做的是拿到下个月的第一天然后往回推即可。公式跟上面的差不多, 有了上面的公式, 可以尝试自己来做做看。

【示例代码】:

```
#include <iostream>
```

```

#include <stdio>

// 根据 年-月-日 通过蔡勒公式计算当前星期几
// 1: 星期一 ... 7: 星期日
int day_of_week(int year, int month, int day)
{
    if (month == 1 || month == 2)
    {
        month += 12;
        year -= 1;
    }

    int century = year / 100;
    year %= 100;
    int week = year + (year / 4) + (century / 4) - 2 * century + 26 * (month + 1) / 10 + day - 1;
    week = (week % 7 + 7) % 7;

    if (week == 0)
    {
        week = 7;
    }

    return week;
}

int day_of_demand(int year, int month, int count, int d_of_week)
{
    int week = day_of_week(year, month, 1); // 求出1号星期数
    // 1 + 7(n - 1) + (所求星期数 + 7 - 1号星期数) % 7
    int day = 1 + (count - 1) * 7 + (7 + d_of_week - week) % 7;
    return day;
}

// 元旦
void new_year_day(int year)
{
    printf("%d-01-01\n", year);
}

// 马丁·路德·金纪念日 (1月的第三个星期一)
void martin_luther_king_day(int year)
{
    printf("%d-01-%02d\n", year, day_of_demand(year, 1, 3, 1));
}

// 总统日 (2月的第三个星期一)
void president_day(int year)
{
    printf("%d-02-%02d\n", year, day_of_demand(year, 2, 3, 1));
}

// 阵亡将士纪念日 (5月的最后一个星期一)

```

```

void memorial_day(int year)
{
    // 从 6 月往前数
    int week = day_of_week(year, 6, 1);
    // 星期一的话, 从 31 号往前数 6 天, 否则, 数 week - 2 天
    int day = 31 - ((week == 1) ? 6 : (week - 2));
    printf("%d-05-%02d\n", year, day);
}

// 国庆
void independence_day(int year)
{
    printf("%d-07-04\n", year);
}

// 劳动节 (9月的第一个星期一)
void labor_day(int year)
{
    printf("%d-09-%02d\n", year, day_of_demand(year, 9, 1, 1));
}

// 感恩节 (11月的第四个星期四)
void thanks_giving_day(int year)
{
    printf("%d-11-%02d\n", year, day_of_demand(year, 11, 4, 4));
}

// 圣诞节
void christmas(int year)
{
    printf("%d-12-25\n", year);
}

// 美国节日
void holiday_of_usa(int year)
{
    new_year_day(year);
    martin_luther_king_day(year);
    president_day(year);
    memorial_day(year);
    independence_day(year);
    labor_day(year);
    thanks_giving_day(year);
    christmas(year);
}

int main()
{
    int year;
    while (std::cin >> year)
    {
        holiday_of_usa(year);
        putchar('\n');
    }
}

```

```
}  
}
```

2、题目ID: 759 分解因数

链接: <https://www.nowcoder.com/questionTerminal/0f6976af36324f8bab1ea61e9e826ef5>

【题目解析】:

注: 题目中没有说明的是: 如果出现了质数, 需要打印成 $13 = 13$ 的形式。

小学的知识, 分解因数。想想我们小学的时候怎么做的?

$$\begin{array}{r} 2 \overline{) 90} \\ 3 \overline{) 45} \\ 3 \overline{) 15} \\ 5 \end{array}$$

所以 $90 = 2 * 3 * 3 * 5$ 。

【解题思路】:

看到短除法后, 我们很清楚的知道, 要想求出它的每一个质因数, 我们需要用质数去试除。90能被2整除, 那就拿商继续除以2, 除不尽就换3, 一直到除到质数为止。基础代码框架类似判断质数, 只是被判断的数字在过程中不断被除, 最终循环结束的时候, 那个被处理过的数字, 就是最后一个质因数。以下代码注释以90为例。

【示例代码】:

```
#include <stdio>
#include <cmath>
#include <iostream>

int main()
{
    unsigned int n;
    while (std::cin >> n)
    {
        printf("%d =", n); //输出90 =

        for (unsigned i = 2; i <= std::sqrt(n); i++)
        {
            while (n % i == 0 && n != i) //反复除同一个数, 直到除不尽, 排除刚好是该数的n次方的情况
            {
                printf(" %u *", i); //第一次打印 2 *,第二次打印两个 3 *
                n /= i; //能整除就修改n的值
            }
        }
    }
}
```

```
    printf(" %d\n",n); //跳出后, n已经是处理过的一个质数, 就是最后一个质因数: 5
}

return 0;
}
```

比特科技整理