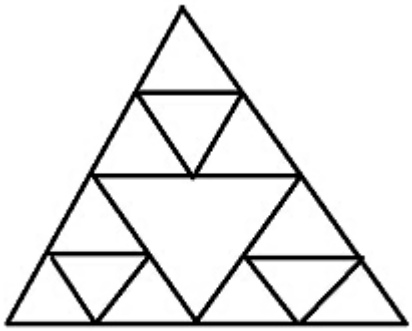
2014 级 C++第六次上机解题报告

——**14211059** 左宗源

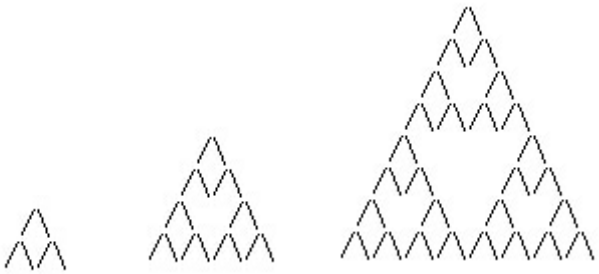
F - jhljx 学画画

Problem Description

jhljx 是一个爱好广泛的人。最近他迷恋上了画画。还记得上次他教松辰学妹学画画。这次他决定自己动手画画啦。。
真是好棒耶。。~\(≧▽≦)/~啦啦啦
他要画的图形是这样的。。



但是为了简化，我们将图形变成了这样。



Input

输入多组数据。
每组数据输入一个 n，表示图形的行数。(保证 n 为 2,4,8,16……，且 n<=10000)

Output

输出正确的图形。

Sample Input

8

Sample Output

Analysis1

这道题乍一看好像并不太难，只需要用一个递归输出三个较小的三角形就可以画出一个大三角形了。但是仔细研究之后可以发现，由于该题存在两个三角形处于同一个水平面（出题的好阴险=_=.....），会出现光标移动的问题，只用递归的话会非常不容易输出（起码咱是没用单纯的递归做出来.....表示咱是弱渣 T--T.....）。因此常规的思路是需要开一个二维 bool 数组储存是否打印三角形来解决这个问题。但是开到 8192*8192 会爆内存（再次鄙视阴险的出题人=_=），因此要少开一个三角形，即 4196*4196。

Code1

```
#include <stdio.h>
#define _MAX 4096
#define _DMAX 8192
using namespace std;

char _map[4100][4100]={0};
void DrawTriangle(int,int,int);

int main()
{
    int n;

    for(int i=0;i<4100;i++)
    {
        for(int j=0;j<4100;j++)
```

```
                _map[i][j]=' ';
        }

        while(~scanf("%d",&n))

            if(n==_DMAX)
            {
                DrawTriangle(_MAX,_MAX,1);

                for(int i=1;i<=_MAX;i++)
                {
                    for(int j=0;j<_MAX;j++)
                        printf(" ");

                    for(int j=1;j<=_DMAX;j++)
                        printf("%c",_map[i][j]);

                    printf("\n");
                }

                for(int i=1;i<=_MAX;i++)
                {
                    for(int j=1;j<=_DMAX;j++)
                        printf("%c",_map[i][j]);

                    for(int j=1;j<=_DMAX;j++)
                        printf("%c",_map[i][j]);

                    printf("\n");
                }

                continue;
            }

            DrawTriangle(n,_MAX,1);

            for(int i=1;i<=n;i++)
            {
                for(int j=1-n;j<=n;j++)
                    printf("%c",_map[i][_MAX+j]);

                printf("\n");
            }
        }
```

```
    return 0;
}

void DrawTriangle(int n,int xpos,int ypos)
{
    if(n==1)
    {
        _map[ypos][xpos]='/';
        _map[ypos][xpos+1]='\\';
        return;
    }

    int m=n/2;

    DrawTriangle(m,xpos,ypos);
    DrawTriangle(m,xpos-m,ypos+m);
    DrawTriangle(m,xpos+m,ypos+m);
}
```

Analysis2（源于寇宇增同学的提醒）

把所有有三角形的地方设为 1，没有的设为 0，并删去行首空格，得到如下表格：

1
11
101
1111
10001
110011
1010101
11111111
.....

仔细观察并寻找规律的话，会发现每行除了最边上的两个数外的其他数都是其上一行同一位置的数和左边的数的异或。以第四行为例，它包含四个数 1111，那么第五行的五个数就是第四行的四个数（11110，空位用 0 补全）与第四行每一位向右移所得到的四个数（01111，空位用 0 补全）求异或，得到 10001 这五个数。于是只用开一个一维数组就够了，总算不用爆内存啦(☆ω☆)~

Code2

```
#include <stdio.h>
#define _MAX 1024
using namespace std;

```

```
short *_map;
void DrawTriangle(int,int);

int main()
{
    int n,temp;

    while(~scanf("%d",&n))
    {
        temp=(n-1)/8+1;
        _map=new short[temp];

        for(int i=0;i<temp;i++)
            _map[i]=0;

        DrawTriangle(n,n);
    }

    delete[] _map;
    return 0;
}

void DrawTriangle(int n,int hwidth)
{
    if(n==1)
    {
        _map[0]=0x80;

        for(int i=n;i<hwidth;i++)
            printf(" ");

        printf("/\\\n");
        return;
    }

    DrawTriangle(n-1,hwidth);

    for(int i=(hwidth-1)/8;i>0;i--)
        _map[i]^=((_map[i]>>1)|((_map[i-1]&0x01)<<7));

    _map[0]^=(_map[0]>>1);

    for(int i=n;i<hwidth;i++)
        printf(" ");
```

```
        for(int i=0,temp=(n-1)/8;i<=temp;i++)
        {
            for(int j=7;j>-1;j--)
            {
                if(((_map[i]>>j)&0x01)==1)
                    printf("/\\");
                else
                    printf("  ");
            }
        }

        printf("\n");
    }
```

G - 冰点

Problem Description

A human baby when will they find out?
That at a point they are born We are
winners of Earth.

给定从 2000 年 1 月 1 日逝去的天数，
请输出这一天的具体日期。

Input

多组数据，每组一个整数 n(n>=0) 保证结果不超过 9999 年（保证数据量小于 1e6)

Output

输出具体日期，格式为"year-month-day dayofweek”
需要前导 0

Sample Input

1
30
365
366

Sample Ouput

2000-01-02 Sunday
2000-01-31 Monday
2000-12-31 Sunday
2001-01-01 Monday

Hint

"Monday","Tuesday","Wednesday","Thursday","Friday","Saturday","Sunday"
难题，慎入

Analysis

其实这道题不算太难啦~~只是情况考虑稍微复杂一些。还有把 2001 年之前和之后分开算会更好算一些。具体方法就是 400 年 400 年的减，再 100 年 100 年的减，再 4 年 4 年的减，再 1 年 1 年的减，最后再判断月数和日期数。这样就完成啦~~ =￣ω￣=

Code

```
#include <stdio.h>
using namespace std;

int main()
{
    int n,_year=0,_month=0,_day=0,_week,temp,
        _years[4]={365,1461,36524,146097},
_months[2][12]={{31,29,31,30,31,30,31,31,30,31,30,31},{31,28,31,30,31,30,31,31,30,31,30,31}};
    bool flag=0;
    char
_weeks[7][10]={"Saturday","Sunday","Monday","Tuesday","Wednesday","Thursday","Friday"};

    while(~scanf("%d",&n))
    {
        _week=n%7;

        if(n<366)
        {
            for(_month=0;_month<12;_month++)
            {
                if(n<_months[0][_month])
```

```
                    {
                        _day=n;
                        break;
                    }
                    else
                        n-=_months[0][_month];
                }

printf("2000-%02d-%02d %s\n",1+_month,1+_day,_weeks[_week]);
                continue;
            }

            n-=366;

            _year=((n/_years[3])*400);
            n%=_years[3];

            temp=n/_years[2];
            temp=(temp==4?3:temp);
            _year+=(temp*100);
            n-=(_years[2]*temp);

            _year+=((n/_years[1])*4);
            n%=_years[1];

            temp=n/_years[0];
            temp=(temp==4?3:temp);
            _year+=temp;
            n-=(_years[0]*temp);

            if(_year%400==399||(_year%4==3&&_year%100!=99))
                flag=0;
            else
                flag=1;

            for(_month=0;_month<12;_month++)
            {
                if(n<_months[flag][_month])
                {
                    _day=n;
                    break;
                }
                else
```

```
                n-=_months[flag][_month];
            }

printf("%d-%02d-%02d %s\n",2001+_year,1+_month,1+_day,_weeks[_week]);
        }

        return 0;
    }
```