

C++方向编程题答案

第一周

day6

题目ID: 45840-不要二

链接: <https://www.nowcoder.com/practice/1183548cd48446b38da501e58d5944eb?tpId=85&&tqId=29840&rp=1&ru=/activity/oj&qru=/ta/2017test/question-ranking>

【题目解析】:

本题看起来很难, 实际是一个中等难度的题。本题如果没记错, 是一个往年网易的笔试题, 大家可以看到大厂的题的难度。

本题的重点是要读懂题意, 并且需要多读两遍, 才能读懂, 本题本质就是在二维数组中每个坐标去放蛋糕, 一个坐标位置放了蛋糕, 跟他欧几里得距离为2的位置不能放蛋糕, 这个就是关键点。对于两个格子坐标 $(x1,y1), (x2,y2)$ 的欧几里得距离为: $((x1-x2) * (x1-x2) + (y1-y2) * (y1-y2))$ 的算术平方根。

也就是说: 如果 $(x1,y1)$ 放了蛋糕, 则满足 $((x1-x2) * (x1-x2) + (y1-y2) * (y1-y2)) == 4$ 的 $(x2,y2)$ 不能放蛋糕。

$((x1-x2) * (x1-x2) + (y1-y2) * (y1-y2)) == 4$ 看起来是一个无解的表达式。

但是可以进行加法表达式分解:

$$1+3=4$$

$$3+1=4$$

$$2+2=4$$

$$0+4=4$$

$$4+0=4$$

仔细分析前三个表达式是不可能的, 因为 $(x1-x2) * (x1-x2)$ 表达式结果不能等于2或3。

也就是说 $(x1-x2) * (x1-x2)$ 和 $(y1-y2) * (y1-y2)$ 两个表达式一个等于0, 一个等于4。

可以看出: 假设放蛋糕的位置是 $(x1,y1)$, 则不能放蛋糕的位置 $(x2,y2)$, 满足 $x1==x2, y1-y2==2$ 或者 $x1-x2==2, y1==y2$ 。

【解题思路】:

仔细读理解了上面的题目解读, 本题就非常简单了, 使用 `vector<vector<int>>` 定义一个二维数组, `resize` 开空间并初始化, 每个位置初始化为1, 表示当蛋糕, $a[i][j]$ 位置放蛋糕, 则可以标记处 $a[i][j+2]$ 和 $a[i+1][j]$ 位置不能放蛋糕, 遍历一遍二维数组, 标记处不能放蛋糕的位置, 统计也就统计出了当蛋糕的位置数。

```
// 直接暴力计算, 默认所有蛋糕的位置标记成1, 不能放的地方标记成0
// 1 1 0 0 1 1
// 1 1 0 0 1 1
// 0 0 1 1 0 0
// 0 0 1 1 0 0
```

```

#include<iostream>
#include<vector>
using namespace std;
int main()
{
    int w,h,res = 0;
    cin >> w >> h;
    vector<vector<int>> a;
    a.resize(w);
    for(auto& e : a)
        e.resize(h, 1);

    for(int i=0;i<w;i++)
    {
        for(int j=0;j<h;j++)
        {
            if(a[i][j]==1)
            {
                res++;
                // 标记不能放蛋糕的位置
                if((i+2)<w)
                    a[i+2][j] = 0;

                if((j+2)<h)
                    a[i][j+2] = 0;
            }
        }
    }
    cout << res;
    return 0;
}

```

23292-字符串转成整数

<https://www.nowcoder.com/practice/1277c681251b4372bdef344468e4f26e?tpId=13&&tqId=11202&rp=6&ru=/activity/oj&gru=/ta/coding-interviews/question-ranking>

【题目解析】：

本题本质是模拟实现实现C库函数atoi，不过参数给的string对象

【解题思路】：

解题思路非常简单，就是上次计算的结果*10，相当于10进制进位，然后加当前位的值。

例如：“123”转换的结果是

sum=0

sum*10+1->1

sum*10+2->12

sum*10+3->123

本题的关键是要处理几个关键边界条件：

1. 空字符串
2. 正负号处理
3. 数字串中存在非法字符

```
class Solution {
public:
    int StrToInt(string str)
    {
        if(str.empty())
            return 0;

        int symbol = 1;
        if(str[0] == '-') //处理负号
        {
            symbol = -1;
            str[0] = '0'; //这里是字符'0',不是0
        }
        else if(str[0] == '+') //处理正号
        {
            symbol = 1;
            str[0] = '0';
        }

        int sum = 0;
        for(int i=0;i<str.size();++i)
        {
            if(str[i] < '0' || str[i] > '9')
            {
                sum = 0;
                break;
            }

            sum = sum *10 + str[i] - '0';
        }
        return symbol * sum;
    }
};
```