# 《C++ 程序设计》OJC（参考答案）

VCG

2023 年 5 月 27 日

# 1 A+B

## 【描述】

小 A 参加了一次考试，已知考试一共有 5 科，现知道他每科的成绩，求小 A 此次考试的总成绩。

## 【输入描述】

输入一行，输入 5 个整数，表示小 A 每科的成绩。每个整数都在范围 [1, 100] 内。

## 【输出描述】

输出一行，输出一个整数，表示小 A 这 5 科的总成绩。

## 【输入样例】

```
98 95 87 100 90
```

## 【输出样例】

```
470
```

## 【提示】

### main.cpp

```cpp
#include<iostream>

int main()
{
    int sum = 0;
    for (auto i = 0; i < 5; i++)
    {
        int score;
        std::cin >> score;
        sum = sum + score;
    }

    std::cout << sum;
    return 0;
}
```

# 2 3n+1

## 【描述】

对于任意大于 1 的自然数 n，若 n 为奇数，则将 n 变为 3n+1，否则变为 n 的一半；经过若干次这样的变换，一定会使 n 变为 1。比如：
3->10->5->16->8->4->2->1
要求：输入 n，输出变换的次数。

## 【输入描述】

输入一行，输入一个正整数 n，1 <= n <= 100。

## 【输出描述】

输出一行，输出一个正整数，表示变换的次数。

## 【输入样例】

3

## 【输出样例】

7

## 【提示】

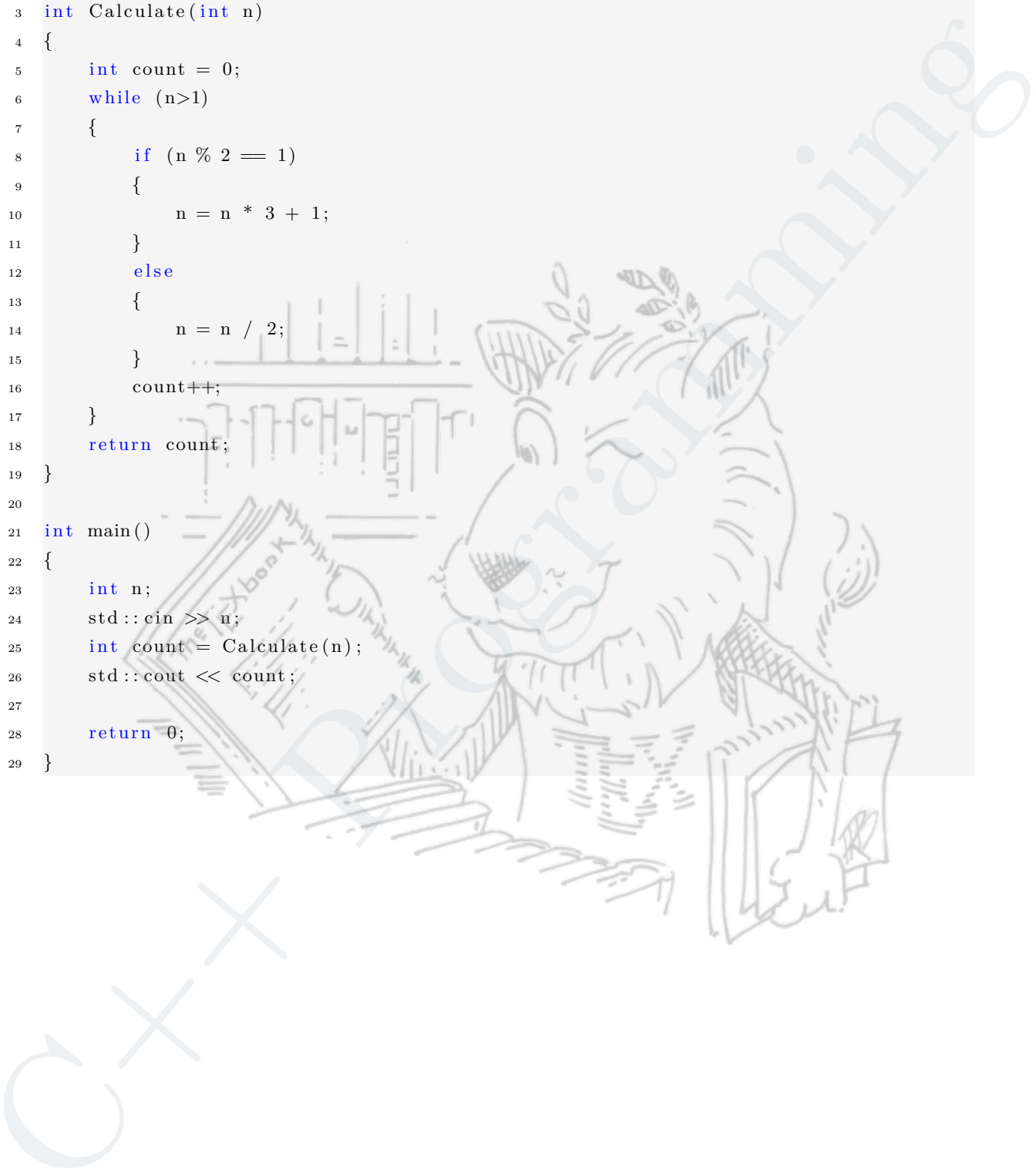
### main.cpp

```cpp
#include<iostream>

int Calculate(int n)
{
    int count = 0;
    while (n>1)
    {
        if (n % 2 == 1)
        {
            n = n * 3 + 1;
        }
        else
        {
            n = n / 2;
        }
        count++;
    }
    return count;
}

int main()
{
    int n;
    std::cin >> n;
    int count = Calculate(n);
    std::cout << count;

    return 0;
}
```



# 3 稀疏矩阵

**【描述】**

大部分元素为 0 的矩阵称为稀疏矩阵。假设有 k 个非 0 元素，则可把稀疏矩阵用 k*3 的矩阵简记之，其中第一列为行号，第二列为列号，第三列是该行、该列下的非 0 元素的值。

**【输入描述】**

首先输入一行，输入两个正整数，表示稀疏矩阵的行号和列号，再输入一个稀疏矩阵，行和列不超过 100。

**【输出描述】**

输出它的简记形式，按从上到下，从左到右的顺序依次输出。每一行输出 3 个整数，每两个整数之间一个空格，最后一个整数后面没有空格。

**【输入样例】**

```
3 4
0 0 0 5
0 2 0 0
0 1 0 0
```

**【输出样例】**

```
1 4 5
2 2 2
3 2 1
```

**【提示】**

### main.cpp

```cpp
#include<iostream>

int main()
{
    int rows, cols;
    std::cin >> rows >> cols;

    for (auto row = 1; row <= rows; row++)
    {
        for (auto col = 1; col <= cols; col++)
        {
            int value;
            std::cin >> value;
            if (value != 0)
            {
                std::cout << row << " " << col << " " << value << std::endl;
            }
        }
    }

    return 0;
}
```

# 4 开灯问题

**【描述】**

有 n 盏灯，编号为 1 n。第 1 个人把所有灯打开，第 2 个人按下所有编号为 2 的倍数的开关（这些灯将被关掉），第 3 个人按下所有编号为 3 的倍数的开关（其中关掉的灯将被打开，开着的灯将被关闭），以此类推。一共有 k 个人，问最后有哪些灯开着?

**【输入描述】**

输入一行，输入两个正整数 n 和 k。(k<=n<=1000)

**【输出描述】**

输出一行，输出所有开着的灯的编号，每个编号后面跟一个空格。

**【输入样例】**

```
7 3
```

**【输出样例】**

```
1 5 6 7
```

**【提示】**

可先使用 string.h 里的 memset 函数将数组全部置 0；使用 C 语言的逻辑非 (!) 操作符

### main.cpp

```cpp
#include<iostream>

int main()
{
    int n, k;
    std::cin >> n >> k;

    int light[1001] = { 0 };

    for (auto p = 1; p <= k; p++)
    {
        for (auto l = 1; l <= n; l++)
        {
            if (l % p == 0)light[l] ++;
        }
    }

    for (auto l = 1; l <= n; l++)
    {
        if (light[l] % 2 == 1)
        {
            std::cout << l << " ";
        }
    }

    return 0;
}
```

# 5 还是一个不妖艳的问题

## 【描述】

U 君与 S 君想测试下双方的缘分，测试规则是两人各想一个数字，如果两个数字互质则代表有缘分。

## 【输入描述】

多组测试，每组测试输入两个数字 m 和 n(m 和 n 均不超过 int 取值范围)，空格隔开。

## 【输出描述】

对于每组，如果有缘分的话，输出“Yes”；否则，输出“No”

## 【输入样例】

```
10 5
17 13
```

## 【输出样例】

```
No
Yes
```

## 【提示】

可使用 `while (scanf("\%d\%d", \&n, \&m) != EOF)` 完成多组输入

### main.cpp

```cpp
#include<iostream>
#define _CRT_SECURE_NO_WARNINGS

//最大公约数（Greatest Common Divisor，GCD，辗转相除法
int GCD(int m, int n)
{
    if (m >= n)
    {
        if (m % n == 0)return n;
        return GCD(m % n, n);
    }
    return GCD(n, m);
}

int main()
{
    int m, n;
    while (std::cin >> m >> n)
    {
        if (GCD(m, n)==1)
            std::cout << "Yes\n";
        else
            std::cout << "No\n";
    }

    return 0;
}
```

# 6 回文串的排序

## 【描述】

输入 n 个字符串，将其中是回文串的字符串，按照长度从小到大的顺序输出，如果长度相同，则按照输入的顺序输出即可。
回文的含义是：正着看和倒着看相同，如 abba 和 yyxyy。

## 【输入描述】

第一行首先是一个整数 n（n<50），表示有 n 个字符串（每个字符串的都长度小于 100）。接下来有 n 行，每行输入一个字符串。

## 【输出描述】

按照长度从小到大的顺序输出其中的回文串，如果长度相同，则按照输入的顺序输出。

## 【输入样例】

```
4
abbb
abccba
abba
acca
```

## 【输出样例】

```
abba
acca
abccba
```

## 【提示】

### main.cpp

```cpp
#define _CRT_SECURE_NO_WARNINGS
#include<iostream>
#include<cstring>

bool PalindromicWord(const char* word)
{
    int len = strlen(word);
    for (auto i = 0; i < len / 2; i++)
    {
        if (word[i] != word[len - 1 - i])return false;
    }
    return true;
}


void bubble_sort(char arr[][101], int len)
{
    for (auto i = 0; i < len - 1; i++)
    {
        for (auto j = 0; j < len - 1 - i; j++)
        {
            if (strlen(arr[j]) > strlen(arr[j + 1]))
            {
                char temp[101];
                sprintf(temp, "%s", arr[j]);
                sprintf(arr[j], "%s", arr[j + 1]);
                sprintf(arr[j + 1], "%s", temp);
            }
        }
    }
}

void show(char arr[][101], int len)
{
    for (auto i = 0; i < len; i++)
    {
        std::cout << arr[i] << std::endl;
    }
}

int main()
{
    char strsIn[50][101];
    int n, count=0;
    std::cin >> n;
    for (auto i = 0; i < n; i++)
    {
        char str[1001];
```

```
49          std::cin >> str;
50          bool flag = PalindromicWord(str);
51          if (flag)
52          {
53              sprintf(strsIn[count], "%s", str);
54              count++;
55          }
56      }
57      bubble_sort(strsIn, count);
58      show(strsIn, count);
59
60      return 0;
61  }
```

# 7 蛇形填数

## 【描述】

在 n*n 方阵里填入 1,2,…,n*n，要求填成蛇形。例如 n=4 时方阵为：

```
10 11 12 1
9 16 13 2
8 15 14 3
7 6 5 4
```

## 【输入描述】

本题为多组输入。每组数据输入一行，输入一个正整数 n。n<=8。

## 【输出描述】

输出 n*n 的蛇形矩阵，每行中每两个数之间用一个空格隔开，每行最后一个数后面没有空格。

## 【输入样例】

```
4
```

## 【输出样例】

```
10 11 12 1
9 16 13 2
8 15 14 3
7 6 5 4
```

## 【提示】

### main.cpp

```cpp
#include<iostream>

void SnakeGame(int n)
{
    int state[10][10] = { 0 };

    int row = 0, col = n - 1;
    int index = 1;
    state[row][col] = index++;
    int direction = 1;
    while (index <= n * n)
    {
        int olddir = direction;
        if (direction % 4 == 1)
        {//向下
            if (row < n - 1 && state[row + 1][col] == 0) row++;
            else direction++;
        }
        else if (direction % 4 == 2)
        {//向左
            if (col > 0 && state[row][col - 1] == 0) col--;
            else direction++;
        }
        else if (direction % 4 == 3)
        {//向上
            if (row > 0 && state[row - 1][col] == 0) row--;
            else direction++;
        }
        else if (direction % 4 == 0)
        {//向右
            if (col < n - 1 && state[row][col + 1] == 0) col++;
            else direction++;
        }
        if (direction == olddir)
            state[row][col] = index++;
    }

    for (auto row = 0; row < n; row++)
    {
        for (auto col = 0; col < n; col++)
        {
            std::cout << state[row][col];
            if (col < n - 1)std::cout << " ";
        }
        std::cout << "\n";
    }
}

```

```
49 int main()
50 {
51     int n;
52     while (std::cin >> n)
53     {
54         SnakeGame(n);
55     }
56 
57     return 0;
58 }
```