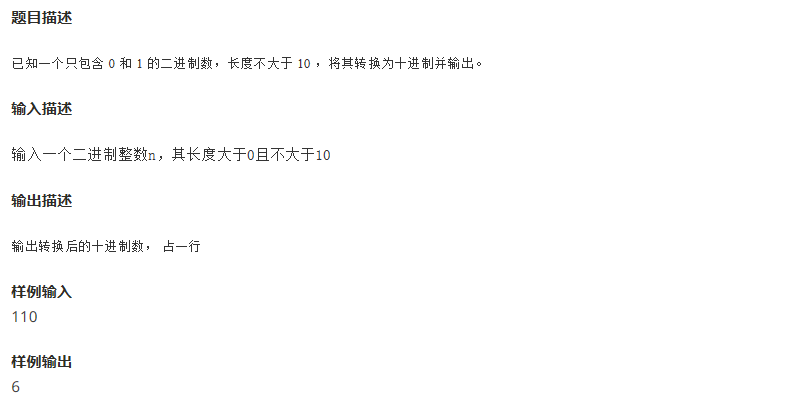
# 第二章

## 1.简单题目



## 2.进制转换



## 3.实心菱形



## 4.整数运算



## 5.打印正方形



## 6.打印空心正方形



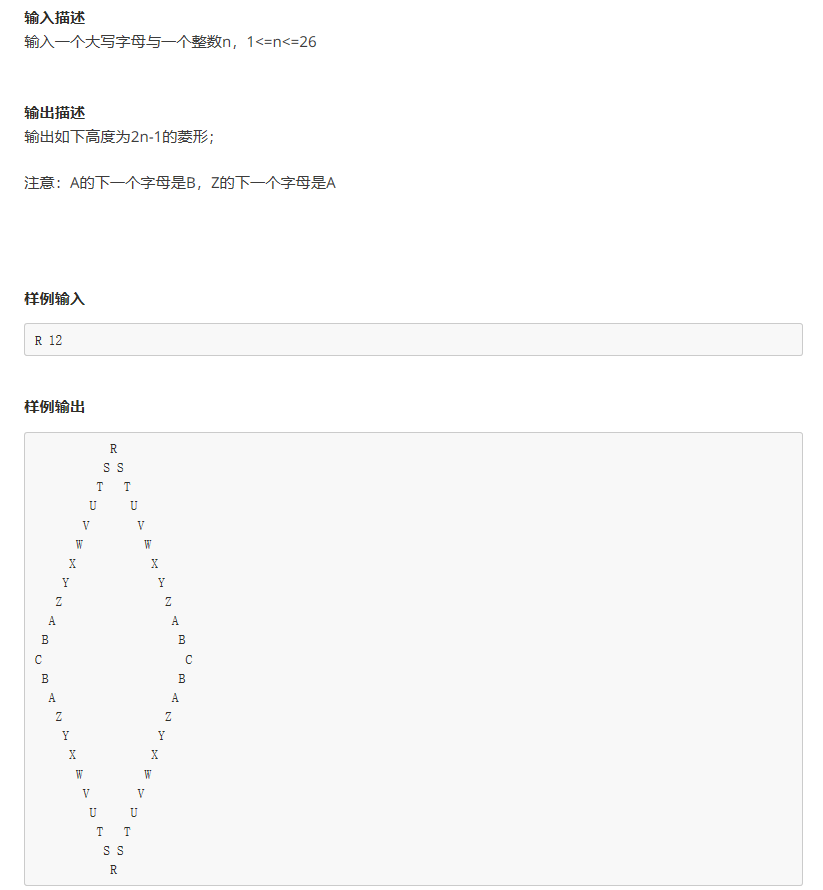
## 7.数字正方形



## 8.螺旋数字正方形



## 9.字母菱形



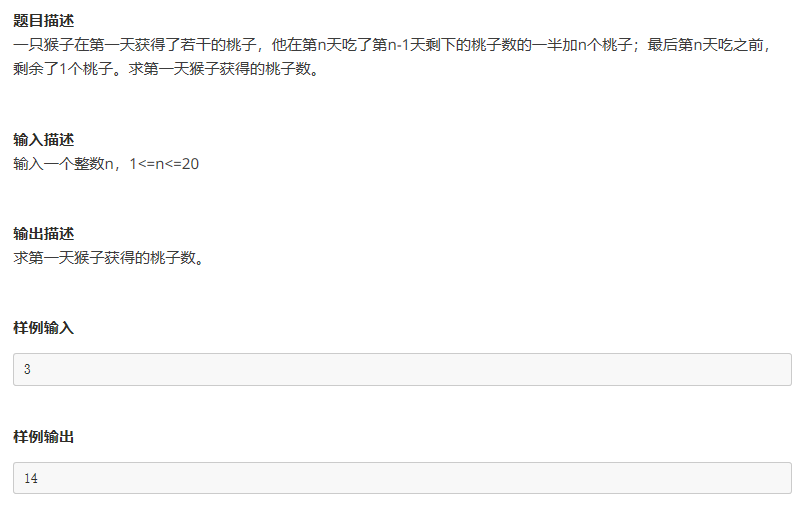
## 10.打印数字梯形



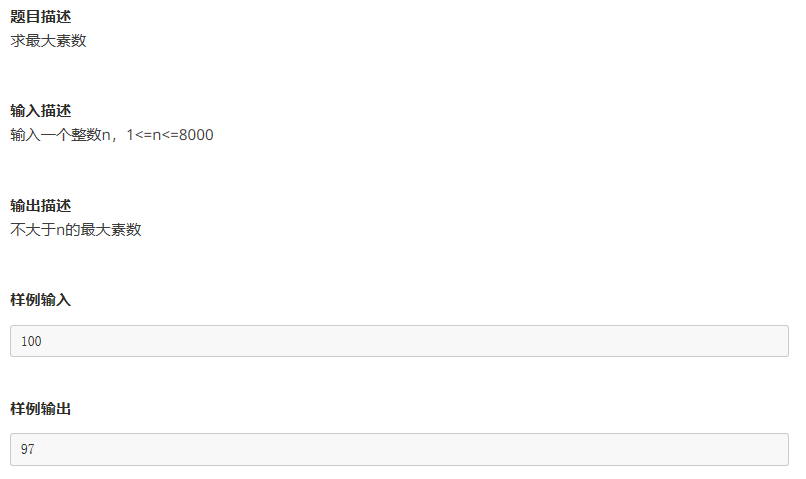
## 11.求和



## 12.猴子吃桃



## 13.最大素数



## 14.最后三位数值



## 15.最长等值数列



## 16.蜗牛问题



## 17.出现次数最多的数字



## 18.圆的面积和周长



## 19.完美数



## 20.平方数之和

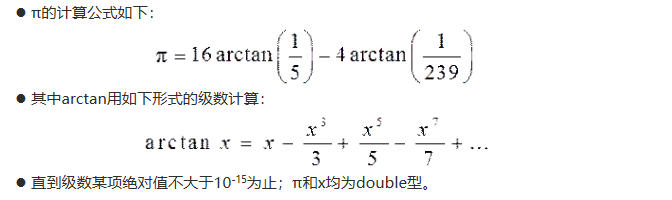


## 21.水仙花数



# 第三章

## 1. 编写程序求π的值



## 最大公约数



## 回文字符串



## 楼梯的走法

**题目描述**

假设有n阶楼梯，小明一次可以选择走1阶，也可以选择走2阶，问多少种走法。比如有5阶台阶，共有8种走法。

1 1 1 1 1

1 1 1 2

1 1 2 1

1 2 1 1

1 2 2

2 1 1 1

2 1 2

2 2 1

要求使用递归解法。  
**输入描述**

输入n，表示n阶楼梯，1<=n<=40  
**输出描述**

输出楼梯的走法总数。  
**样例输入**

5

**样例输出**

8

## 开灯关灯1

**题目描述**

有n盏灯，编号依次为1,2,3,...,n；初始化时，所有灯都是关闭状态；

小明第一次将编号为1,2,3,...,共n盏灯都打开了；第二次将编号为2,4,6,....,共n/2盏灯都关闭了；第三次将编号为3,6,9,...,共n/3盏灯进行操作，若是之前关闭状态则打开，若是打开状态，则关闭；第四次将4,8,12,....,共n/4盏灯进行操作。这样操作了n次，问最后亮了几盏灯。  
**输入描述**

第一行输入整数T，表示有T组测试用例；1<=T<=100

接下来T行，每行有一个整数n，表示当前测试用例有n盏灯；1<=n<=1000  
**输出描述**

输出n行，每次测试用例最后亮灯的数目。  
**样例输入**

2

1

5

**样例输出**

1

2

## 开灯关灯2

**题目描述**

有n盏灯，编号依次为1,2,3,...,n；初始化时，所有灯都是关闭状态；

小明第一次将编号为1,2,3,...,共n盏灯都打开了；第二次将编号为2,4,6,....,共n/2盏灯都关闭了；第三次将编号为3,6,9,...,共n/3盏灯进行操作，若是之前关闭状态则打开，若是打开状态，则关闭；第四次将4,8,12,....,共n/4盏灯进行操作。这样操作了n次，问最后亮了几盏灯。  
**输入描述**

第一行输入整数T，表示有T组测试用例；1<=T<=10000

接下来T行，每行有一个整数n，表示当前测试用例有n盏灯；1<=n<=10000

提示：总结开灯关灯规律。  
**输出描述**

输出n行，每次测试用例最后亮灯的数目。  
**样例输入**

2

1

5

**样例输出**

1

2

## A+B

**题目描述**

输入两个整数A,B，求A+B  
**输入描述**

第一行，整数A

第二行，整数B

A,B位数长度均小于160  
**输出描述**

A+B的结果  
**样例输入**

1234567890987654321099999

12345678901

**样例输出**

1234567890987666666778900

# 第四章

## 1.最大公约数

**题目描述**

求两个正整数a 和 b的最大公约数。

要求使用c++ class编写程序。可以创建如下class

#include <iostream>

using namespace std;

class Integer {

private:

int \_num;

public:

//构造函数

Integer(int num) {

}

//计算当前Integer 和 b之间的最大公约数

int gcd(Integer b) {

}

};

int main(){

int a, b;

cin >> a >> b;

Integer A(a);

Integer B(b);

cout << A.gcd(B) << endl;

return 0;

}

**输入描述**

两个正整数a,b，并且1=<a,b <=10000

**输出描述**

a和b的最大公约数

**样例输入**

1000 1000

**样例输出**

1000

## 2.反转整数

**题目描述**

对于输入的一个正整数，输出其反转形式

要求使用c++ class编写程序。可以创建如下class

#include <iostream>

using namespace std;

class Integer{

private:

int \_num;

//getLength()函数获取\_num长度

int getLength(){

}

public:

//Integer类构造函数

Integer(int num){

}

//反转\_num

int inversed(){

}

};

int main() {

int n;

cin >> n;

Integer integer(n);

cout << integer.inversed() << endl;

return 0;

}

**输入描述**

一个正整数a ，且1=<a<=1,000,000,000

**输出描述**

a的反转形式

**样例输入**

1011

**样例输出**

1101

## 3.一元二次方程求解

**题目描述**

对于一元二次方程ax^2 + bx + c = 0,解可以分为很多情况。

若该方程有两个不相等实根，首先输出1，换行，然后从小到大输出两个实根，换行；

若该方程有两个相等实根，首先输出2，换行，然后输出这个这个实根，换行；

若该方程有一对共轭复根，输出3，换行；

若该方程有无解，输出4，换行；

若该方程有无穷个解，输出5，换行；

若该方程只有一个根，首先输出6，换行，然后输出这个跟，换行；

要求使用c++ class编写程序。可以创建如下class

#include <iostream>

#include <cmath>

using namespace std;

class Equation{

private:

int \_a, \_b, \_c;

public:

Equation(int a, int b, int c){

}

void solve(){

}

};

int main(){

int a, b, c;

cin >> a >> b >> c;

Equation tmp(a, b, c);

tmp.solve();

return 0;

}

**输入描述**

该一元二次方程的系数a,b,c，且-100=<a,b,c<=100

**输出描述**

解的情况。输出解的时候保留两位小数

**样例输入**

1 4 3

**样例输出**

1

-3.00 -1.00