//知识点：运算符重载、动态内存管理

1.已知字符串类 MyString 的定义为：

```
class MyString
{
public:
    MyString(const char* pData= NULL);          // 普通构造函数
    MyString(const MyString &);                 // 拷贝构造函数
    ~ MyString();                               // 析构函数
    MyString & operator =(const MyString &);    // 赋值函数
    MyString& operator += (const MyString &);
    operator char* () const;                    // 自动转换函数
private:
    char *mpData;                               // 用于保存字符串
};
```

全局函数：

```
const MyString operator + (const MyString &,const MyString &);//字符串连接
ostream& operator<<(ostream& os, const MyString& str); //定向输出
```

请完整实现 MyString 类和指定的全局函数，可以使用 new，delete 运算以及 strcpy，strlen，…等库函数，还可以定义其他辅助函数。

2.对任意一个正的实数，总可以唯一地表示成 a0+1/(a1+1/(a2+1/(a3+1/.....)))的形式，简记为 a0+a1+a2+a3+...+ak，ai 为大于 0 的正整数，长度为 k，若数为无理数，则 k 为无穷大，并称这种形式的数为连分数。

a)请设计并实现连分数类, 此类的每个对象代表一个数的连分数形式，且假定连分数的长度均小于 MAXLEN（MAXLEN 假设为 30）。该类的主要功能有：

1)对指定的 i，返回 ai 值；

2)计算连分数的前 q 项（a0+a1+a2+a3+..+aq）所对应分数的分子和分母；

3)输出此分数的前 q 项，格式为:a0+a1+a2+a3+..+aq；

b)给出主程序，使用该类计算连分数的前 n 项对应的分数逼近 PI 值的程度（计算差即可，PI 可用<cmath>头文件中的 M_PI 常量）。

例如主程序的输出可能如下：

```
PI=3.141592653589793

前 1 项为 3
前 1 项对应分数为 3/1
前 1 项对应分数的值为 3
前 1 项对应分数与 3.141592653589793 的差为 0.1415926535897931

前 2 项为 3+7
前 2 项对应分数为 22/7
前 2 项对应分数的值为 3.142857142857143
```

前 2 项对应分数与 3.141592653589793 的差为-0.001264489267349741

前 3 项为 3+7+15
前 3 项对应分数为 333/106
前 3 项对应分数的值为 3.141509433962264
前 3 项对应分数与 3.141592653589793 的差为 8.321962752896503e-005

前 4 项为 3+7+15+1
前 4 项对应分数为 355/113
前 4 项对应分数的值为 3.141592920353983
前 4 项对应分数与 3.141592653589793 的差为-2.667641891848736e-007

前 5 项为 3+7+15+1+292
前 5 项对应分数为 103993/33102
前 5 项对应分数的值为 3.141592653011902
前 5 项对应分数与 3.141592653589793 的差为 5.778905119192823e-010

前 6 项为 3+7+15+1+292+1
前 6 项对应分数为 104348/33215
前 6 项对应分数的值为 3.141592653921421
前 6 项对应分数与 3.141592653589793 的差为-3.316279286770529e-010

前 7 项为 3+7+15+1+292+1+1
前 7 项对应分数为 208341/66317
前 7 项对应分数的值为 3.141592653467437
前 7 项对应分数与 3.141592653589793 的差为 1.223564103768754e-010

前 8 项为 3+7+15+1+292+1+1+1
前 8 项对应分数为 312689/99532
前 8 项对应分数的值为 3.141592653618936
前 8 项对应分数与 3.141592653589793 的差为-2.914350748575711e-011

前 9 项为 3+7+15+1+292+1+1+1+2
前 9 项对应分数为 833719/265381
前 9 项对应分数的值为 3.141592653581078
前 9 项对应分数与 3.141592653589793 的差为 8.715344852056051e-012

前 10 项为 3+7+15+1+292+1+1+1+2+1
前 10 项对应分数为 1146408/364913
前 10 项对应分数的值为 3.141592653591404
前 10 项对应分数与 3.141592653589793 的差为-1.610862485068587e-012

.....

3.在全局函数 void f(int n,int m)中动态建立一个大小为 n*m 的二维整数数组，并对每个数组元素依次赋值为 1，2，…，m*n，再输出每行和每列元素的和，退出函数前释放此数组。

4.已有类 A 声明如下：

```
class A
{
public:
    A(int n):data(n) { }
    int Data() const { return data; }
private:
    int data;
};
```

在全局函数 void g(int n)中动态建立一个大小为 n 的一维指针数组，数组元素指向 A 类对象，各对象的 data 数据成员各不相同，分别为 1，2，3，…，n。创建数组后，使用（例如输出）各对象的 data 值，最后释放此数组。

5.建立一个二维整数数组类，使得其大小可动态决定，并且访问时，可以像普通数组一样使用。例如，建立这个二维整数数组类的一个对象 obj 后，访问其第二行、第三列元素，可写成：obj[1][2] = 5; cout << obj[1][2];