题目描述

多项式的知识相信大家已经在中学数学中学习过了，相关知识应该已经烂熟于心。

本次实验需要实现多项式类，完成多项式加、减等基本运算以及多项式输出、值计算等操作。首先，对多项式做如下的约定：

- 多项式中只会出现一个变量：x
- 每个项的系数均为整数；次数是自然数，在0-99范围内，当次数等于0时，代表常数项。
- 相同次数的项只能出现一次，也就是说需要合并同类项，$2x^2 + 3x^2 + 6x$ 应该写成$5x^2 + 6x$
- 系数为1时，省略系数。不出现$1x^3$之类的情况，应该相应地写作x^3

以下的多项式都是不合法的：

$$\frac{1}{2}x^2 + 3x + 1 \quad (\text{系数不可以出现分数})$$

$$3x^{-2} + 5x^8 \quad (\text{次数不可为负数})$$

$$3x^3 + (-2x^6) + 1 \quad (\text{应写为} 3x^3 - 2x^6 + 1)$$

$$x + 1 - (-x^2) \quad (\text{应写为} x + 1 + x^2)$$

Tips：

为**降低**实验难度，本次实验**提供部分源码**供同学们使用，在实验要求中的注意事项有说明，请**仔细**查看！

实验要求

实现类**Polynomial**，至少包含以下接口：

```
class Polynomial {
private:
  // 记录不同次数的项系数，例如：index = 3，  coefficient_array[index] = 4，代表4x3的项
  int* coefficient_array;
  // 记录在构造时的传入原始表达式的字符串
  char* original_expr;

public:
  // 构造函数，通过字符串初始化出对应的多项式对象
  // 需要为两个指针分配对应的空间
  Polynomial(const char*);

  // 拷贝构造函数，以拷贝的方式初始化一个对象
  // 请注意拷贝构造函数中需要对数据成员进行拷贝，而不是简单的直接赋值操作
  Polynomial(const Polynomial&);
```

```
    // 析构函数，释放申请的空间内存
    ~Polynomial();

    // 返回表示原始表达式的字符串指针
    char* getOriginal();

    // 输出次数最高的项，若多项式为空，则输出0，不需要换行
    void printHighest();

    // 输出按次数降序的多项式，若多项式为空，则输出0，不需要换行
    void printExpression();

    // 计算在 x = x1时的表达式值
    int computeValue(int x1);

    // 重载操作符+，实现多项式加法
    Polynomial operator + (const Polynomial&) const;

    // 重载操作符-，实现多项式减法
    Polynomial operator - (const Polynomial&) const;

    // 重载操作符=，实现多项式类的赋值操作
    void operator = (const Polynomial&);

    // 重载操作符==，用于比较两个多项式是否相等
    bool operator == (const Polynomial&) const;
}
```

注意

1. 构造函数中，char* original_expr不要简单地进行指针赋值操作，需要开辟一块相应的空间，并将构造函数的参数字符串数组的值拷贝到新开辟的空间中。否则会导致错误。
2. 拷贝构造函数中同样需要注意上述事项，请注意两个指针之间直接赋值的操作容易带来无法预料的错误。
3. 字符串解析成int* 数组已经实现，请在OJ网站中下载查看，可以在构造函数中直接使用。
4. 类的析构函数也已经给出，请各位依据个人实现使用。

调用示例

```
Polynomial poly1("x4+2x8-10x5+x2+1");
Polynomial poly2("2x2-5");

poly1.printHighest();
// 输出 2x8
poly1.printExpression();
// 输出 2x8-10x5+x4+x2+1
```

```
cout << poly2.computeValue(1);
// 输出 -3
cout << poly1.getOriginal();
// 输出 x4+2x8-10x5+x2+1

Polynomial poly3 = poly1 + poly2;
// Polynomial + 操作
poly3.printExpression();
// 输出 2x8-10x5+x4+3x2-4
Polynomial poly4 = poly1 - poly2;
// Polynomial - 操作
poly4.printExpression();
// 输出 2x8-10x5+x4-x2+6
Polynomial poly5 = poly1 - poly1;
poly5.printExpression();
// 输出 0

Polynomial poly6(poly1);
poly6.printExpression();
// 输出 2x8-10x5+x4+x2+1
cout << poly6.getOriginal();
// 输出 x4+2x8-10x5+x2+1
```

Tips:

char* 转int，可能会用到的函数： int atoi(const char*)，将字符串表示转换为对应的int值。

int转char*，参考操作：

```
#include<string>
......
int num = 123;
char* c = to_string(num).c_str();
 // c指向的内容为"123"
```

注意

- 请正确处理头文件和实现文件之间的关系，文件、函数的命名严格按照给定要求，注意大小写。
- 将2个文件(Polynomial.cpp, Polynomial.h)打包成ZIP压缩包上传(ZIP 包中不要包含文件夹或者其他文件)。
- 请不要在你提交的代码中包含main函数。