Project1 实验报告

22336216　陶宇卓

1、 程序功能简要说明。

该一元稀疏多项式计算器使用链表，具有下列功能：

输入两个多项式 pa、pb 并即刻展示到状态区，按指数降序排列；

计算 pa + pb 并保存到 pc 中；

计算 pa - pb 并保存到 pc 中；

计算 pa * pb 并保存到 pc 中；

计算 pa 或 pb 的导函数并保存到 pd 中；

计算 pa 或 pb 在特定 x 处的值并输出。

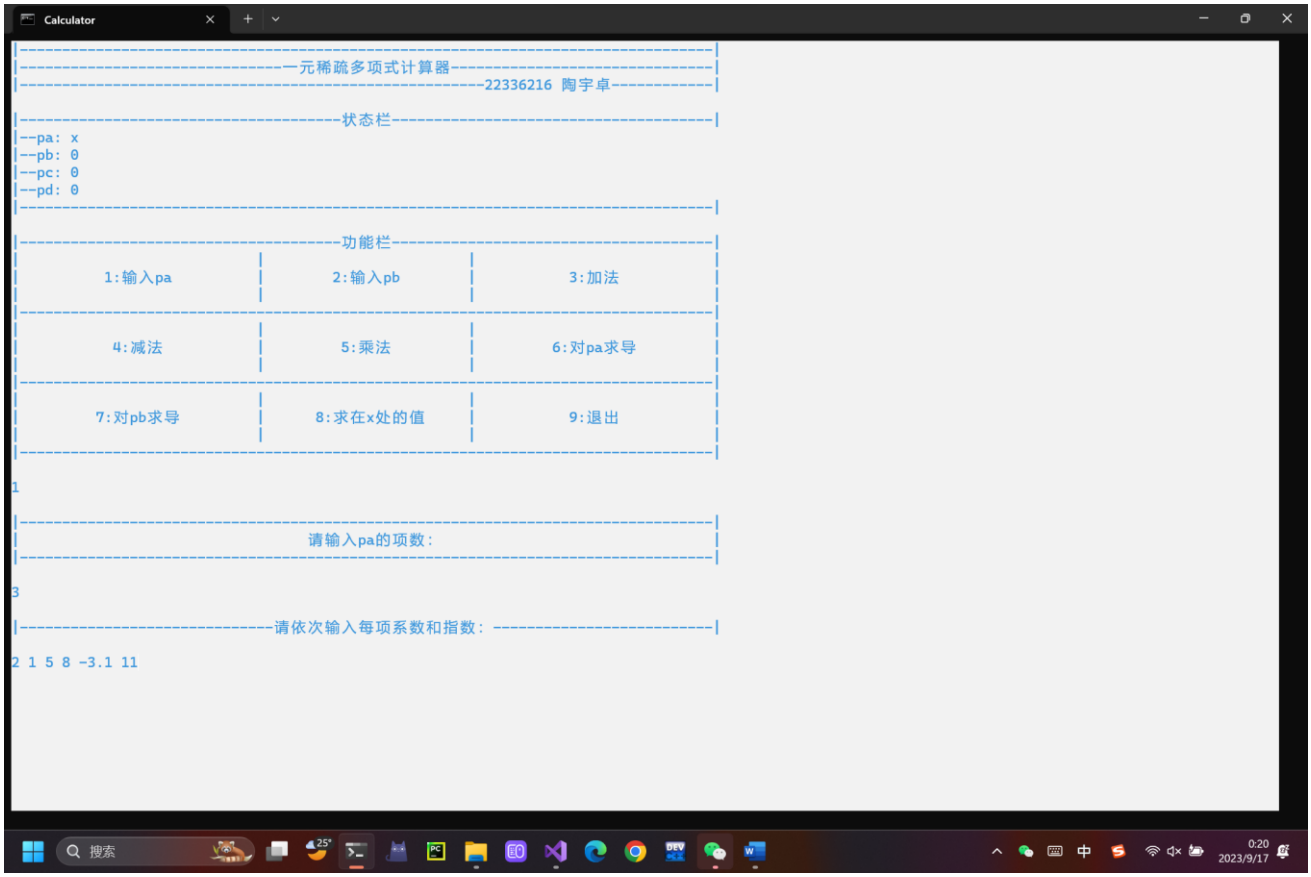
程序可能还会有漏洞，但是大部分错误程序都可以识别。

2、 程序运行截图，包括计算功能演示、部分实际运行结果展示、命令行或交互式界面效果等。

接下来以测试数据 1 $(2x+5x^8-3.1x^{11})+(7-5x^8+11x^9)=(-3.1x^{11}+11x^9+2x+7)$ **为例**

初始界面：

输入 pa：



输入回车，清屏输出 pa，显示在状态栏里：

同理按 2 输入 pb 并显示：

按 3 计算 pa + pb：

除此之外还可以输入其他功能数进行其他计算：

减法：


Calculator
一元稀疏多项式计算器
22336216 陶宇卓
状态栏
--pa: -3.1x^11+5x^8+2x
--pb: 11x^9-5x^8+7
--pc: -3.1x^11-11x^9+10x^8+2x-7
--pd: 0
功能栏
1:输入pa
2:输入pb
3:加法
4:减法
5:乘法
6:对pa求导
7:对pb求导
8:求在x处的值
9:退出

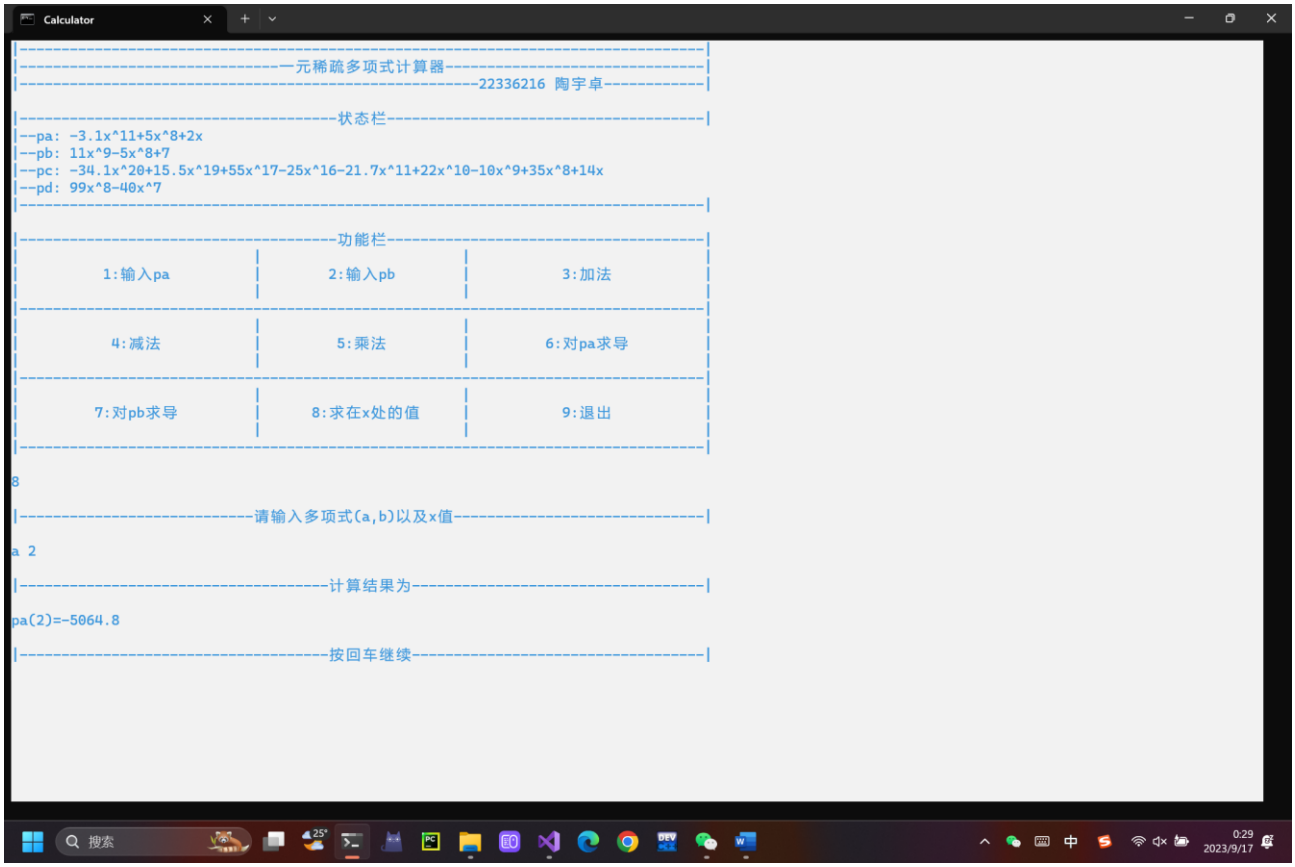
乘法：


Calculator
一元稀疏多项式计算器
22336216 陶宇卓
状态栏
--pa: -3.1x^11+5x^8+2x
--pb: 11x^9-5x^8+7
--pc: -34.1x^20+15.5x^19+55x^17-25x^16-21.7x^11+22x^10-10x^9+35x^8+14x
--pd: 0
功能栏
1:输入pa
2:输入pb
3:加法
4:减法
5:乘法
6:对pa求导
7:对pb求导
8:求在x处的值
9:退出

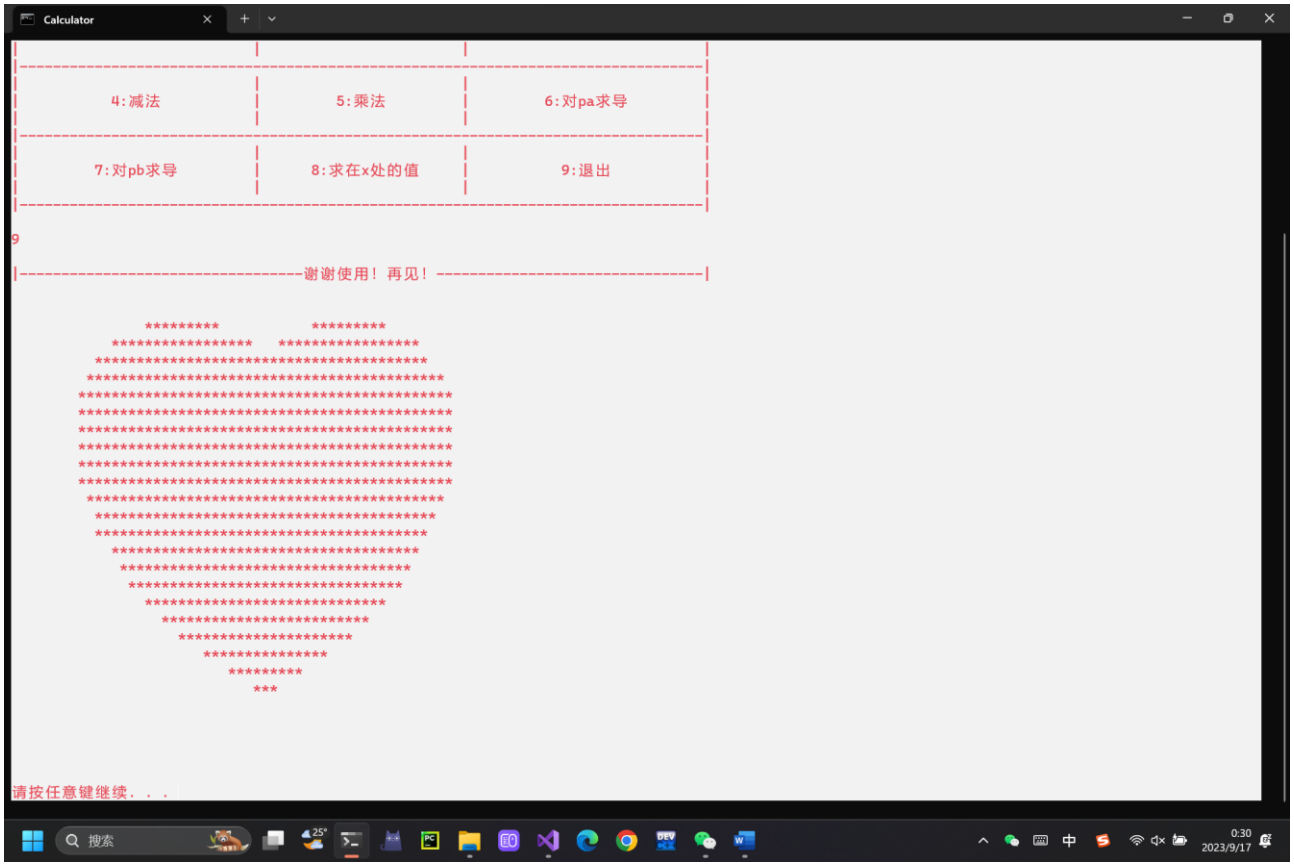
对 pa 求导：

对 pb 求导：

计算 pa 在 x=2 时的值：



最后是退出：



3、 部分关键代码及其说明。

链表定义以及初始化：定义结构体，包含 float 类型的系数 coef，int 类型的指数 expo。初始化链表：为自定义的链表分好空间并创建头节点，置空。

```
typedef struct pnode {
    float coef = 0.00;      //系数
    int expo = 0;       //指数
    struct pnode* next = NULL;      //指针
}pnode, * polynomial;
int InitList(polynomial& L) {
    L = new pnode;  //为创建的链表自动分配空间
    L->next = NULL; //创建头结点，其next域置为NULL
    return 0;
}//初始化
```

将元素压入链表并完成按指数降序排列：遍历链表，找到当前数据应该在的位置并插入。

```
void Push(polynomial& px, float coef, int expo) {
    polynomial q, pre, p3;
    q = px->next;
    pre = px;
    p3 = new pnode;
    p3->coef = coef;
    p3->expo = expo;
    while (q && q->expo > p3->expo) {
        pre = q;
        q = q->next;
    }
    if (q && q->expo == p3->expo) {
        q->coef += p3->coef;
        return;
    }
    p3->next = q;
    pre->next = p3;
    return;
}//
```

创建多项式：根据所输入的项数定义循环次数，循环输入数据并 Push 进链表里。

```cpp
void CreatePolyn(polynomial& p, int n) {
    float coe;
    int exp;
    cout << endl;
    cout << "|----------------------------请依次输入每项系数和指数：-------------------------|" << endl << endl;
    for (int i = 1; i <= n; ++i) {
        cin >> coe;
        while (cin.fail()) {
            cin.clear();
            cin.ignore(10000000, '\n');
            cout << endl << "|-----------------------------------输入错误！---------------------------------|" << endl << endl;
            cin >> coe;
        }
        if (coe) {
            cin >>exp;
            while (cin.fail()) {
                cin.clear();
                cin.ignore(10000000, '\n');
                cout << endl << "|-----------------------------------输入错误！---------------------------------|" << endl << endl;
                cin >> coe >> exp;
            }
        }
        else {
            if (n == 1) {
                rewind(stdin);
                continue;
            }
            cin >> exp;
            continue;
        }
        Push(p, coe, exp);
    }
    rewind(stdin);
}
```

输出函数：

```cpp
void Printresult(polynomial& p) {
    pnode* q;   //用一个指针指向需要输出的链表
    q = p->next;    //指向它的首元结点
    if (!q) {
        cout << "0";
        return;
    }
    if (q->coef == 1) {
        if (q->expo == 1) cout << "x";
        else if (q->expo == 0) cout << 1;
        else cout << "x^" << q->expo;
    }
    else if (q->coef == -1) {
        if (q->expo == 1) cout << "-x";
        else if (q->expo == 0) cout << -1;
        else cout << "-x^" << q->expo;
    }
    else {
        if (q->expo == 1) cout << q->coef << "x";
        else if (q->expo == 0) cout << q->coef;
        else cout << q->coef << "x^" << q->expo;
    }
    q = q->next;    //让指针指向首元结点的下一个结点，后面的就只需要做一个循环解决系数和指数问题
    while (q) {
        if (q->coef == 1) {
            if (q->expo == 1) cout << "+x";
            else if (q->expo == 0) cout << "+1";
            else cout << "+x^" << q->expo;
        }
        else if (q->coef == -1) {
            if (q->expo == 1) cout << "-x";
            else if (q->expo == 0) cout << "-1";
            else cout << "-x^" << q->expo;
        }
        else {
            if (q->coef > 0) {
                if (q->expo == 1) cout << "+" << q->coef << "x";
                else if (q->expo == 0) cout << "+" << q->coef;
                else cout << "+" << q->coef << "x^" << q->expo;
            }
            else {
                if (q->expo == 1) cout << q->coef << "x";
                else if (q->expo == 0) cout << q->coef;
                else cout << q->coef << "x^" << q->expo;
            }
        }
        q = q->next;    //指针再移动到下一个结点
    }
}
```

刷新函数：每次执行完功能后都要 system(“cls”)清屏，这个函数能打印开头并刷新 pa，pb，pc，pd 的实时状态：

```
void Cls(polynomial& pa, polynomial& pb, polynomial& pc, polynomial& pd) {
    cout << "|--------------------------------------------------------------------------------------------|" << endl;
    cout << "|----------------------------一元稀疏多项式计算器----------------------------|" << endl;
    cout << "|-----------------------------------------------22336216 陶宇卓------------|" << endl;
    cout << endl << "|-----------------------------------状态栏-----------------------------------|";
    cout << endl << "|--pa: ";
    Printresult(pa);
    cout << endl << "|--pb: ";
    Printresult(pb);
    cout << endl << "|--pc: ";
    Printresult(pc);
    cout << endl << "|--pd: ";
    Printresult(pd);
    cout << endl << "|--------------------------------------------------------------------------------------------|" << endl << endl;
}
```

加法函数：初始化 pc，遍历 pa，pb。指数相同时比较系数之和与 0 的关系并将元素 push 进链表里。指数不同时分别计算并插入，减法同理

```
void AddPolyn(polynomial& pa, polynomial& pb, polynomial& pc, polynomial& pd) {
    delete pc;
    InitList(pc);
    float sum = 0.00;   //用于保存指数相同时，系数相加后的结果
    polynomial p1, p2;
    p1 = pa->next;
    p2 = pb->next;
    while (p1 && p2) {
        if (p1->expo == p2->expo) {
            sum = p1->coef + p2->coef;
            if (sum != 0) {
                Push(pc, sum, p1->expo);
                p1 = p1->next;
                p2 = p2->next;
            }
            else {
                p1 = p1->next;
                p2 = p2->next;
            }
        }
        else if (p1->expo < p2->expo) {
            Push(pc, p2->coef, p2->expo);
            p2 = p2->next;
        }
        else {
            Push(pc, p1->coef, p1->expo);
            p1 = p1->next;
        }
    }
    if (p1) {
        while (p1) {
            Push(pc, p1->coef, p1->expo);
            p1 = p1->next;
        }
    }
    else if (p2) {
        while (p2) {
            Push(pc, p2->coef, p2->expo);
            p2 = p2->next;
        }
    }
    cout << endl;
    system("cls");
    Cls(pa, pb, pc, pd);
}
```

乘法函数：设置两个 while 循环，从 pa 的第一项开始遍历 pb，依次将他们的系数和指数相乘并将新元素插入 pc 中。

```
void MultiplyPolyn(polynomial pa, polynomial pb, polynomial& pc, polynomial& pd){
    delete pc;
    InitList(pc);
    polynomial p1, p2;
    p1 = pa->next;
    p2 = pb->next;
    if (p1 == nullptr || p2 == nullptr) {
        cout << endl;
        system("cls");
        Cls(pa, pb, pc, pd);    //将pc打印出来
        return;
    }
    while(p1){
        while(p2){
            int sum = 0;    //保存指数相加的结果
            sum = p1->expo + p2->expo;      //保存指数相加的结果
            Push(pc, p1->coef * p2->coef, sum);
            p2 = p2->next;
        }
        p2 = pb->next;
        p1 = p1->next;
    }
    cout << endl;
    system("cls");
    Cls(pa, pb, pc, pd);
}
```

求值函数：从多项式 p 的第一项开始遍历，求出的每一项的结果累加到 double 类型的 sum 里面并传出。

```
double Value(polynomial& p, double x) {
    double sum = 0.00;
    double tmp = 0.00;
    polynomial p1;
    p1 = p->next;
    while (p1) {
        sum += double(double(p1->coef) * pow(x, p1->expo));
        p1 = p1->next;
    }
    return sum;
}
```

求导函数：初始化 pd，从 p 的第一项开始遍历，如果指数不为零，则把该项的系数与指数相乘作为导函数对应项的系数，原来项的指数减 1 作为导函数对应项的指数，并刷新状态栏。

```
void DerivativeA(polynomial& pa, polynomial& pb, polynomial& pc,polynomial& pd) {
    delete pd;
    InitList(pd);
    polynomial p1 = pa->next;
    while (p1) {
        if (p1->expo != 0) {
            Push(pd, (p1->coef * p1->expo), (p1->expo) - 1);
            p1 = p1->next;
        }
        else p1 = p1->next;
    }
    cout << endl;
    system("cls");
    Cls(pa, pb, pc, pd);
}
```

4、 程序运行方式简要说明。

首先，创建并初始化 pa，pb，pc，pd。当输入非法数据时程序会让用户重新输入。打印开头和功能栏。定义字符串 f，进入，输入对应的数字可以执行相应的功能，输入 9 退出。

输入 1，主函数代码如下：

```
if (f == "1") {
    delete pa;
    InitList(pa);
    int n;
    cout << endl << "|-------------------------------------------------------------------------------------------|" << endl;
    cout << "|                                    请输入pa的项数：                                       |" << endl;
    cout << "|-------------------------------------------------------------------------------------------|" << endl << endl;
    cin >> n;
    while (n == 0||cin.fail()) {
        cin.clear();
        cin.ignore(10000, '\n');
        cout << endl << "|-----------------------------------输入错误！-----------------------------------|" << endl << endl;
        cin >> n;
    }
    if(!cin.fail()) {
        rewind(stdin);
    }
    CreatePolyn(pa, n);
    system("cls");
    Cls(pa, pb, pc, pd);
}
```

输入 2，主函数代码如下：

```
else if (f == "2") {
    delete pb;
    InitList(pb);
    int m;
    cout << endl << "|-------------------------------------------------------------------------------------------|" << endl;
    cout << "|                                    请输入pb的项数：                                       |" << endl;
    cout << "|-------------------------------------------------------------------------------------------|" << endl << endl;
    cin >> m;
    while (m == 0 || cin.fail()) {
        cin.clear();
        cin.ignore(10000, '\n');
        cout << endl << "|-----------------------------------输入错误！-----------------------------------|" << endl << endl;
        cin >> m;
    }
    if (!cin.fail()) {
        rewind(stdin);
    }
    CreatePolyn(pb, m);
    system("cls");
    Cls(pa, pb, pc, pd);
}
```

输入 3 到 7，主函数代码如下：

```
else if (f == "3"){
    AddPolyn(pa, pb, pc, pd);
}
else if (f == "4"){
    SubPolyn(pa, pb, pc, pd);
}
else if (f == "5"){
    MultiplyPolyn(pa, pb, pc, pd);
}
else if (f == "6") {
    DerivativeA(pa, pb, pc, pd);
}
else if (f == "7") {
    DerivativeB(pa, pb, pc, pd);
}
```

输入 8，函数会让用户输入 a，b 之间的一个字母和 int 类型的数字 x，随后调用 Value 函数进行计算并返回，输出在屏幕上：

```
else if (f == "8") {
    double x;
    char p;
    cout << endl << "|----------------------------请输入多项式(a,b)以及x值----------------------------|" << endl << endl;
    cin >> p >> x;
    while (cin.fail()||(p!='a'&& p!='b')) {
        cin.clear();
        cin.ignore(10000, '\n');
        cout << endl << "|-----------------------------------输入错误！ -----------------------------------|" << endl << endl;
        cin >> p >> x;
    }
    if (p == 'a') {
        cout << endl << "|-----------------------------------计算结果为-----------------------------------|" << endl << endl << "pa(" << x << ")=" << Value(pa, x) << endl << endl;
        cout << "|-----------------------------------按回车继续-----------------------------------|" << endl;
    }
    else if (p == 'b') {
        cout << endl << "|-----------------------------------计算结果为-----------------------------------|" << endl << endl << "pb(" << x << ")=" << Value(pb, x) << endl << endl;
        cout << "|-----------------------------------按回车继续-----------------------------------|" << endl;
    }
    rewind(stdin);
    system("pause>nul");
    system("cls");
    Cls(pa, pb, pc, pd);
}
```

输入 9，退出循环，送你一颗爱心，另外，输入 0-9 之外的字符，程序会提示 用户重新输入：

```
        }
        else if (f == "9") {
            cout << endl;
            cout << "|---------------------------------谢谢使用！再见！ ---------------------------------|" ;
            break;
        }
        else {
            cout << "|---------------------------------请输入合法功能---------------------------------|" << endl << endl;
            continue;
        }
    }
    double x, y, a;
    for (y = 1.5; y > -1.5; y -= 0.1) {
        for (x = -1.5; x < 1.5; x += 0.05) {
            a = x * x + y * y - 1;
            cout << (a * a * a - x * x * y * y * y <= 0.0 ? "*" : " ");
        }
        system("color Fc");
        cout << endl;
    }
    return 0;
}
```