```
public class Polynomial {
    public static void main(String[] args) {
        Polynomial p1 = new Polynomial();
        p1.append(2.2, 1);
        p1.append(3.3, 2);
        p1.append(4.1, 7);
        System.out.println("p1: " + p1);
        System.out.println("最高项: " + p1.findDegree());
        System.out.println("求导: " + p1.differentiation());

        Polynomial p2 = new Polynomial();
        p2.append(2.2, 5);
        p2.append(3.4, 6);
        p2.append(5.7, 1);
        System.out.println("p2: " + p2);
        System.out.println("最高项: " + p2.findDegree());
        System.out.println("求导: " + p2.differentiation());

        System.out.println("加: " + p1.add(p2));
        System.out.println("减: " + p1.substract(p2));
        System.out.println("乘: " + p1.multiply(p2));
    }

    private Mono first; // 首项

    //添加单项式
    public void append(Mono Mono) {
        if (Mono == null) {
            //为空就什么也不做
        } else if (first == null) {
            first = Mono;
        } else {
            Mono current = first;
            while (current != null) {
                // 如果指数相同，则相加
                if (current.index == Mono.index) {
                    current.coeff += Mono.coeff;
                    break;
                } else if (current.next == null) { // 否则直接扔到最后
                    current.next = Mono;
                    break;
                }
                current = current.next;
            }
        }
    }

    public void append(double c, int i) {
        append(new Mono(c, i));
    }

    //格式化输出此类的信息
    public String toString() {
        StringBuffer sb = new StringBuffer();
        Monomial current = first;

        while (current.next != null) {
            sb.append("(" + String.format("%1$.1f", current.coefficient) + "x^" + current.index + ")
+ ");
            current = current.next;
        }
        sb.append("(" + String.format("%1$.1f", current.coefficient) + "x^" + current.index + ")");
        return sb.toString();
```

```
    }

    public int findDegree() {
        int result = 0;
        Mono current = this.first;
        while (current != null) {
            if(result<current.index){
                result = current.index;
            }
            current = current.next;
        }
        return result;
    }
    // 两个多项式相加
    public Polynomial add(Polynomial p2) {
        Polynomial result = new Polynomial();
        Mono current = this.first;
        while (current != null) {
            result.append(current.coeff, current.index);
            current = current.next;
        }
        current = p2.first;
        while (current != null) {
            result.append(current.coeff, current.index);
            current = current.next;
        }
        return result;
    }
    // 两个多项式相减
    public Polynomial substract(Polynomial p2) {
        Polynomial result = new Polynomial();
        Mono current = this.first;
        while (current != null) {
            result.append(current.coeff, current.index);
            current = current.next;
        }
        current = p2.first;
        while (current != null) {
            result.append(-current.coeff, current.index);
            current = current.next;
        }
        return result;
    }

    public Polynomial multiply(Polynomial p2) {
        Polynomial result = new Polynomial();
        Mono c1 = this.first;
        Mono c2 = p2.first;
        while (c1 != null) {
            while (c2 != null) {
                result.append(c1.coeff * c2.coeff, c1.index
                        + c2.index);
                c2 = c2.next;
            }
            c1 = c1.next;
            c2 = p2.first;
        }
        return result;
    }

    public Polynomial differentiation() {
        Polynomial result = new Polynomial();
        Mono current = this.first;

        while (current != null) {
```

```
            result.append(current.coeff*current.index, current.index-1);
            current = current.next;
        }
        return result;
    }
}

// 单项式
class Mono {
    double coeff; // 系数
    int index; // 指数
    Mono next; // 后继结点

    public Mono() {
    }
    public Mono(double c, int i) {
        this.coeff = c;
        this.index = i;
    }
}
```