資料結構-hw3

張帟淇

41243129

january ,8 2024

內容

解題說明

實作多項式連結串列，並且讓運算子多載內容如下:

```
class Polynomial{
private:
    Node* head;   //第一個節點

public:
    Polynomial();
    Polynomial(const Polynomial& a);
    ~Polynomial();

    Polynomial& operator=(const Polynomial& a);         //等於
    Polynomial operator+(const Polynomial& b) const;    //加法
    Polynomial operator-(const Polynomial& b) const;    //減法
    Polynomial operator*(const Polynomial& b) const;    //乘法
    float Evaluate(float x) const;                      //計算多項式值

    friend istream& operator>>(istream& is, Polynomial& x);       //輸入
    friend ostream& operator<<(ostream& os, const Polynomial& x); //輸出
};
```

Node 結構:

```
struct Node{
    int coef;   //係數
    int exp;    //指數
    Node* link; //下一個節點

    Node(int c=0,int e=0):coef(c),exp(e),link(this){}
};
```

程式實作

等於:

```
//等於
Polynomial& Polynomial::operator=(const Polynomial& a){
    head=new Node();
    head->link=head;
    Node* p=a.head->link;      //從a的頭後面一個開始
    Node* r=head;              //指向新串列的第一個
    //開始複製
    while(p!=a.head){          //把每一項都跑一遍
        Node* newNode=new Node(p->coef,p->exp);
        r->link=newNode;       //把新串列串在一起
        r=newNode;             //再往新串列的下一個指
        p=p->link;             //原串列指向下一個
    }
    r->link=head;              //環狀
    return *this;
}
```

帶入 X 值:

```
// 計算帶入X後的值
float Polynomial::Evaluate(float x) const{
    float result=0;
    Node* p=head->link;                 //從a的頭後面一個開始
    while(p!=head){                     //把每一項都跑一遍
        result+=p->coef*pow(x,p->exp); //係數*指數的X次方
        p=p->link;                      //往下一個跑
    }
    return result;
}
```

加法:

```
//加法a+b
Polynomial Polynomial::operator+(const Polynomial& b) const{
    Polynomial result;
    Node *p=head->link,*q=b.head->link;
    Node *r=result.head;

    while(p!=head && q!=b.head){
        //這邊會先指數做比大小 大的就會先被放到串列裡面
        if (p->exp>q->exp){
            r->link=new Node(p->coef,p->exp);
            r=r->link;
            p=p->link;
        }else if(p->exp<q->exp){
            r->link=new Node(q->coef,q->exp);
            r=r->link;
            q=q->link;
        //如果相同的話就會把係數相加 然後加入串列
        }else{
            int sumCoef=p->coef+q->coef;
            if(sumCoef!=0){
                r->link=new Node(sumCoef,p->exp);
                r=r->link;
            }
            p=p->link;
            q=q->link;
        }
    }

    //如果有剩下的項目會再被加進去
    while(p!=head){
        r->link=new Node(p->coef,p->exp);
        r=r->link;
        p=p->link;
    }
    //如果有剩下的項目會再被加進去
    while(q!=b.head){
        r->link=new Node(q->coef,q->exp);
        r=r->link;
        q=q->link;
    }
    r->link=result.head;  //形成環狀
    return result;
}
```

乘法:

```
//乘法
Polynomial Polynomial::operator*(const Polynomial& b) const{
    Polynomial result;
    Node* p=head->link;
    //把每一項都相乘
    while(p!=head){              //把每一項都跑一遍
        Polynomial temp;         //儲存相乘後結果
        Node* q=b.head->link;
        Node* r=temp.head;
        while(q!=b.head){
            r->link=new Node(p->coef*q->coef,p->exp+q->exp);
            r=r->link;
            q=q->link;
        }
        r->link=temp.head;       //形成環狀
        result=result+temp;      //使用加法多載把同指數的部分相加
        p=p->link;
    }
    return result;
}
```

運算子多載 輸入:

```
//輸入
istream& operator>>(istream& is,Polynomial& x){
    int n;
    is>>n;                      //讀取有幾項
    Node* r=x.head;
    for(int i=0;i<n;++i){
        int coef,exp;
        is>>coef>>exp;          //輸入係數、指數
        Node* newNode=new Node(coef,exp);     //加入到新串列
        r->link=newNode;                      //把新節點連結到串列裡面
        r=newNode;                            //把R指向最新新的節點
    }
    r->link=x.head;                           //形成環狀
    return is;
}
```

運算子多載 輸出:

```
//輸出
ostream& operator<<(ostream& os,const Polynomial& x){
    Node* p=x.head->link;
    bool first=true;         //檢查是不是第一個
    while(p!=x.head){        //把整個串列跑過一遍            //a為係數  b為指數
        if(!first && p->coef>0) os<<"+";      //如果不是第一個且為正數的話就會加正號
        if(p->exp == 0) {                     //如果指數為0就直接把係數印出
            os<<p->coef;
        }else if(p->coef!=1 && p->coef!=-1 && p->exp>1){     //係數不為正負1且指數大於1就依ax^b的格式印出
            os<<p->coef<<"x^"<<p->exp;
        }else if(p->coef==-1 && p->exp>1){    //係數為正負1且指數大於1就依x^b或是-x^b的格式印出
            os<<"-x^"<<p->exp;
        }else if(p->coef==1 && p->exp>1){
            os<<"x^"<<p->exp;
        }else if(p->coef==1 && p->exp==1){    //係數為正負1且指數等於1就依x或是-x的格式印出
            os<<"x";
        }else if(p->coef==-1&& p->exp==1){
            os<<"-x";
        }else if(p->exp=1){                   //指數為1就直接依ax的格式印出
            os<<p->coef<<"x";
        }
        first=false;
        p=p->link;
    }
    return os;
}
```

效能分析

時間複雜度

輸入:O(n)

輸出:O(n)

加法:O(n1+n2)　n1:poly1 的長度　n2:poly2 的長度

減法: O(n1+n2)

乘法: O(n1*n2)

Evaluete:O(n)

空間複雜度

輸入:O(n)

輸出:O(1)

加法:O(n1+n2)　n1:poly1 的長度　n2:poly2 的長度

減法:O(n1+n2)

乘法:O(n1*n2)

Evaluete:O(1)

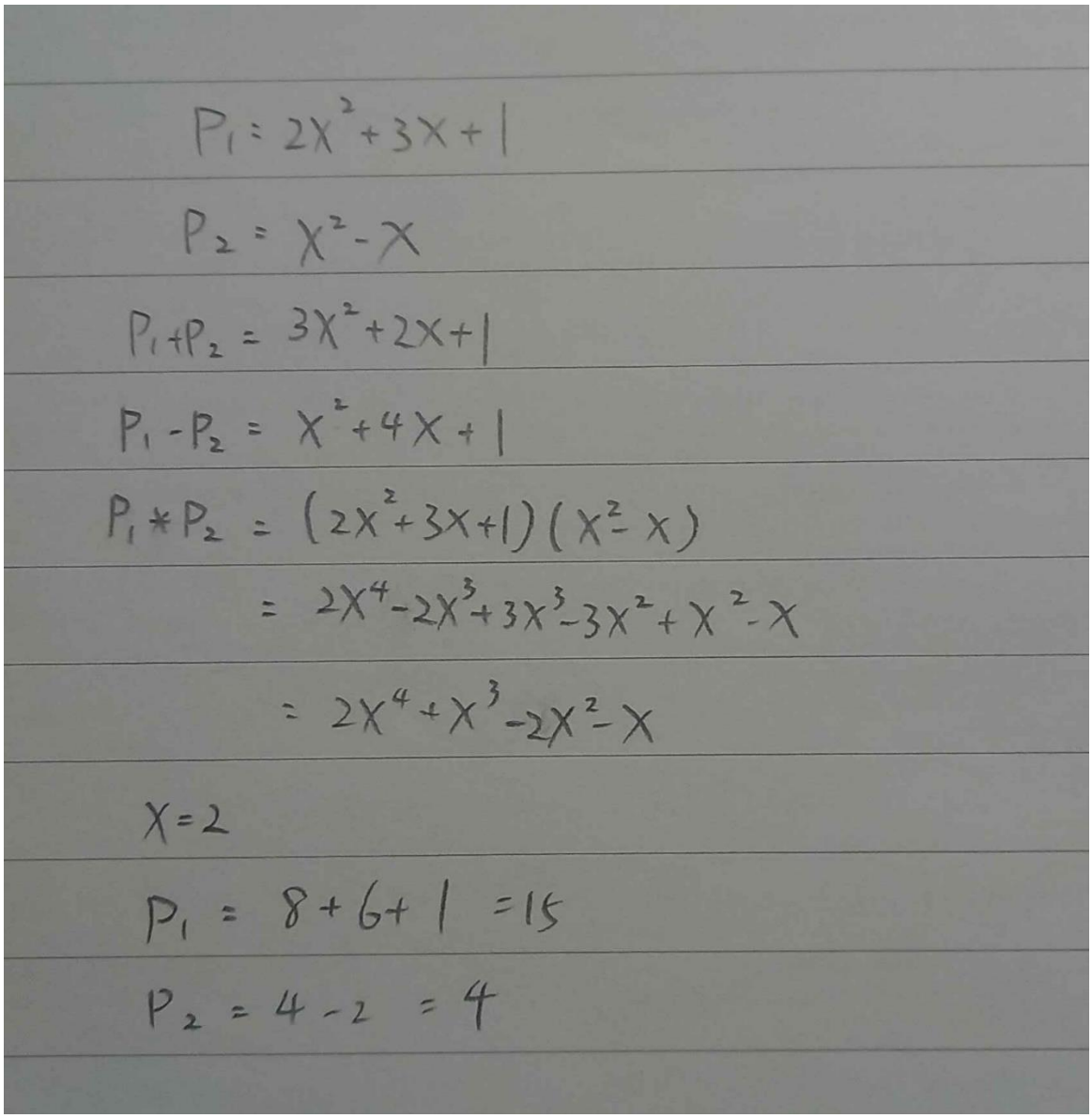
測試與驗證

主程式:

```
int main() {
    Polynomial p1,p2;
    //輸入多項式
    cout<<"請輸入多項式(格式為:n(項數) cof1(係數1) exp1(指數1) cof2(係數2) exp2(指數2).....)\np1:";
    cin>>p1;
    cout<<"p2:";
    cin>>p2;

    Polynomial sum=p1+p2;       //相加
    Polynomial diff=p1-p2;      //相減
    Polynomial prod=p1*p2;      //相乘

    //印出結果
    cout<<"p1: "<<p1<<endl;
    cout<<"p2: "<<p2<<endl;
    cout<<"p1+p2: "<<sum<<endl;
    cout<<"p1-p2: "<<diff<<endl;
    cout<<"p1*p2: "<<prod<<endl;

    //輸入x值
    float x;
    cout<<"請輸入要帶入的x值:";
    cin>>x;
    cout<<"p1("<<x<<")="<<p1.Evaluate(x)<<endl;
    cout<<"p2("<<x<<")="<<p2.Evaluate(x)<<endl;
    return 0;
}
```

```
請輸入多項式(格式為:n(項數) cof1(係數1) exp1(指數1) cof2(係數2) exp2(指數2).....)
p1:3 2 2 3 1 1 0
p2:2 1 2 -1 1
p1: 2x^2+3x+1
p2: x^2-x
p1+p2: 3x^2+2x+1
p1-p2: x^2+4x+1
p1*p2: 2x^4+x^3-2x^2-x
請輸入要帶入的X值:2
p1(2)=15
p2(2)=2
```

驗證:


$P_1 = 2X^2 + 3X + 1$
$P_2 = X^2 - X$
$P_1 + P_2 = 3X^2 + 2X + 1$
$P_1 - P_2 = X^2 + 4X + 1$
$P_1 * P_2 = (2X^2 + 3X + 1)(X^2 - X)$
$= 2X^4 - 2X^3 + 3X^3 - 3X^2 + X^2 - X$
$= 2X^4 + X^3 - 2X^2 - X$
$X = 2$
$P_1 = 8 + 6 + 1 = 15$
$P_2 = 4 - 2 = 4$

申論及開發報告

加法

主要是看指數是否相等或是哪邊比較大，大的會先新增到串列裡，如果一樣就相加再加入串列。

減法

跟加法的做法其實大同小異，只是要注意 p1 - p2 指數比大小的時候如果 p2 比較大的話，加入串列時係數就要加一個負號。

乘法

把每一項都跑過一遍，相乘之後再把他跟 reslut 相加這樣可以把同指數的數字加起來，直到 p1 跑完一圈。

Evaluete

把每一項帶入 x 的值算出來，然後加總。

運算子多載 輸入

輸入多項式後先看有幾項，然後再依序加入到新串列裡面，把 r->link 指向 newnode 再把 r 也指向 newnode 之後就可以一直往後添加資料。

運算子多載 輸出

把輸出讀取進來後，會經過很多判斷式，基本上就是在看指數是否為 0 如果式的話就直接印出係數，再來就是判斷係數是否為正負 1，有三種情況:

1. 不為正負 1 且指數大於 1 就依 ax^b 的格式印出
2. 為正負 1 且指數大於 1 就依 x^b 或是-x^b 的格式印出
3. 為正負 1 且指數等於 1 就依 x 或是-x 的格式印出

最後指數為 1 就直接依 ax 的格式印出

這隻程式其實只要把串列的邏輯搞明白，就蠻容易理解了，可以把圖畫出來幫助思考，但最後再輸出的那一部份邏輯整個打結導致最後花太多時間，找了一陣子才發現問題在哪裡。