DS HW3

M56121041 黃啟維

程式解說

print_add_minus（ char *filemname, int arra_len, int arrb_len)

我將原本main裡面的print和fuantion call它寫一個fucntion包刮了以下幾個功能

1. file proccessing (將file裡的兩行讀成兩個arr）但要給兩個arr的int各數
2. construct node list a和b
3. 呼叫plus和minus函數做兩poly的加減法 並將結果印出
4. 解決memory leak問題

```
#define MAXREAD 10
void print_add_minus(char *filemname, int arra_len, int arrb_len){
    FILE *in;
    in = fopen(filemname, "r");//只讀就好
    if (!in) {
        perror("Error opening file");
        return;
    }
    int reada[MAXREAD] = {0},readb[MAXREAD] = {0};
    for (int i = 0; i < arra_len; i++) {
        fscanf(in, "%d", &reada[i]);
    }

    // 讀取第二行到 arr2
    for (int i = 0; i < arrb_len; i++) {
        fscanf(in, "%d", &readb[i]);
    }
    fclose(in);
    Node *A = construct(reada, arra_len);
    Node *B = construct(readb, arrb_len);
    printf("%s\n", filemname);
    printf("\n");

    printf("A:");
    print_poly(A);

    printf("B:");
    print_poly(B);
    Node *sum = plus(A, B);
    printf("A+B:");
    print_poly(sum);
    Node *diff = minus(A, B);
    printf("A-B:");
    print_poly(diff);
    printf("\n");

    free_poly(A);
    free_poly(B);
    free_poly(sum);
    free_poly(diff);
}
```

所以main舊址會有檔名和arr int數

```
int main(){
    // Node declare

    // file read and construct two given polynomial

    // A+B and A-B

    //For print
    print_add_minus("test1.txt", 6, 8);
    print_add_minus("test2.txt", 8, 8);
    print_add_minus("test3.txt", 8, 8);
    print_add_minus("test4.txt", 8,8);
    print_add_minus("test5.txt", 6,6);
    return 0;
}
```

Node* construct(int arr[], int listsize)

我的construct是以ptp去實做 先創造一個*head並用 ** curr去指向它
先用** curr去尋找新的node的expo在list中的大小位置
找到後插入這個新的node 並且再次將** curr指向head
即可達到降冪排列 並回傳head

```
Node* construct(int arr[], int listsize ){
    if(listsize < 2 && arr == NULL)return NULL;
    Node *head = new_node(arr[0], arr[1]);
    for(int i = 2; i < listsize ; i += 2){
        int coef = arr[i];
        int expo = arr[i+1];
        //創件一個ptp
        Node **curr = &head;
        //找新的node的linklist中的位置 *curr要存在且下一個expo不能比新的大
        while (*curr != NULL && (*curr)->expo > expo){
            //把**curr一到下一個node
            curr = &((*curr)->next);
        }
        //如果下一個和新的相同次數的加在一起
        if (*curr != NULL && (*curr)->expo == expo){
            (*curr)->coef += coef;
            //如果下一個項相加後係數是零就把它去掉
            if ((*curr)->coef == 0){
                Node *temp = *curr;
                *curr = (*curr)->next;
                free(temp);
            }
        }
        else {
            //在下一個創一個新的node
            Node *temp = new_node(coef, expo);
            temp ->next = *curr;
            *curr = temp;
        }

    }
    return head;
}
```

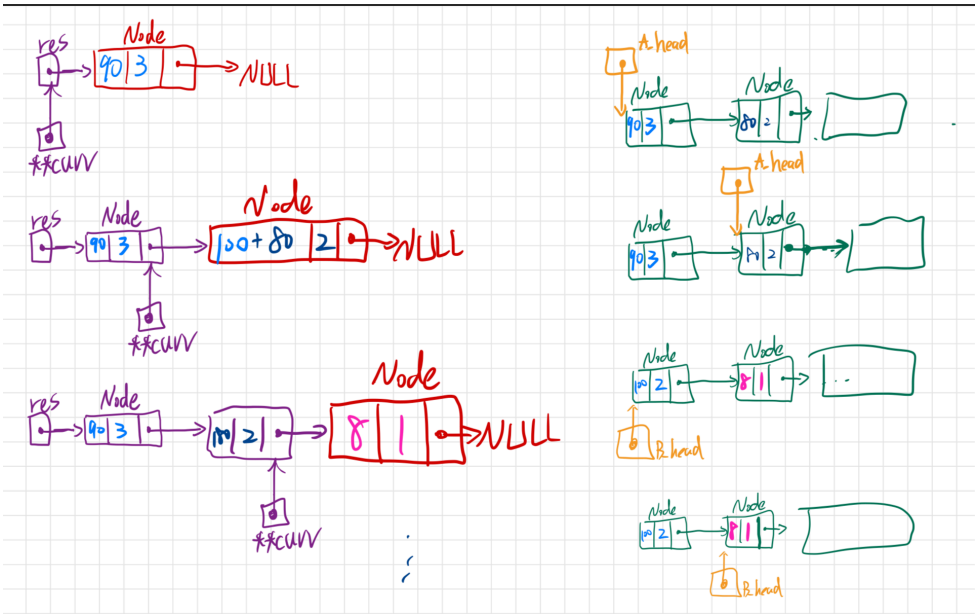
Node* new_node

要malloc一個新的node

```
Node* new_node(int coef, int expo){
    Node *node = malloc(sizeof(Node));
    node->coef = coef;
    node->expo = expo;
    node->next = NULL;
    return node;
}
```

Node* plus(Node * A_head , Node* B_head)

比較兩個head的次數把大的複製進res list中
若ＡＢ次數相等則把兩個的係數相加放進res list中
直到ＡＢ其中一個ｌｉｓｔ結束後把另一個剩餘的複製進ｒｅｓ　ｌｉｓｔ中
以下為簡單圖示說明




```
Node* plus(Node *A_head , Node* B_head){
    //創一個reslist來接
    Node *res = NULL, **curr = &res;
    while(A_head && B_head){
        if(A_head->expo == B_head->expo){
            //如果相加係數不等於0創新的node
            if((A_head->coef + B_head->coef) != 0){
                *curr = new_node((A_head->coef + B_head->coef), A_head->expo);
                curr = &(*curr)->next;
            }
            A_head = A_head->next;
            B_head = B_head->next;

        }
        else if(A_head->expo > B_head->expo){
            *curr = new_node(A_head->coef, A_head->expo);
            curr = &(*curr)->next;
            A_head = A_head->next;
        }
        else {
            *curr = new_node(B_head->coef, B_head->expo);
            curr = &(*curr)->next;
            B_head = B_head->next;
        }
    }
    //A_head 或 B_head有一個不存在時 代表一個讀完了 把另一個的後面全部複製進去res就好
    while((!A_head) && B_head){
        *curr = new_node(B_head->coef, B_head->expo);
        curr = &(*curr)->next;
        B_head = B_head->next;
    }
    while((!B_head) && A_head){
        *curr = new_node(A_head->coef, A_head->expo);
        curr = &(*curr)->next;
        A_head = A_head->next;
    }
    return res;
}
```

Node* minus(Node * A_head , Node* B_head)

大致的邏輯概念和plus差不多 僅更動B的coef正負號

```
Node* minus(Node *A_head , Node* B_head){
    Node *res = NULL, **curr = &res;
    while(A_head && B_head){
        if(A_head->expo == B_head->expo){
            //如果相加係數不等於0創新的node
            if((A_head->coef - B_head->coef) != 0){
                *curr = new_node((A_head->coef - B_head->coef), A_head->expo);
                curr = &(*curr)->next;
            }
            A_head = A_head->next;
            B_head = B_head->next;
        }
        else if(A_head->expo > B_head->expo){
            *curr = new_node(A_head->coef, A_head->expo);
            curr = &(*curr)->next;
            A_head = A_head->next;
        }
        else {
            *curr = new_node(-(B_head->coef), B_head->expo);
            curr = &(*curr)->next;
            B_head = B_head->next;
        }
    }
    //A_head 或 B_head有一個不存在時 代表一個讀完了 把令一個的後面全部複製進去res就好
    while(B_head){
        *curr = new_node(-(B_head->coef), B_head->expo);
        curr = &(*curr)->next;
        B_head = B_head->next;
    }
    while(A_head){
        *curr = new_node(A_head->coef, A_head->expo);
        curr = &(*curr)->next;
        A_head = A_head->next;
    }
    return res;
}
```

結果截圖

以下維test1~test5的輸出結果

```
test1.txt

A:
(3)X^90 + (2)X^5 + (10)X^0
B:
(5)X^40 + (3)X^20 + (6)X^3 + (10)X^0
A+B:
(3)X^90 + (5)X^40 + (3)X^20 + (2)X^5 + (6)X^3 + (20)X^0
A-B:
(3)X^90 + (-5)X^40 + (-3)X^20 + (2)X^5 + (-6)X^3

test2.txt

A:
(10)X^100 + (5)X^50 + (3)X^1 + (99)X^0
B:
(10)X^100 + (5)X^50 + (3)X^1 + (99)X^0
A+B:
(20)X^100 + (10)X^50 + (6)X^1 + (198)X^0
A-B:
0

test3.txt

A:
(9)X^10 + (89)X^4 + (85)X^2 + (10)X^0
B:
(10)X^11 + (10)X^6 + (89)X^4 + (-5)X^2
A+B:
(10)X^11 + (9)X^10 + (10)X^6 + (178)X^4 + (80)X^2 + (10)X^0
A-B:
(-10)X^11 + (9)X^10 + (-10)X^6 + (90)X^2 + (10)X^0

test4.txt

A:
(99)X^10 + (98)X^5 + (97)X^1 + (96)X^0
B:
(98)X^10 + (97)X^5 + (96)X^1 + (95)X^0
A+B:
(197)X^10 + (195)X^5 + (193)X^1 + (191)X^0
A-B:
(1)X^10 + (1)X^5 + (1)X^1 + (1)X^0

test5.txt

A:
(-1)X^50 + (-3)X^25 + (-6)X^0
B:
(-10)X^51 + (-3)X^25 + (-14)X^0
A+B:
(-10)X^51 + (-1)X^50 + (-6)X^25 + (-20)X^0
A-B:
(10)X^51 + (-1)X^50 + (8)X^0
```