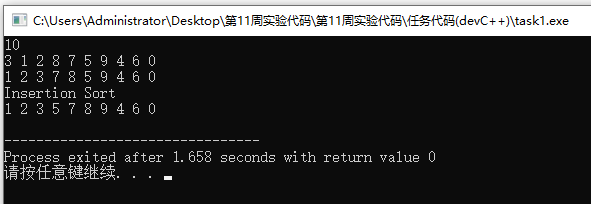
Task1



Task2





class Solution {

public:

    vector<int> pancakeSort(vector<int>& arr) {

        vector<int> result;

        int len=arr.size();

        while(len){

            int k=0;

            for(int i=1;i<len;i++){

                if(arr[i]>arr[k]) k=i;

            }

            if(k!=len-1){

                for(int j=0;j<=k/2;j++){

                    swap(arr[j],arr[k-j]);

                }

                result.push\_back(k+1);

                for(int j=0;j<len/2;j++){

                    swap(arr[j],arr[len-j-1]);

                }

                result.push\_back(len);

            }

            len--;

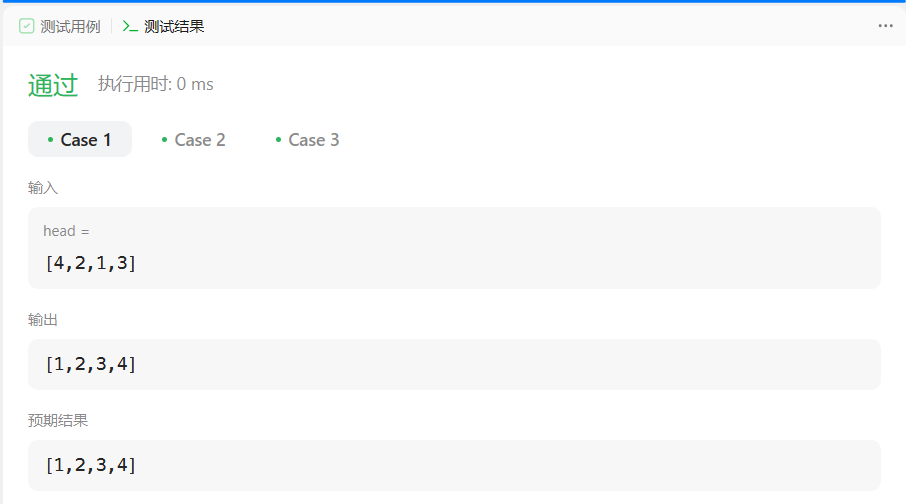
        }

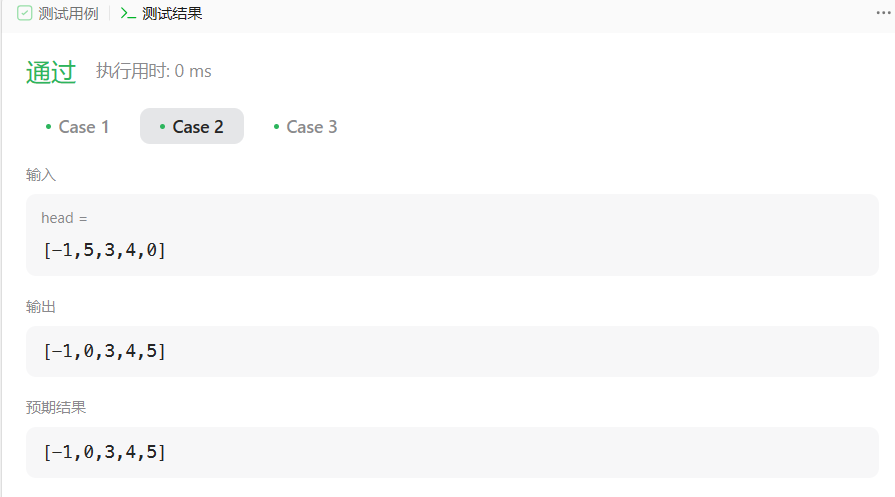
        return result;

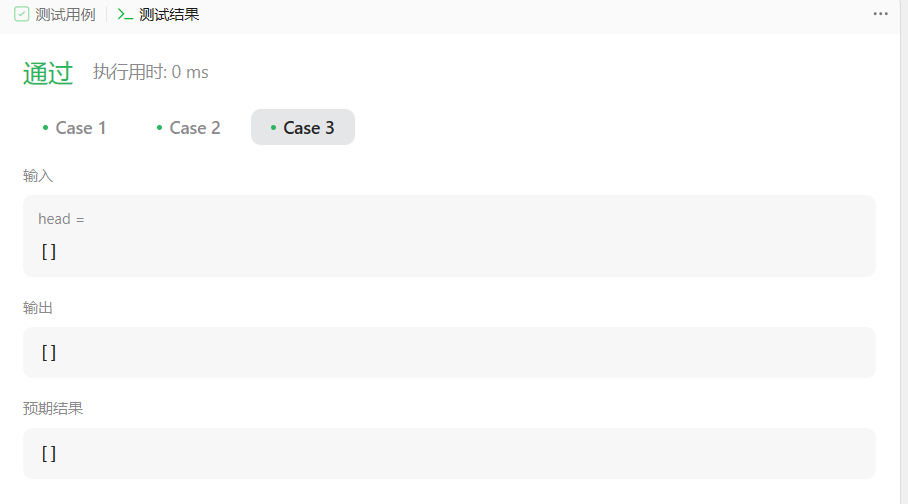
    }

};

Task3







/\*\*

 \* Definition for singly-linked list.

 \* struct ListNode {

 \*     int val;

 \*     ListNode \*next;

 \*     ListNode() : val(0), next(nullptr) {}

 \*     ListNode(int x) : val(x), next(nullptr) {}

 \*     ListNode(int x, ListNode \*next) : val(x), next(next) {}

 \* };

 \*/

class Solution {

public:

    ListNode\* sortList(ListNode\* head) {

        if(!head) return head;

        ListNode\* p=head;

        while(p){

            int tmp=p->val;

            ListNode\* q=p->next;

            ListNode\* min\_val=p;

                while(q){

                    if(q->val<min\_val->val){

                        min\_val=q;

                    }

                    q=q->next;

                }

                if(min\_val){

                    p->val=min\_val->val;

                    min\_val->val=tmp;

                }

            p=p->next;

        }

        return head;

    }

};

Task4



