10주차 실습

Today’s Topic

- Structure
- Algorithm

Hw1. Complex number calculator

Description

Prof. Lee asked to you to make calculator for complex number.

Implement 4 blank functions to complete professor's request using structure.

Skeleton code

```
#include <stdio.h>
struct complexNum {
        int real;
        int img;
};
typedef void (*calcFuncPtr)(struct complexNum, struct complexNum, int*, int*);

struct complexNum createComplexNum(int real, int img)
{
    // fill here
}

void plus (struct complexNum input1, struct complexNum input2, int* output_real,
int* output_img){
    // fill here
}

void minus (struct complexNum input1, struct complexNum input2, int* output_real,
int* output_img){
    // fill here
}

void mult (struct complexNum input1, struct complexNum input2, int* output_real,
int* output_img){
    // fill here
}

void calc (struct complexNum input1, struct complexNum input2, int* r3, int* i3,
calcFuncPtr func){
    // fill here
}

int main(){
    calcFuncPtr f = NULL;
    int r1, r2, r3, i1, i2, i3;
    struct complexNum newCom1, newCom2;
    char op;

    scanf("%d %d %c %d %d", &r1, &i1, &op, &r2, &i2);
    switch(op){
        case '+': f = plus; break;
        case '-': f = minus; break;
        case '*': f = mult; break;
    }

    newCom1 = createComplexNum(n1[0], n1[1]);
    newCom2 = createComplexNum(n2[0], n2[1]);
    calc(newCom1, newCom2, &r3, &i3, f);
    printf("%d %d\n", r3, i3);
    return 0;
}
```

Input	Output
2 3 + 4 5	6 8
5 3 – 6 2	-1 1
-2 3 * 5 -1	-7 17

Hw2. Maximum subarray problem

Description

- Mr.O ended in bronze tier this season again. Never to make this mistake again, Mr.O is trying to plan the perfect schedule for gaming. He has records of his rating changes. From this records, find the contiguous subarray with the largest sum.(array's length >= 1)
- Use the naïve solution (with double loop statement traversing starting index and ending index) for smaller than 1000 inputs.
- Use good solution (time complexity O(n)) for big inputs

Hw2. Maximum subarray problem

Input

-First line : the number of inputs, lower than 100000
-Second line : inputs, with absolute values lower than or equal to 1000

Output

-First line : “big” for more than or equal to 1000 inputs, “small” for else cases.
-Second line : maximum subarray’s starting index, ending index (index starts at 0)
-If there’s more than one maximum subarray, print the one with miminum starting index.
-Third line : largest sum

Input	Output
5 1 2 3 -4 5	small 0,4 7
1 1	small 0,0 1
3 -100 -1 -100	small 1,1 -2
5 1 2 -100 1 2	small 0,1 3
1000 1 2 3 4 5 6 7 …500 -501 -502 -503 … -1000	big 0,499 125250

Hw3. Minimum Distance

Description

There are N points on 2D plane. Find the Closest point to each point.

Distance (p1, p2) = sqrt((x_2 – x_1)^2 + (y_2 – y_1)^2) [Not Manhattan distance]

Hint : you can use **Distance(p1, p2)^2**

Hw3. Minimum Distance

Input

On the first line, number of points N is given (2 <= N < 400)

On the following N lines, each points' x and y coordinates are given. The point number starts from 0 and is based on the input order.

(|x|, |y| <= 20000, all coordinates are integer)

Output

Output the closest points based on the input order.

Constraint

Use "structure" for the position of point

Input	Output
4 0 0 0 1 1 2 1 -1	1 0 1 0
2 -1 -1 -1 -1	1 0

Hw4

Description

You are given N rectangles.

Define S_i : $Area\ of\ intersection\ of\ rectangle\ 1\ ,2\ ,\ldots,i$

Your task is to find $S_1\ ,S_2\ ,\ldots\ ,S_N$.

Hw4

Input

On the first line, number of rectangles N is given.

On the (i+1)-th line, information of i-th rectangle are given. Four integers $\mathrm{x1}, \mathrm{y1}, \mathrm{x2}, \mathrm{y2}$ are given. $(\mathrm{x1}, \mathrm{y1})$ is left-bottom point of the rectangle, and $(\mathrm{x2}, \mathrm{y2})$ is is right-top point of the rectangle.

Output

Print N lines. On the i-th line, print one integer S_i

Constraints

N <= 100,000

-1,000,000,000 <= $\mathrm{x1}, \mathrm{y1}, \mathrm{x2}, \mathrm{y2}$ <= 1,000,000,000

Input	Output
3 1 2 4 4 2 3 5 7 3 1 6 5	6 2 1
2 -100 -100 100 100 -1 -1 1 1	40000 4
4 -1 -1 1 1 -2 -2 2 2 -3 -3 3 3 -4 -4 4 4	4 4 4 4

공지사항

- 제출 : ~11/12 오후 11:59 까지
- 질문사항 : snupp2017@gmail.com