CSC11F: Advanced Data Structures and Algorithms

May 25, 2020 *Final Assignment

Yutaka Watanobe (142-B, yutaka@u-aizu.ac.jp)

1 Final Assignments

Problems

- **Problem A** DSL_3_D: Sliding Minimum Elements
- **Problem B** GRL_5_D: Range Query on a Tree
- **Problem C** DSL_2_C: Range Search (kD Tree)

Note that you can use the advanced data structures which you have obtained from this course to solve Problem A and Problem B.

Problem C is optional. If your research is related to data mining, database, computational geometry and other advanced algorithms, you should try to solve it by kD Tree (which was not provided in this lecture, but I hope you study by yourself).

Place

https://onlinejudge.u-aizu.ac.jp/beta/room.html#CSC11F_2020_Week_06

Duration

2019/05/25 13:20 - the end of Q1

Report

Please choose at least one problem and write a report which includes the following items for each problem:

- Breaf description or summary of the problem
- Constraints and critical cases of input for the problem
- The main data structure employed to implement the algorithm
- Description of your algorithm to solve the problem
- Time and space complexity of your algorithm
- Source code which has solved the problem

Submit your report to Moodle.

Problem A: Sliding Minimum Element

For a given array $a_1, a_2, a_3, ..., a_N$ of N elements and an integer L, find the minimum of each possible sub-arrays with size L and print them from the beginning. For example, for an array $\{1, 7, 7, 4, 8, 1, 6\}$ and $L = 3$, the possible sub-arrays with size $L = 3$ includes $\{1, 7, 7\}$, $\{7, 7, 4\}$, $\{7, 4, 8\}$, $\{4, 8, 1\}$, $\{8, 1, 6\}$ and the minimum of each sub-array is 1, 4, 4, 1, 1 respectively.

Constraints

- $1 \leq N \leq 10^6$
- $1 \leq L \leq 10^6$
- $1 \leq a_i \leq 10^9$
- $L \leq N$

Input

The input is given in the following format.

N L
a_1 a_2 ... a_N

Output

Print a sequence of the minimum in a line. Print a space character between adjacent elements.

Sample Input 1

```
7 3
1 7 7 4 8 1 6
```

Sample Output 1

```
1 4 4 1 1
```

Problem B: Range Query on a Tree

Write a program which manipulates a weighted rooted tree T with the following operations:

- $add(v, w)$: add w to the edge which connects node v and its parent
- $getSum(u)$: report the sum of weights of all edges from the root to node u

The given tree T consists of n nodes and every node has a unique ID from 0 to $n-1$ respectively where ID of the root is 0. Note that all weights are initialized to zero.

Input

The input is given in the following format.

n
$node_0$
$node_1$
$node_2$
:
$node_{n-1}$
q
$query_1$
$query_2$
:
$query_q$

The first line of the input includes an integer n, the number of nodes in the tree.

In the next n lines, the information of node i is given in the following format:

k_i c_1 c_2 ... c_k

k_i is the number of children of node i, and c_1 c_2 ... c_{k_i} are node IDs of 1st, ... kth child of node i.

In the next line, the number of queries q is given. In the next q lines, ith query is given in the following format:

0 v w

or

1 u

The first integer represents the type of queries. '0' denotes $add(v, w)$ and '1' denotes $getSum(u)$.

Constraints

- All the inputs are given in integers
- $2 \le n \le 100000$
- $c_j < c_{j+1}$ $(1 \le j \le k-1)$
- $2 \le q \le 200000$
- $1 \le u, v \le n-1$
- $1 \le w \le 10000$

Output

For each $getSum$ query, print the sum in a line.

Sample Input 1

```
6
2 1 2
2 3 5
0
0
0
1 4
7
1 1
0 3 10
1 2
0 4 20
1 3
0 5 40
1 4
```

Sample Output 1

```
0
0
10
60
```

Sample Input 2

```
4
1 1
1 2
1 3
0
6
0 3 1000
0 2 1000
0 1 1000
1 1
1 2
1 3
```

Sample Output 2

```
1000
2000
3000
```

Sample Input 3

```
2
1 1
0
4
0 1 1
1 1
0 1 1
1 1
```

Sample Output 3

```
1
2
```

Problem C: Range Search (kD Tree)

The range search problem consists of a set of attributed records S to determine which records from S intersect with a given range.

For n points on a plane, report a set of points which are within in a given range. Note that you do not need to consider insert and delete operations for the set.

Input

n
$x_0\ y_0$
$x_1\ y_1$
:
$x_{n-1}\ y_{n-1}$
q
$sx_0\ tx_0\ sy_0\ ty_0$
$sx_1\ tx_1\ sy_1\ ty_1$
:
$sx_{q-1}\ tx_{q-1}\ sy_{q-1}\ ty_{q-1}$

The first integer n is the number of points. In the following n lines, the coordinate of the i-th point is given by two integers x_i and y_i.

The next integer q is the number of queries. In the following q lines, each query is given by four integers, sx_i, tx_i, sy_i, ty_i.

Output

For each query, report IDs of points such that $sx_i \le x \le tx_i$ and $sy_i \le y \le ty_i$. The IDs should be reported in ascending order. Print an ID in a line, and print a blank line at the end of output for the each query.

Constraints

- $0 \le n \le 500,000$
- $0 \le q \le 20,000$
- $-1,000,000,000 \le x, y, sx, tx, sy, ty \le 1,000,000,000$
- $sx \le tx$
- $sy \le ty$
- For each query, the number of points which are within the range is less than or equal to 100.

Sample Input 1

```
6
2 1
2 2
4 2
6 2
3 3
5 4
2
2 4 0 4
4 10 2 5
```

Sample Output 1

```
0
1
2
4

2
3
5
```