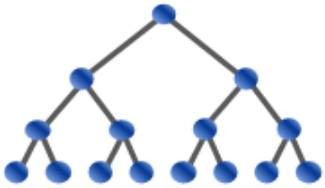

USACO 2019 US Open Contest, Platinum
Problem 1. Tree Boxes

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Farmer John is planning to build N ($1 \le N \le 10^5$) farms that will be connected by $N-1$ roads, forming a tree. Typically, whenever one of his farms is having an issue he is not told the specific farm that is having an issue. Instead, he is told that one of the farms along the path from some farm A to another farm B is having an issue. This is often confusing for Farmer John, as he usually drives offroad tractors and isn't familiar with the road system.

Farmer John considers the location of a farm to be a 2D point. He would prefer to be told that there is a problem in one of the farms in a specified axis-aligned rectangular box. This way Farmer John can decide for himself how to navigate between the farms. Bessie told him that this is a little too ambitious, so he will be satisfied if he is notified with at most two axis-aligned rectangular boxes whose intersection (of farms) is empty and whose union is exactly the farms along the path from A to B. You must help Farmer John determine where he should build his farms such that this condition is satisfied.

This is an interactive problem, you will not be using standard (or file) I/O. **Solutions that use standard (or file) I/O will be disqualified.** However, you ARE ALLOWED to use global and static variables. You must implement the following functions to help Farmer John:

- **void addRoad(int A, int B)**: processes a road between farms A and B ($0 \le A, B \le N-1$).
- **void buildFarms()**: determines where Farmer John should build all his farms.
- **void notifyFJ(int A, int B)**: notifies Farmer John with either one or two boxes that satisfy the aforementioned conditions.

Your implementation of the above functions will be able to call the functions given below. You may assume that `notifyFJ` will be called Q times ($1 \le Q \le 10^5$).

- **int getN()**: gets the value of N.
- **int getQ()**: gets the value of Q.
- **void setFarmLocation(int ID, int X, int Y)**: determines that Farmer John should build farm ID ($0 \le ID \le N-1$) at location (X, Y) where $(1 \le X, Y \le N)$. Should only be called from `buildFarms`.
- **void addBox(int X1, int Y1, int X2, int Y2)**: adds a box to notify Farmer John where $(1 \le X1 \le X2 \le N)$ and $(1 \le Y1 \le Y2 \le N)$. Should only be called from `notifyFJ`.

The interactive protocol works as follows. First, `addRoad` will be called $N-1$ times, to inform your program of the road system. Then, `buildFarms` will be called and you must determine where Farmer John should build each farm and call `setFarmLocation` for every farm accordingly. Finally, there will be Q calls to `notifyFJ` where you must make either one or two calls to `addBox` to notify Farmer John.

It is guaranteed there is always a valid way to notify Farmer John using either one or two boxes. The memory limit for this problem is set to 512MB, above the usual 256MB limit.

For a C++ solution, use this template:

```
#include "grader.h"

void addRoad(int a, int b){
	// Fill in code here
}

void buildFarms(){
	// Fill in code here
}


void notifyFJ(int a, int b){
	// Fill in code here
}
```

For a Java solution, use this template:

```
import java.io.IOException;
// If you find it necessary, you may import other standard libraries here.
public class boxes extends Grader {

        // Copy this exactly:

@Override
        public static void main(String args[]) throws IOException { new boxes().run(); }


@Override
        public void addRoad(int a, int b) {
      // Fill in code here
        }

@Override
        public void buildFarms(){
      // Fill in code here
          }

@Override
        public void notifyFJ(int a, int b){
      // Fill in code here
        }
}
```

Sample Interaction

Grader calls `addRoad(0,1)`

Grader calls `addRoad(1,2)`

Grader calls `buildFarms()`

Solution calls `setFarmLocation(0,1,1)`

Solution calls `setFarmLocation(1,1,2)`

Solution calls `setFarmLocation(2,2,2)`

Solution ends `buildFarms()`

Grader calls `notifyFJ(0,0)`

Solution calls `addBox(1,1,1,1)`

Solution ends `notifyFJ(0,0)`

Grader calls `notifyFJ(0,2)`

Solution calls `addBox(1,1,1,2)`

Solution calls `addBox(2,2,2,2)`

Solution ends `notifyFJ(0,2)`

Grader terminates, and solution passes test-case

(Note: if you do not pass the first test case, the grader will indicate this as usual. However, note that the short sample interaction above does not correspond to the first test case or any other).

Problem credits: Spencer Compton

Contest has ended. No further submissions allowed.