649. Dota2 Senate

In the world of Dota2, there are two parties: the `Radiant` and the `Dire`.

The Dota2 senate consists of senators coming from two parties. Now the senate wants to make a decision about a change in the Dota2 game. The voting for this change is a round-based procedure. In each round, each senator can exercise `one` of the two rights:

1. `Ban one senator's right`:
   A senator can make another senator lose **all his rights** in this and all the following rounds.
2. `Announce the victory`:
   If this senator found the senators who still have rights to vote are all from **the same party**, he can announce the victory and make the decision about the change in the game.

Given a string representing each senator's party belonging. The character 'R' and 'D' represent the `Radiant` party and the `Dire`party respectively. Then if there are `n` senators, the size of the given string will be `n`.

The round-based procedure starts from the first senator to the last senator in the given order. This procedure will last until the end of voting. All the senators who have lost their rights will be skipped during the procedure.

Suppose every senator is smart enough and will play the best strategy for his own party, you need to predict which party will finally announce the victory and make the change in the Dota2 game. The output should be `Radiant` or `Dire`.

Example 1:

```
Input: "RD"

Output: "Radiant"

Explanation: The first senator comes from Radiant and he can just ban the ne
xt senator's right in the round 1.
And the second senator can't exercise any rights any more since his right ha
```

Example 2:

Input: "RDD"

Output: "Dire"

Explanation:

The first senator comes from Radiant and he can just ban the next senator's
right in the round 1.
And the second senator can't exercise any rights anymore since his right has
 been banned.
And the third senator comes from Dire and he can ban the first senator's rig
ht in the round 1.
And in the round 2, the third senator can just announce the victory since he
 is the only guy in the senate who can vote.

Note:

1. The length of the given string will in the range [1, 10,000].

This is obliviously a greedy algorithm problem. Each senate R must ban its next closest senate D who is from another party, or else D will ban its next senate from R's party.

The idea is to use two queues to save the index of each senate from R's and D's parties, respectively. During each round, we delete the banned senates' indices; and plus each remainning senate's index with n(the length of the input string senate), then move it to the back of its respective queue.

```
string predictPartyVictory(string senate) {
        queue<int> q1, q2;
        int n = senate.length();
        for(int i = 0; i<n; i++)
            (senate[i] == 'R')?q1.push(i):q2.push(i);
        while(!q1.empty() && !q2.empty()){
            int r_index = q1.front(), d_index = q2.front();
            q1.pop(), q2.pop();
            (r_index < d_index)?q1.push(r_index + n):q2.push(d_index + n);
        }
        return (q1.size() > q2.size())? "Radiant" : "Dire";
    }
```