

CS2233: Data Structures

Assignment 7

15th October, 2018

Problem Statement

- Input: A collection of disjoint singletons followed by set operations.
- Goal: Serve the following requests:
 - Given elements a and b , are they in the same set?
 - Given an element a , what is the representative element of the set containing a ?
 - Given a and b , perform union of the sets that contain a and b .
 - Given an element a , what is the *rank* of a ?

Input Format

Each line of the input looks like one of the following:

- ‘N’ followed by a positive integer n that indicates number of singleton sets to create.
- ‘?’ followed by two positive integers a and b separated by a space.
- ‘S’ followed by a positive integer a .
- ‘U’ followed by two positive integers a and b separated by a space.
- ‘R’ followed by a positive integer a .

Each of the lines above ends with a ‘\n’ character. All numbers used will fit inside an `int`. End of input is indicated by EOF.

Output Format

- If input line was “N n ”: No corresponding output.
- If input line was “U a b ”: No corresponding output.
- If input line was “? a b ”:
 - Output -1 if either of a or b is not a valid element.
 - Output 0 if a and b belong to different sets.
 - Output 1 if a and b belong to the same set.

- If input line was “S a ”:
Output the representative element of the set that contains a .
- If input line was “R a ”:
Output the rank of a if a is a valid element. Output -1 otherwise.

All output lines have to end with a ‘\n’ character.

Implementation rules

- When the request “N n ” is given, you’ll create n singleton sets namely $\{1\}, \{2\}, \dots, \{n\}$ and each element will be the representative of its own singleton set. Discard the previous collection of sets if any.
- The sets have to be stored using the *disjoint forest* implementation.
- Use an array of pointers to have random access to the node corresponding to each element.
- When the request “U $a\ b$ ” is issued, let the sets containing a and b be S_a and S_b with representative elements r_a and r_b respectively. You have to perform a union of the sets S_a and S_b and remove S_a and S_b from your collection. The union operation has to be implemented using the *Union by Rank* heuristic. Further, if ranks of r_a and r_b are equal, you should make the tree corresponding to S_b a child of r_a and thus increment rank of r_a .
- Do **not** use the *Path Compression* heuristic for Union.

Other Remarks

- **Deadline:** 27th October, 2018.

Example input

Input:

```
-----  
N 5  
R 2  
R 3  
R 9  
U 3 4  
R 3  
U 1 4  
R 4  
R 3  
? 12 21  
? 1 3  
? 3 2  
U 1 62  
S 1  
S 4  
S 62  
U 1 2  
S 2  
R 3  
U 3 5  
? 1 5  
R 3  
R 9  
N 4  
U 1 2  
R 1  
U 3 4  
R 3  
U 2 3  
R 1  
S 4  
-----
```

Output:

```
-----  
1  
1  
-1  
2  
1  
2  
-1  
1  
0  
3  
3  
-1  
3  
2  
1  
2  
-1  
2  
2  
3  
1  
-----
```