3. Write a program that implements the disjoint-set data structure using rooted forests. Also, write functions to implement the ranked union and path compression heuristics on your data structure, and compute the efficiency of the disjoint set data structure find operation by applying neither, either or both of the heuristics, by counting the total number of data accesses performed over the course of the program.

Your program must support the following functions:

> **makeset(*x*) :** creates a singleton set with element *x*.
>
> **find(*x*)** : finds the representative of the set containing the element *x*.
>
> **union(*x,y*) :** merges the sets containing elements *x* and *y* into a single set. The representative of the resulting set is assigned with find(*x*), unless the ranked union heuristic is used and the ranks of both find(*x*) and find(*y*) are different. Otherwise, the representative is assigned in accordance with the ranked union heuristic.

Note that looking up an element in the data structure must be done in O(1) time.

Input format

The input consists of multiple lines, each one containing a character from {'m', 'f', 'u', 's'} followed by zero, one or two integers. The integer(s), if given, is in the range 0 to 10000.

● Call the function makeset(*x*) if the input line contains the character 'm' followed by an integer x . Print “PRESENT” if x is already present in some set, and the value of x, otherwise.

● Call the function find(*x*) if the input line contains the character 'f' followed by an integer x. If x is found, output the value of the representative of the set containing the element x, and “NOT FOUND”, otherwise.

● Call the function union(x,y) if the input line contains the character 'u' followed by space separated integers x and y. If either x or y isn’t present in the disjoint set , print “ERROR”, without terminating. If find(x) = find(y), print find(x) itself. Otherwise, print the representative of the merged set. The representative of the merged set is assigned with find(x), unless the ranked union heuristic is used and the ranks of both find(x) and find(y) are different. Otherwise, the representative is assigned in accordance with the ranked union heuristic.

● If the input line contains the character 's', print the number of data accesses performed by the find commands by each of the data structures over the course of the program and terminate.

Output format

The output consists of multiple lines of space-separated columns. The columns correspond to the following disjoint-set data structures:

a. with neither ranked union nor path compression applied.
b. with only ranked union applied.
c. with only path compression applied.
d. with both ranked union and path compression applied.

Each line in the output contains the output of the corresponding line in the input, after applying to the respective data structures. The final line of the output contains the number of data accesses performed by the find commands by each of the data structures over the course of the program

Sample Input	**Sample Output**
m 1	1
m 2	2
m 3	3
m 4	4
m 5	5
m 6	6
m 7	7
m 8	8
m 9	9
u 1 2	1 1 1 1
u 3 4	3 3 3 3
u 5 6	5 5 5 5
u 7 8	7 7 7 7
u 9 8	9 7 9 7
u 6 8	5 5 5 5
u 4 8	3 5 3 5
u 2 8	1 5 1 5
f 9	1 5 1 5
m 10	10
u 10 9	10 5 10 5
s	38 32 33 30