Functions or Not?

by PRASHANTB1984

Problem Submissions Leaderboard Discussions

Objective

In this problem, we touch upon a basic concept that is fundamental to Functional Programming: identifying a relation which represents a valid function.

Task

You are given a set of unique (x, y) ordered pairs constituting a relation. The x-values form the domain, and the y-values form the range to which they map. For each of these relations, identify whether they may possibly represent a valid *function* or not.

Note: You do not have to find the *actual* function, you just need to determine that the relation may be representative of some valid function.

Input Format

The first line contains an integer, T, denoting the number of test cases. The subsequent lines describe T test cases, and the input for each test case is as follows:

1. The first line contains an integer, N, the number of (x, y) pairs in the test case.
2. The N subsequent lines each contain two space-separated integers describing the respective x and y values for each ordered pair.

Constraints

- $1 \leq T \leq 5$
- $2 \leq N \leq 100$
- $0 \leq x, y \leq 500$
- x and y are both integers.

Output Format

On a new line for each test case, print `YES` if the set of ordered pairs represent a valid function, or `NO` if they do not.

Sample Input

```
2
3
1 1
2 2
3 3
4
1 2
2 4
3 6
4 8
```

Sample Output

```
YES
YES
```

Explanation

Test Case 0:
$N = 3$, Ordered Pairs: $(1, 1), (2, 2), (3, 3)$ The set of ordered pairs represents a relation, which could represent a function such as $f : N \to N$, $f(x) = x$. Thus, we print `YES` on a new line.

Test Case 1:
$N = 4$, Ordered Pairs: $(1, 2), (2, 4), (3, 6), (4, 8)$
The set of ordered pairs represents a relation, which could represent a function such as $f : N \to N$, $f(x) = 2x$. Thus, we print `YES` on a new line.

More

If you cannot find your favorite language here, please check out the algorithm domain. We support over 40 popular languages.

Current Buffer (saved locally, editable) Racket

```
#lang racket
; Enter your code here. Read input from STDIN. Print output to STDOUT

(define (is-function domain)
  (equal? (remove-duplicates domain) domain))

(define (read-points n)
  (cond ((= n 0) '())
        (else (begin
                (cons (cons (read) (read)) (read-points (- n 1)))))))

(define (read-functions n)
  (cond ((= n 1) (if (is-function (map car (read-points (read)))) "YES" "NO"))
        (else
         (begin
           (displayln (if (is-function (map car (read-points (read)))) "YES" "NO"))
           (read-functions (- n 1))))))

(displayln (read-functions (read)))
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code