Walchand College of Engineering, Sangli
Computer Science & Engineering
Third Year
Course: Computer Algorithm Lab

Week 5 Assignment

Divide and conquer strategy

1) You are working on the city construction project. You have *A* houses in the city. You have to divide these houses into *B* localities such that every locality has at least one house. Also, every house in a locality should have a telephone connection wire with each of the other houses in the locality.You are given integers *A and B.*

   Task

   Print the minimum and the maximum number of telephone connections possible if you design the city accordingly.

2)You are working in the Data Consistency team of your company. You are allocated a task as follows:

- You have a data stream consisting of an equal number of odd and even numbers. You can make separations in the data stream but the number of odd elements should be equal to the number of even elements in both partitions after separation. Also, if you make a separation between a number *x* and number *y*, then the cost of this operation will be |x-y| coins.

You are given the following:

- An integer *N*
- An array *arr*
- *A*n integer *K*

Task

Determine the maximum number of separations that can be made in the array by spending no more than *K* coins.

3)

Bob has an array A of size N, and he is very fond of two integers X and Y. Find the length of the longest subarray, such that it contains exactly X distinct integers and Y exist at least once in the subarray.

Input format

- The first line contains an integer T, which denotes the number of test cases.
- The first line of each test case contains three space separated integers N, X, Y denoting the size of array A, the value of X and Y, respectively.
- The second line of each test case contains N space-separated integers, denoting the elements of array A.

Output format

For each test case, print the length of the largest subarray, such that it contains exactly X distinct integers and Y exist at least once in the subarray in a new line.

4) The country of Byteland consists of *n* cities. Between any *2* cities it is possible to have a railway track and a road. Railway tracks are bidirectional, meaning if there exists a railway track between *u* and *v* then you can take a train from *u* to *v* as well as from *v* to *u*. Similarly, roads are bidirectional, meaning if there exists a route between *u* and *v* then you can drive from *u* to *v* as well as from *v* to *u*.

2 cities, *u* and *v* are called **railway-connected** if it is possible to travel between *u* and *v* using railway tracks.

2 cities, *u* and *v* are called **road-connected** if it is possible to travel between *u* and *v* using roads.The transportation network is called **balanced** if for all pairs of cities *u*, *v*:

u,*v* are railway-connected if and only if *u*,*v* are road-connected.

Initially, there are *n* cities and no roads or railways in Byteland. You will be given *q* instructions asking you to build either a railway track or a road between some *2* cities. After each instruction, you must report whether the transportation network is balanced.

Input format

The first line of input will contain *2* integers, *n* and *q*. *q* lines will follow. Each line will contain *3* space-separated integers in one of the following formats:

1 u v : build a railway track between *u* and *v*

2 u v : build a road between *u* and *v*

Output format

You must print *q* lines. The *i*th line contains an answer to the question whether the transport network is balanced after the *i*th instruction. If it is **balanced** print "YES" (without quotes) otherwise print "NO" (without quotes)