CSC 225 SPRING 2023
ALGORITHMS AND DATA STRUCTURES I
ASSIGNMENT 3 - PROGRAMMING
UNIVERSITY OF VICTORIA

Complete the implementation of the match() method in **ArrayMatch.java**. This method determines if a ***match*** (something we are defining for this particular problem) can be found when examining two arrays, $\boldsymbol{A}$ and $\boldsymbol{B}$. $\boldsymbol{A}$ and $\boldsymbol{B}$ are arrays of size $\boldsymbol{n}$, containing the same number of **integer** elements.

Two arrays, $\boldsymbol{A}$ and $\boldsymbol{B}$, are defined to be *matches* of one another if at least one of the following two conditions is satisfied:

I. $\boldsymbol{A} = \boldsymbol{B}$ (the arrays have the same elements at each index)

II. If $\boldsymbol{n}$ is divisible by 2, $\boldsymbol{A}$ and $\boldsymbol{B}$ are divided into two sub-arrays of equal size ($\boldsymbol{A}$ is divided into $\boldsymbol{A_1}$ and $\boldsymbol{A_2}$, $\boldsymbol{B}$ into $\boldsymbol{B_1}$ and $\boldsymbol{B_2}$). Then, at least one of the following conditions is satisfied:

a) ($\boldsymbol{A_1}$ *matches* $\boldsymbol{B_1}$) AND ($\boldsymbol{A_2}$ *matches* $\boldsymbol{B_2}$)
b) ($\boldsymbol{A_1}$ *matches* $\boldsymbol{B_1}$) AND ($\boldsymbol{A_1}$ *matches* $\boldsymbol{B_2}$)
c) ($\boldsymbol{A_2}$ *matches* $\boldsymbol{B_1}$) AND ($\boldsymbol{A_2}$ *matches* $\boldsymbol{B_2}$)

Note: if $\boldsymbol{n}$ is not divisible by 2, condition II is not satisfied.

Additional Information:

You **cannot** change the method signature for match() at all (two integer arrays as parameters, and returns a boolean) or you will receive a score of **0**. If your submission fails to compile, you will receive a score of **0**. You are welcome to create additional methods to aid in your implementation, but again, the match() method must return a boolean when given two integer arrays.

The methods provided for you will handle file I/O. When executed, the program reads from input files, and outputs whether a match is found based on the array data found in the file.

The program is executed in the following way: **java ArrayMatch filename.txt**

Input files must be three lines, formatted in the following way:
<a single integer representing the size of the arrays>
<integer elements for Array A, where the elements are separated by white space>
<integer elements for Array B, where the elements are separated by white space>

You have been provided with some sample files. It is *strongly* recommended you add further tests.

Input Format

The first line has an integer $\mathbf{1} \leq \boldsymbol{n} \leq \mathbf{10^4}$. The second line has the $\boldsymbol{n}$ integers, $\boldsymbol{a_1, a_2, \ldots, a_{n-1}}$, of array $\boldsymbol{A}$, and the third line has the $\boldsymbol{n}$ integers, $\boldsymbol{b_1, b_2, \ldots, b_{n-1}}$, of array $\boldsymbol{B}$, where $\mathbf{0} \leq \boldsymbol{a_i, b_i} \leq \mathbf{10^8}$, for each $\mathbf{0} \leq \boldsymbol{i} \leq \boldsymbol{n} - \mathbf{1}$.

Output Format

On one line print "YES" if the arrays match, and "NO" if they do not. This output is case-sensitive and the quotes are just for clarity.

Sample Input 0

```
5
10 11 8 19 11
10 11 8 19 11
```

Sample Output 0

```
YES
```

Explanation 0

In this sample, condition 1 is satisfied since the two arrays are the same.

Sample Input 1

```
8
10 2 8 9 3 7 4 1
10 2 8 9 4 1 4 1
```

Sample Output 1

```
YES
```

Explanation 1

In this sample, condition 2 is satisfied in a recursive manner. In fact, A_1 and B_1 are the same and so they are a match, as well. Moreover, A_2 and B_2 are a match recursively, by condition 4.

Sample Input 2

```
6
10 2 8 9 3 7
10 10 10 9 3 7
```

Sample Output 2

```
NO
```

Explanation 2

The arrays are not exactly the same so condition 1 is not satisfied. To check other conditions, we divide the arrays in half but none of the conditions are satisfied, even recursively.

Submission

You must solve this problem using Java. You will read from standard input and print to standard output. You must use the template provided, ArrayMatch.java.

Evaluation Criteria

The programming assignment will be marked out of 20, based on a combination of automated testing and human inspection. The following score ranges will apply to this assignment; to reach top of the ranges you will need to include a correct analysis if the worst-case runtime of your algorithm (space is provided in the java template for this:

Score	Description
0 – 4	Submission does not compile.
4 – 8	Compiles but incorrectly reports YES or NO.
8 – 12	Correctly reports the solution but does so in $T(n) \in O(n^{2+\varepsilon})$ time, where $\varepsilon > 0$.
12 – 16	Correctly reports the solution but does so in $T(n) \in O(n^2)$ time.
16 – 20	Correctly reports the solution in $T(n) \in O(n^{2-\varepsilon})$ time, where $\varepsilon > 0$.