https://course.acciojob.com/idle?question=f6d441af-b555-4fcd-8b97-5c75f0fca3c9

MEDIUM

Max Score: 40 Points

Minimize the Cost

Given an array `arr` consisting of `N` integers. Your task is to find the minimum cost to sort the given array `arr` in ascending order by swapping any pair of elements `(X, Y)` such that the cost of swapping is `(X + Y)`.

Input Format

The first line contains the size of arr i.e. 'N'.

The second line contains 'N' space-separated integers 'a1, a2, a3, ..., an'.

Output Format

Print a single line containing one integer — representing the minimum cost.

Example 1

Input

```
3
3 2 1
```

Output

```
4
```

Explanation

Following are the swapping of array elements performed to sort the array:

Swapping the array elements at index 0 and 2 modifies the array to {1, 2, 3}. The cost of this swapping operation is (arr[0] + arr[2]) = (3 + 1) = 4. After the above steps, the given array is sorted and the total cost is 4, which is the minimum among all possible combinations of swapping

Example 2

Input

```
3
7 9 15
```

Output

```
0
```

Explanation

Similar explanation as example 1.

Constraints

`1 ≤ n ≤ 10^4`

`1 ≤ arr[i] ≤ 10^6`

Topic Tags

Hashing

Arrays

My code

```
// in java
import java.util.Arrays;
import java.util.HashMap;
```

```
import java.util.Map;
import java.util.Scanner;

class Solution{
	static int findMinimumCost(int[] arr, int n) {
		int ans = 0;

		int v[] = new int[n];
		boolean vis[] = new boolean[n];

		for(int i = 0; i < n; i++) {
			v[i] = arr[i];
			vis[i] = false;
		}

		Arrays.sort(v);

		Map<Integer, Integer> map = new HashMap<>();

		for(int i = 0; i < n; i++) {
			map.put(v[i], i);
		}

		for(int i = 0; i < n; i++) {
			if(vis[i] == false) {
				if(map.get(arr[i]) == i) {
					vis[i] = true;
					continue;
				}
			}
```

```
        int min_v = arr[i], sum = 0;
        int cost1, cost2;
        int j = i, k = 0;

        while(vis[j] == false) {
            // System.out.println(j);
            sum += arr[j];
            k++;

            if(arr[j] < min_v)
                min_v = arr[j];

            vis[j] = true;

            j = map.get(arr[j]);
        }


        sum -= min_v;

        cost1 = (k-1) * min_v + sum;
        cost2 = (k+1) * v[0] + 2*min_v + sum;

        if(k > 0)
            ans += Math.min(cost1, cost2);

}

return ans;
```

```
        }



}
public class Main {
        public static void main(String[] args) {
                Scanner scanner = new Scanner(System.in);
                int N = scanner.nextInt();
                int[] arr = new int[N];
                for (int i = 0; i < N; ++i) {
                        arr[i] = scanner.nextInt();
                }
                System.out.println(Solution.findMinimumCost(arr, N));
                scanner.close();
        }
}
```