**Мистер Робот и Корпорация Зла**

*Хакер Эллиот (Мистер Робот) подбирает код для проникновения в хранилище данных "Стальная гора". Он собирается взломать систему климат-контроля, чтобы уничтожить все магнитные ленты Корпорации Зла. Помогите Эллиоту подобрать подходящий смарт-контроллер, который бы допускал потенциальную возможность взлома.*

Для анализа поступает массив, в котором случайно перемешаны числа от 1 до N (без пропусков), N > 4.   
Например, N=7 [1,3,4,5,6,2,7]

Хакерская утилита может делать только одну операцию: брать любые три идущие подряд элемента массива, и сдвигать их по кругу влево произвольное количество раз. Но эту операцию она может выполнять неограниченное количество раз.

Например:

[1,3,4,5,6,2,7] [5,6,2] -> [6,2,5] -> [2,5,6]

[1,3,4,2,5,6,7] [3,4,2] -> [4,2,3] -> [2,3,4]

[1,2,3,4,5,6,7] OK

**Задача**: определить, можно ли с помощью этой операции превратить массив в упорядоченный по возрастанию. Программа должна работать быстро (укладываться в 1 секунду при N ~= 10).

Функция

bool MisterRobot(int N, int [] data)

получает значение N и сам массив, и возвращает true, если этот массив возможно упорядочить вышеописанным способом.

[как постить решение](https://skillsmart.ru/algo/lvl1/how.html)

int n = 9;  
int [] testN = new int[]{1,3,5,2,4,6,8,7,9};

проверяем массив с конца на наличие наибольшего значения

(в данном случае 9)

Дальше «вращаем влево» три элемента начиная с индекса

Оставляем наибольший из них справа

Переходим на 1 индекс влево, «вращаем» снова

Идея похожа на сортировку пузырьком, пытаемся наибольшее значиние

Постоянно сдвигать вправо

В конце сравним полученный массив и изначальный отсортированый

Если совпали - true

Вот рабочее (вроде) решение:

import java.util.\*;  
  
public class MrRobot {  
 static boolean MisterRobot(int N, int [] data){  
  
 int offset = N - 1;  
 int temp = 0;  
 boolean canBeSorted = false;  
  
 int [] sortedArr = new int[data.length];  
 for (int a = 0; a < data.length; a++) {  
 sortedArr[a] = data[a];  
 }  
 Arrays.*sort*(sortedArr);  
  
 //checking rightmost values, descending from the end  
 for (int i = data.length - 1; i > 0; i--){  
 System.*out*.println("checking index: " + i + " against value: " + N);  
 if (!(data[i] == N)){  
 offset = i;  
 System.*out*.println("offset = " + offset);  
 break;  
 }  
 N--;  
 }  
  
 //loop through remaining elements of testN, descending  
 for (int j = offset; j >= 2; j--){  
 System.*out*.println("working on indexes " + (j - 2) + " to " + j);  
 System.*out*.println("values: " + data[j - 2] + " " + data[j - 1] + " " + data[j]);  
  
 //rotating 3 elements to the left  
 for (int r = 0; r < 3; r++) {  
  
 //if rightmost element is the biggest  
 if (data[j] > data[j - 1] && data[j] > data[j - 2]) {  
 System.*out*.println("rotation complete : " + Arrays.*toString*(data));  
 break;  
 }  
  
 //moving left  
 else {  
 temp = data[j - 2];  
 data[j - 2] = data[j - 1];  
 data[j - 1] = data[j];  
 data[j] = temp;  
 }  
  
 }  
 }  
 System.*out*.println(Arrays.*equals*(sortedArr, data));  
 if (Arrays.*equals*(data, sortedArr)) {  
 canBeSorted = true;  
 }  
  
  
  
 return canBeSorted;  
 }  
  
 public static void main(String[] args) {  
 int n = 6;  
 int [] testN = new int[]{2,3,5,4,6,1};  
  
  
 *MisterRobot*(6, testN);  
  
  
  
 }  
}