**Alternative Sorting**

Show Topic Tags   

Given an array Arr[] of N **distinct** integers, print the array in such a way that the first element is first maximum and second element is first minimum and so on.  
  
**Input:**  
First line of input contains a single integer T which denotes the number of test cases. Then T test case follows. First line of each test case contains a single integer N which denotes the number of elements in the array. Second line of each test case contains N space separated integers.

**Output:**  
For each test case print the given array in such a way that the first element is first maximum and second element is first minimum and so on.  
  
**Constraints:**  
1<=T<=100  
1<=N<=104  
1<=Arr[i]<=105  
  
**Example:  
Input:**  
2  
7  
7 1 2 3 4 5 6  
8  
1 6 9 4 3 7 8 2  
**Output:**  
7 1 6 2 5 4 3  
9 1 8 2 7 3 6 4

\*\*For More Examples Use Expected Output\*\*

Contributor: Harsh Agarwal

<http://practice.geeksforgeeks.org/problems/alternative-sorting/0>

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication250;

import java.io.\*;

import java.math.\*;

import java.util.\*;

/\*\*

\*

\* @author Administrador

\*/

public class JavaApplication250 {

public static void main(String[] args) throws IOException {

// TODO code application logic here

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int t = Integer.parseInt(br.readLine());

while(t-- > 0) {

int n = Integer.parseInt(br.readLine());

String[] input = br.readLine().trim().split(" ");

int[] arr = new int[n];

for(int i =0; i<n; i++) {

arr[i] = Integer.parseInt(input[i]);

}

Arrays.sort(arr);

int i =0, j=arr.length-1;

while(i<j) {

System.out.print(arr[j] + " " + arr[i] + " ");

i++;

j--;

}

if(i == j) {

System.out.print(arr[j]);

}

System.out.println();

}

}

}