**6 kyu**

**N smallest elements in original order**

24292% of 11034 of406[GiacomoSorbi](https://www.codewars.com/users/GiacomoSorbi)

C++

* [TRAIN AGAIN](https://www.codewars.com/kata/n-smallest-elements-in-original-order/train/cpp)
* [NEXT KATA](https://www.codewars.com/trainer/cpp)

Details

[Solutions](https://www.codewars.com/kata/n-smallest-elements-in-original-order/solutions/cpp)

[Forks (5)](https://www.codewars.com/kata/n-smallest-elements-in-original-order/forks/cpp)

[Discourse (25)](https://www.codewars.com/kata/n-smallest-elements-in-original-order/discuss/cpp)

* Add to Collection
* |
* Share this kata:

Your task is to write a function that does just what the title suggests (so, fair warning, be aware that you are not getting out of it just throwing a lame bas sorting method there) with an array/list/vector of integers and the expected number n of smallest elements to return.

Also:

* the number of elements to be returned cannot be higher than the array/list/vector length;
* elements can be duplicated;
* in case of duplicates, just return them according to the original order (see third example for more clarity).

Same examples and more in the test cases:

firstNSmallest({1,2,3,4,5},3) == {1,2,3}

firstNSmallest({5,4,3,2,1},3) == {3,2,1}

firstNSmallest({1,2,3,4,1},3) == {1,2,1}

firstNSmallest({1,2,3,-4,0},3) == {1,-4,0}

firstNSmallest({1,2,3,4,5},0) == {}

[Performance version by FArekkusu](https://www.codewars.com/kata/5aeed69804a92621a7000077) also available.

<https://www.codewars.com/kata/n-smallest-elements-in-original-order/cpp>

1. #include <vector>
2. #include <set>
3. #include <iostream>
4. #include <algorithm>
5. #include <map>
6. #include <stdio.h>
8. using namespace std;
10. std::vector<int> sort(std::vector<int> arr) {
11. std::map<int, int> hash;
13. for (int i = 0; i < arr.size(); i++)
14. {
15. hash[arr[i]]++;
16. }
18. std::vector<int> ans;
19. //foreach (KeyValuePair<int, int> kvp in hash)
20. for(std::map<int,int>::iterator it = hash.begin() ; it != hash.end(); it++ )
21. {
22. for (int j = 0; j < it->second; j++)
23. {
24. //ans.Add(kvp.Key);
25. ans.push\_back(it->first);
26. }
27. }
29. return ans;
30. }
32. std::vector<int> firstNSmallest(const std::vector<int> arr, int n){
33. //your code here
35. std::vector<int> ordenado = sort(arr);
37. std::vector<int> menoresN;
39. for (int i = 0; i<arr.size() && i < n; i++)
40. {
41. menoresN.push\_back(ordenado[i]);
42. //menoresN.RemoveAt(i);
43. }
44. std::map<int, int> dic\_menores;
46. for(int i =0; i<menoresN.size(); i++)
47. {
48. dic\_menores[menoresN[i]]++;
49. }

52. std::vector<int> ans;
53. int cont = 0;
54. for (int i = 0; i < arr.size(); i++)
55. {
56. //if (menoresN.Contains(arr[i]))
57. //{
58. //    ans.Add(arr[i]);
59. //    cont++;
60. //    if (cont >= n) break;
61. //}
62. if(dic\_menores.find(arr[i]) != dic\_menores.end())
63. {
64. ans.push\_back(arr[i]);
65. dic\_menores[arr[i]]--;
66. if(dic\_menores[arr[i]] == 0)
67. {
68. dic\_menores.erase(arr[i]);
69. }
70. }
72. }
74. return ans;
76. }