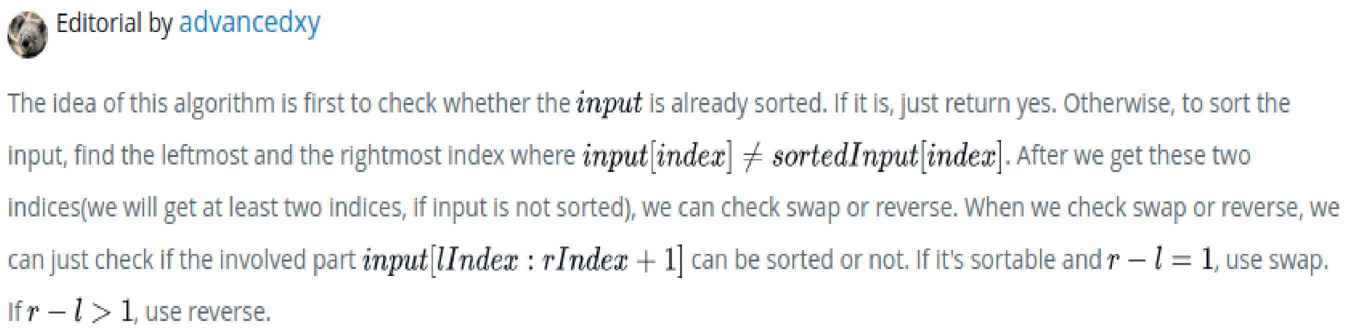
Almost Sorted

problem link: https://www.hackerrank.com/challenges/almost-sorted/problem



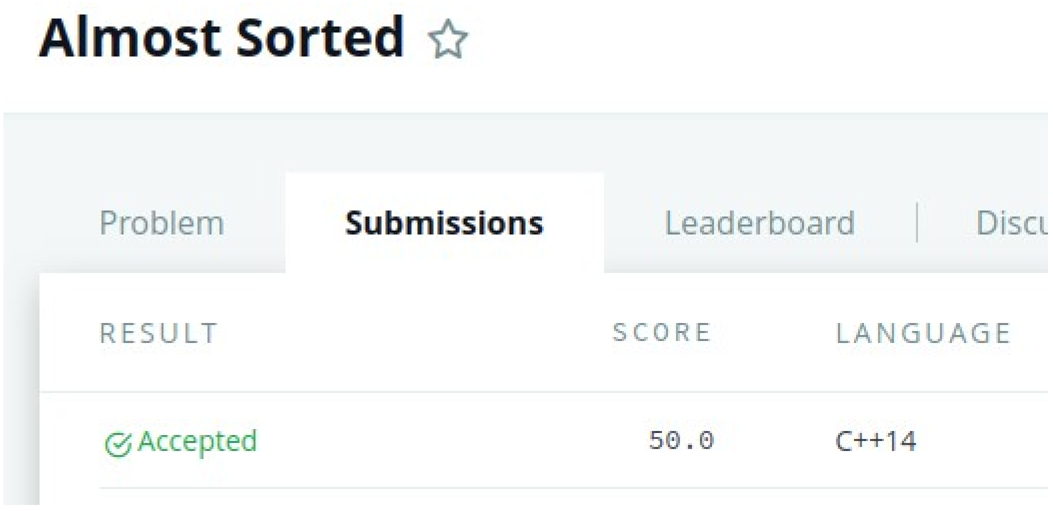
Editorial by advancedxy

The idea of this algorithm is first to check whether the $input$ is already sorted. If it is, just return yes. Otherwise, to sort the input, find the leftmost and the rightmost index where $input[index] \neq sortedInput[index]$. After we get these two indices(we will get at least two indices, if input is not sorted), we can check swap or reverse. When we check swap or reverse, we can just check if the involved part $input[lIndex : rIndex + 1]$ can be sorted or not. If it's sortable and $r - l = 1$, use swap. If $r - l > 1$, use reverse.

Whole c++ code

```
#include <bits/stdc++.h>
using namespace std;

void swap(vector<int> & arr, int i, int j){
  int tmp = arr[i];
  arr[i] = arr[j];
  arr[j] = tmp;
}

void almost_sorted(vector<int> arr, int n) {

  // Check if the array is already sorted
  if (is_sorted(arr.begin(), arr.end())) {
    cout << "yes\n";
    return;
  }

  // If not, save origial array, and sort it
  vector<int> arr_sorted = arr;
  sort(arr_sorted.begin(), arr_sorted.end());

  // Compute the left most index
  int left_most_index = 0;
  while (left_most_index < n && arr[left_most_index] ==
arr_sorted[left_most_index])
    left_most_index++;

  // Compute the right most index
  int right_most_index = n-1;
  while (right_most_index >= 0 && arr[right_most_index] ==
arr_sorted[right_most_index])
    right_most_index--;

  // Do a swap
  swap(arr, left_most_index, right_most_index);

  // Check if the array is sorted after the swap above
  if (is_sorted(arr.begin(), arr.end())) {
    cout << "yes\n";
    cout << "swap " << left_most_index+1 << ' ' << right_most_index+1 <<'\n';
    return;
  }

  // If the array still not sorted, swap back
  swap(arr, left_most_index, right_most_index);
  // Try a reverse
  reverse(arr.begin() + left_most_index , arr.begin() + right_most_index+1);

  // Check if the array is sorted after the reverse above
  if (is_sorted(arr.begin(), arr.end())) {
    cout << "yes\n";
    cout << "reverse " << left_most_index+1 << ' ' << right_most_index+1 <<'\n';
    return;
  }

  // If not, the array can not be sorted by doing a swap or a reverse
  cout << "no\n";
}
```

```

int main(){
    int n;
    cin >> n;

    vector<int> arr(n);

    for (int i = 0; i < n; i++)
      cin >> arr[i];


    almost_sorted(arr, n);



    return 0;
}
```



Almost Sorted

Problem | Submissions | Leaderboard | Discu

RESULT	SCORE	LANGUAGE
Accepted	50.0	C++14