Max wants to make an easy game. He has an array of numbers, and a number K. The player has to find such subsequence of Kadjacent elements, that the difference between its maximum and minimum elements is the greatest. If there are several possible answers, return the leftmost one.

Be the first one to play this game, and find the answer for Max's array of Numbers.

**Example:**

MaximumDisSubset([10,15,20,27,29,17], 3) = [15,20,27]

There are 4 subsequences of 3 adjacent elements, and the differences between their maximum and the minimum elements are:

* [10,15,20] --> 20 - 10 = 10;
* [15,20,27] --> 27 - 15 = 12;
* [20,27,29] --> 29 - 20 = 9;
* [27,29,17] --> 29 - 17 = 12.

So the answer is [15,20,27]

* **[input] array.integer Numbers**
  + Array of numbers, 3 ≤ |Numbers| ≤ 15.
* **[input] integer K**
  + Subsequence size, 1 ≤ K ≤ |Numbers|.
* **[output] array.integer**

<https://codefights.com/challenge/sxQ6uZphZLbP4N7gr>

--ACEPTADO--

#include <iostream>

#include <cstdio>

#include <stdio.h>

#include <vector>

using namespace std;

std::vector<int> MaximumDisSubset(std::vector<int> Numbers, int K)

{

std::vector<int> ans;

int max\_dif = INT\_MIN;

for (int i = 0; i < Numbers.size() - K + 1; i++)

{

std::vector<int> actual;

int min = Numbers[i], max = Numbers[i];

for (int j = i; j < i + K; j++)

{

min = std::min(min, Numbers[j]);

max = std::max(max, Numbers[j]);

actual.push\_back(Numbers[j]);

}

int dif = max - min;

if (dif > max\_dif)

{

max\_dif = dif;

ans = actual;

}

}

return ans;

}

int main() {

std::vector<int> res = MaximumDisSubset(std::vector<int> Numbers, int K)

return 0;

}