1) Cate inversiuni are un vector ?

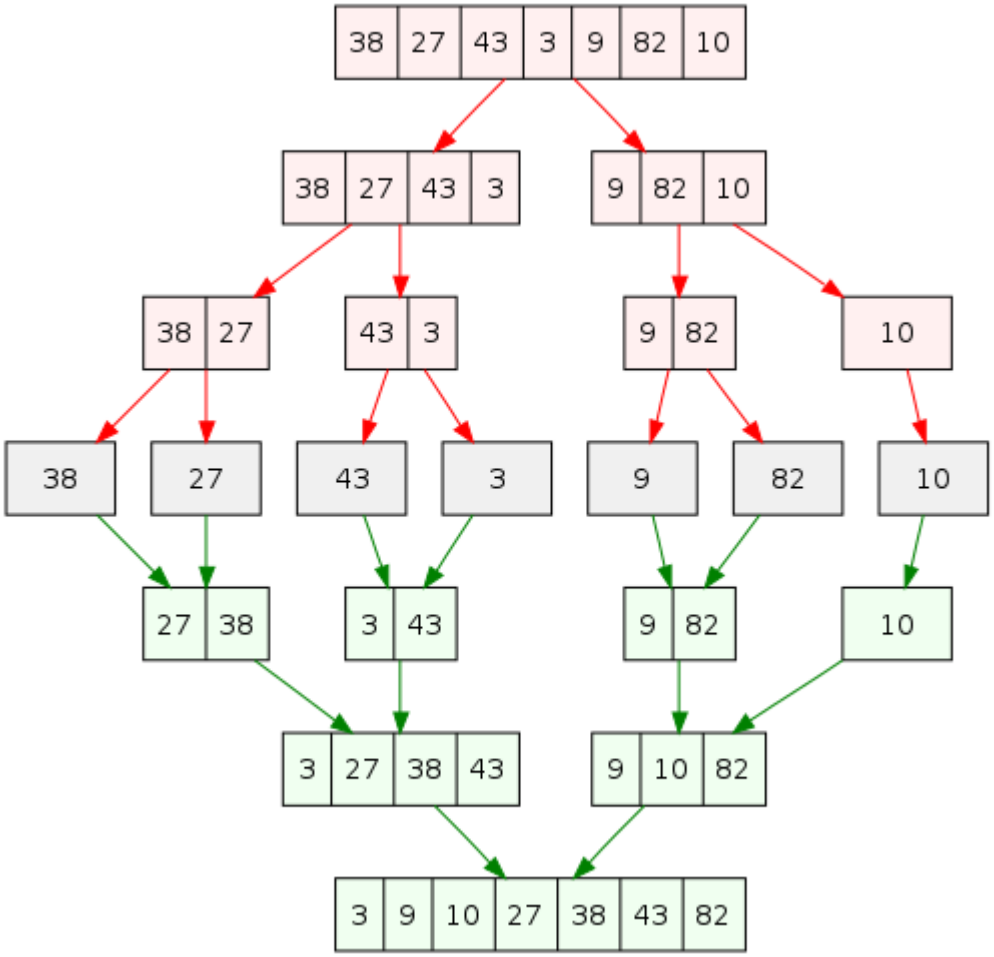
i) 3 4 1 2 5 -> are 4 inversiuni `(3 1), (3 2), (4 1), (4 2)`

ii) https://infoarena.ro/problema/inv

iii) **https://leetcode.com/problems/global-and-local-inversions/**

```
Bubble sort! O(n^2)
for (i = 1 ->n)
  for (j = 1 ->n-1)
    if(v[j] < v[j+1]) {
      swap(v[j], v[j+1]);
      nr++;
    }
return nr;

for (i =1 -> n-1)
  for ( j = i+1 ->n)
    if( v[j] < v[i]) nr++;
return nr;
4 2 3 1 5 -> 5 inversiuni ( 4, 1) (2,1) (3,1) (4,2) (4,3)
```




```
Merge sort -> numărăm inversiunile la interclasare.
Complexitate O(n log n)
```

2) Se da un vector cu n elemente **sortate.** Găsiți cate perechi de numere au suma K.
   a) Dar daca numerele nu sunt sortate?
   b) Daca nu putem folosi memorie suplimentara decat O(1) ?

1 2 10 12 14 15 16 16 16 16 **17** 4 4 **4 6 8**
K = 20

Solutii:

1. Cautare binara Pentru fiecare element caut binar pe K - v[i] … Complexitate O(n log n)..
2. Folosim 2 indici. Plasam un indice pe prima pozitie, un indice pe ultima pozitie. Daca suma e prea mica mutam indicele din stanga o pozitie spre dreapta. Altfel mutam indicele din dreapta o pozitie spre stanga. Daca suma e exact K adunam 1. Complexitate O(n)
3. Tabele Hash sau Vectori de Frecventa O(n) cu O(n) memorie nu trebuie sa fie sortat vectorul.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1		3		1		1		1		1		1	1	4	1			

```
unordered_map <int,int> fr;
for ( i = 0->n)
  fr[v[i]]++;
for ( i = 0 ->n)
 if(v[i] < S/2)
    nr += fr[S-i];
```

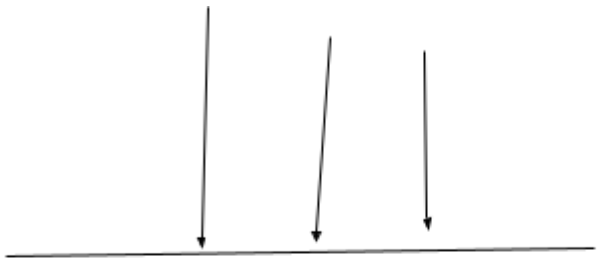
3) Se da un vector inițial sortat cu n elemente care a fost permutat circular de un număr necunoscut de ori. Găsiți cel mai mare element!
   i) 4 54 62 71 88 90 110 1 2 3 -> 110

Solutii:
1. Parcurgem element si gasim primul element care e urmat de un element mai mic. El e solutia. Daca nu gasim nici un element atunci este ultimul element. Complexitate O(n)
   a. Ar putea fi eficienta O(k) daca stim ca numarul de rotatii e mic
2. Cautam maximul clasic … O(n)..
3. Facem o cautare binara normala dar speciala :) pentru ca trebuie sa compar elementul curent cu V[0]. Daca elementul este mai mare -> ma uit in dreapta altfel ma uit in stanga….

Cautare binara bazata pe puteri a lui 2

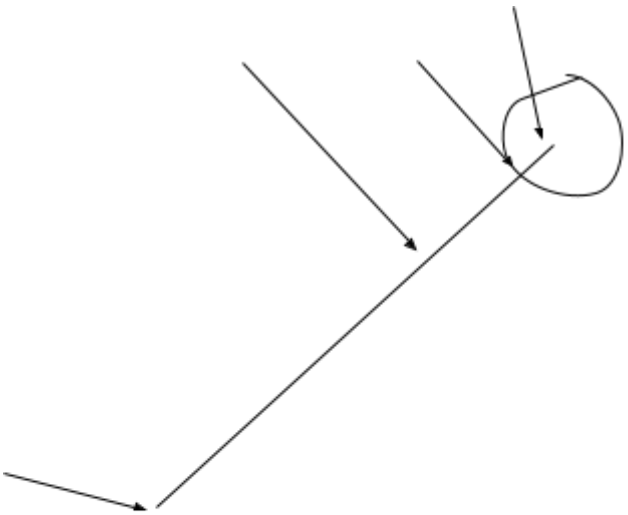
```
class Solution {
public:
    int search(vector<int>& nums, int target) {
        int max_step = (1 << 13);
        int poz = 0;
        while (max_step) {
            if (poz + max_step < nums.size() && nums[poz + max_step] <= target)
                poz += max_step;
            max_step /= 2;
```

```
        }
        if (nums[poz] == target) return poz;
        return -1;
    }
};
```



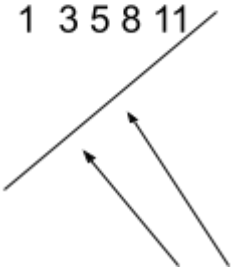
0 1 2 3 4 5 6 7 **8** 9 10 11
5 11 15 23 27 32 44 59 62 67 **82** 99

cea mai mare valoare mai mica ca 57



14 15 22 29 44 52 67 74

1 3 5 8 11



4a) Se dau 2 șiruri de caractere sunt ele anagrame ?
{aer, era, rea}
{annnr, rnann}

SEBASTIAN DANCAU 5:47 PM

```
dict = {chr(c):0 for c in range(ord('a'),ord('z')+1
cuv1 = 'aec'
cuv2 = 'era'

if len(cuv1)!=len(cuv2):
    print('Nu sunt anagrame')
    exit()

for c in cuv1:
    dict[c] += 1
for c in cuv2:
    dict[c] -= 1
for key in dict.keys():
    if dict[key] != 0:
        print('Nu sunt anagrame')
        exit()

print('Sunt anagrame')
```

Complexitate O(lungime cuvant)

a)

b) return sort(a) == sort(b)

4b) Se dau n cuvinte cate grupari de anagrame avem ?

- ana, pana, naa, leme, mele, eelm, napa, aan, papa

4 grupari

- ana, naa, aan
- pana, napa
- leme mele, eelm
- papa

fără solutie…

5) Tema optionala (adusa fizic pe foaie)...

Se dau n<=10^7 numere mai mici ca <10^9 nesortate gasiti cele mai apropiate numere:

1 9 6 4 5
-> 3 (intre 6 si 9 si 1 si 4) este diferenta 3 cea mai mica intre 2 numere consecutive dupar sortare (1 4 5 6 9)

Se cauta complexitate mai buna de O(n log n).