SEMINAR 5
Complexitatea algoritmilor. Metoda Greedy

1. *Problema candidatului majoritar*
Se consideră o listă v formată din n numere naturale nenule reprezentând voturile a n alegători. Să se afișeze, dacă există, câștigătorul alegerilor, adică un candidat care a obținut cel puțin $\left[\frac{n}{2}\right] + 1$ voturi (*candidatul majoritar*).

Exemple:
- dacă $v = [1, 5, 5, 1, 1, 5]$, atunci nu există niciun câștigător al alegerilor
- dacă $v = [7, 3, 7, 4, 7, 7]$, atunci candidatul 7 a câștigat alegerile

2. Se citește o listă de numere naturale sortată strict crescător și un număr natural S. Să se afișeze toate perechile distincte formate din valori distincte din lista dată cu proprietatea că suma lor este egală cu S.

Exemplu: Pentru lista $L = [2, 5, 7, 8, 10, 12, 15, 17, 25]$ și $S = 20$, trebuie afișate perechile $(5, 15)$ și $(8, 12)$.

3. *Problema mulțimii de acoperire*
Fie n intervale închise $I_1 = [a_1, b_1], \ldots, I_n = [a_n, b_n]$. Să se determine o mulțime M cu număr minim de elemente astfel încât $\forall k \in \overline{1, n}, \exists x \in M$ astfel încât $x \in I_k = [a_k, b_k]$. Mulțimea M se numește *mulțime de acoperire* a șirului de intervale respectiv.

Exemplu:

intervale.txt	acoperire.txt
570 670	590
500 590	680
600 680	790
690 840	930
730 790	
700 800	
900 930	

4. Fie n intervale închise $I_1 = [a_1, b_1], \ldots, I_n = [a_n, b_n]$. Să se determine reuniunea intervalelor date, precum și lungimea sa.

Exemplu:

intervale.txt	reuniune.txt
570 670	Reuniunea intervalelor:
500 590	[500, 680]
690 840	[690, 840]
600 680	[900, 930]
730 790	
700 800	Lungimea reuniunii: 360
900 930	

5. *Planificarea unor proiecte cu profit maxim*

Se consideră n proiecte, pentru fiecare proiect cunoscându-se profitul, un termen limită (sub forma unei zi din luna curentă) și faptul că implementarea sa durează exact o zi. Să se găsească o modalitate de planificare a unor proiecte astfel încât profitul total să fie maxim.

Exemplu:

proiecte.in

```
BlackFace   2 800
Test2Test   5 700
Java4All    1 150
BestJob     2 900
NiceTry     1 850
JustDoIt    3 1000
FileSeeker  3 950
OzWizard    2 900
```

proiecte.out

```
Ziua 1: BestJob 900.0
Ziua 2: FileSeeker 950.0
Ziua 3: JustDoIt 1000.0
Ziua 5: Test2Test 700.0

Profit maxim: 3550.0
```

PA

Examen 26 ian. vineri

07.12.2023

Seminar 5

(5) Din seminar 4

t = ("Popa Ion", 131, [5,4,0,3], False)

L = lista cu img despre studenți (tuplurile)

L.sort(key = lambda t: (t[1], t[0]))

L.sort(key = lambda t: (-t[3], t[0]))

L.sort(key = lambda t: (-sum(t[2]), t[1], t[3]))

L.sort(key = lambda t: (t[1], -t[3], sum(t[2], t[0])))

L.sort(key = lambda t:(

(1) v = [2,3,1,3,2,3] => nu există câștigători

v = [2,5,22,7,2] => câștigător e pe poz 2

```
def câștigător(L):
    for i in set(L):
        if L.count(i) > len(L)//2
            return i
    return None
```

=> $O(m^2)$ varianta 1

```
def câștigător(L):
    L.sort()
    m.len(L)
    if L.count(L[m//2]) > m//2:
        return L[m//2]
    return None
```

=> $O(m \log_2 m)$ varianta 2

```
def câștigător(L):
    d = {}
    for x in L:
        if x in d:
            d[x] += 1
        else:
            d[x] = 1
    for x in d:
        if d[x] > n//2
            return x
    return None
```

=> O(n) varianta 4

```
def câștigător(L):
    cmaj = None
    avantaj = 0
    for x in L:
        if avantaj == 0:
            cmaj = x
            avantaj = 1
        elif x == cmaj:
            avantaj += 1
        else:
            avantaj -= 1
    if avantaj == 0:
        return None
    elif L.count(cmaj) > len(L)//2:
        return cmaj
    return None
```

Varianta 5

=> $O(n)$

Alg Boyer - Moore (1981)

③ Optim: plec din stg și dr. listei

Dacă
- L[dr] + L[stg] > S => dr -= 1
- L[dr] + L[stg] < S => st += 1
- L[dr] + L[stg] = S => gasi pereche și dr -= 1 și str += 1

while stg < dr: $O(n)$