1 Scrieți o funcție care sortează crescător, in place, o lista de numere folosind: quicksort. Subiect eliminatoriu. (1p)

2 Specificați si testați următoarea funcție (2p):

```
def f(n):
    if n<0: raise ValueError()
    if n <= 1: return n
    l = [0] * (n + 1)
    l[1] = 1
    for i in range(2, n+1):
        l[i] = l[i - 1] + l[i - 2]
    return l[n]
```

3 Analizați complexitatea timp si spațiu a următorului algoritm. (2p).

```
def f(n):
    s = 0
    for i in range(n):
        for j in range(i):
            k = 1
            while k < n:
                k *= 2
                s += 1
```

4 Folosind metoda Divide et impera scrieți o funcție pură care găsește minimul într-o lista de numere date. Datele trebuie împărțite in 2 parți egale la fiecare pas. Ex. L = [2,3,4,1,4,5,6,7] returnează 1 (2p).

5 Pentru un n dat generați toate secvențele de paranteze si acolade care se închid corect. Exemplu: n=4 → 8 soluții: (()), ()(), (){}, ({}), {()}, {{}}, {}(), {}{}. Descrieți schematic soluția (candidat, consistent, soluție) bazată pe metoda Backtracking (fără implementare) (2p)

Obs: Subiectele se rezolva pe foaie, scris de mana.

Fiecare pagina, in coltul din dreapta sus, sa conțină: nume prenume, grupa, numărul subiectului, numerotare pagina.

Subiectele se pot rezolva in orice ordine pe foaie.

Nu trebuie sa copiat enunțul problemei (doar sa indicați clar numărul problemei rezolvate)

Daca nu se rezolva subiectul eliminatoriu (problema 1) examenul scris este picat.

Înainte de expirarea timpului trebuie sa trimiteți un singur fișier pdf, care conține poze de pe fiecare pagina de rezolvare. Pozele sa fie cat mai clare, sa aibă orientarea corecta in pdf, o poza per pagina de pdf.

Se corectează doar paginile trimise corect care se pot citi si au fost trimise pana la timpul limita anunțat.