1.(4p) Se consideră o serie de valori stocate într-un tablou $x[1..n]$. (a) Scrieţi o funcţie care primeşte ca parametri tabloul x şi indicii $k1$ şi $k2$ şi returnează valoarea medie a elementelor din subtabloul $x[k1..k2]$. (b) Scrieţi o funcţie care primeşte ca parametri tabloul x şi o valoare k $(1 < k < n)$ şi returnează tabloul y ce conţine pe poziţia i $(1 \leq i \leq n-k+1)$ media aritmetică a elementelor din subtabloul $x[i..i+k-1]$ (pentru calculul mediei se va folosi funcţia de la punctul anterior). (c) Scrieţi o funcţie care ordonează descrescător tabloul x, folosind metoda selecţiei. (d) Stabiliţi ordinul de complexitate al algoritmului descris la punctul (b).
2. (2p) Se consideră următoarea relaţie de recurenţă asociată numărului de execuţii, $T(n)$, ale operaţiei dominante dintr-un algoritm recursiv:

$$T(n) = \begin{cases} 0 & n \leq 1 \\ T(n-1)+1 & n > 1 \end{cases}$$

a) Rezolvaţi relaţia de recurenţă şi stabiliţi ordinul de complexitate al lui $T(n)$. b) Daţi un exemplu de algoritm recursiv pentru care numărul de execuţii, $T(n)$ ale operaţiei dominante satisface relaţia de recurenţă.
3.(2.5p) Se consideră un polinom de grad k specificat prin tabloul $a[0..k]$ al coeficienţilor săi. (a) Descrieţi un algoritm de complexitate liniară ($\mathcal{O}(k)$) care determină valoarea polinomului pentru un argument dat ($\sum_{i=0}^{k} a_i \cdot x^i$). (b) Stabiliţi ordinul de complexitate al algoritmului propus.
4. (2.5p) Scrieţi un algoritm care generează toate cele 2^n şiruri binare cu n elemente (elementele unui şir binar sunt valori din mulţimea $\{0, 1\}$).

1.(4p) (a)Fie $x[1..n]$ şi $y[1..n]$ două tablouri cu n elemente. Scrieţi un algoritm primeşte tablourile x şi y ca parametri şi returnează $\sum_{i=1}^{n} x_i y_i$. (b) Scrieţi un algoritm care primeşte ca parametri două matrici $A[1..n, 1..n]$ şi $B[1..n, 1..n]$ şi returnează matricea produs $C = A \cdot B$ (se poate folosi algoritmul de la punctul (a)) (c) Stabiliţi ordinul de complexitate al algoritmului descris la pct. (b); (d) Scrieţi un algoritm care transformă matricea A prin interschimbări de linii astfel încât aceastea să fie în ordinea crescătoare a sumei elementelor lor.
2.(3p) Se consideră următorul algoritm recursiv:

```
alg(x[s..d])
if s = d then return(x[s])
else  m ← (s + d)/2; r1 ← alg(x[s..m]); r2 ← alg(x[m + 1..d]);
      if r1 > r2 then return(r1) else return(r2) endif
endif
```

(a) Ce returnează algoritmul când este apelat pentru un tablou $x[1..n]$? (b) Presupunând că $n = 2^k$ şi luând în considerare doar operaţia de comparare $(r1 > r2)$ să se scrie relaţia de recurenţă care descrie evoluţia timpului de execuţie $T(n)$ (numărul de comparaţii) (c) Să se rezolve relaţia de recurenţă şi să se determine ordinul de complexitate a algoritmului (în raport cu n).
3.(2.5p) Fie $x[1..n]$ un tablou ordonat descrescător şi $a < b$ două valori care se află printre elementele tabloului. (a) Scrieţi un algoritm de complexitate $\mathcal{O}(\log n)$ care să determine numărul de elemente din x care se află în intervalul $[a, b]$ (de exemplu pentru $x = [10, 9, 7, 5, 3, 2]$ şi $a = 3$, $b = 9$ sunt 4 elemente cu proprietatea cerută); (b) Justificaţi faptul că algoritmul propus la pct. (a) are complexitatea cerută.
4. (2.5p) Se consideră o mulţime de numere naturale A. (a) Scrieţi un algoritm care verifică dacă A poate fi descompusă în două submulţimi B şi C astfel încât $A = B \cup C$ şi $\sum_{a \in A} a = \sum_{b \in B} b$ (sumele elementelor celor două submulţimi sunt egale). (b) Analizaţi complexitatea algoritmului propus.

1.(4p) Se consideră o matrice $A[1..m, 1..n]$ având elemente din mulţimea $\{1, 2, \ldots, k\}$. (a) Scrieţi un algoritm care determină toate valorile distincte din matricea A (Indicaţie: se poate utiliza un tabel de frecvenţe)(b) Scrieţi un algoritm care determină cea mai frecventă şi cea mai puţin frecventă valoare din matrice (în cazul în care mai multe valori au frecvenţa maximă/minimă se vor determina toate) (c) Stabiliţi ordinul de complexitate al algoritmului descris la pct. (b); (d) Scrieţi un algoritm care afişează valorile distincte din matrice în ordinea descrescătoare a numărului lor de apariţii.
2.(3p) Se consideră următorul algoritm recursiv:

```
alg(k)
if k = 1 then x[k] ← 0; print (x[1..n])
                x[k] ← 1; print (x[1..n])
else x[k] ← 0; alg(k − 1)
     x[k] ← 1; alg(k − 1)
endif
```

(a) Ce returnează algoritmul când este apelat pentru o valoare naturală nenulă n (în ipoteza că $x[1..n]$ este o variabilă globală)? (b) Considerând ca operaţie dominantă atribuirea de valori elementelor din tabloul x să se scrie relaţia de recurenţă care descrie evoluţia timpului de execuţie $T(n)$ (numărul de atribuiri) (c) Să se rezolve relaţia de recurenţă şi să se determine ordinul de complexitate a algoritmului.
3.(2.5p) Fie $a[1..m]$ un şir ordonat crescător, $b[1..n]$ un şir ordonat descrescător iar v o valoare. (a) Scrieţi un algoritm de complexitate $\mathcal{O}(\max\{m, n\})$ (folosind tehnica interclasării) care verifică dacă există cel puţin o pereche (i, j) cu proprietatea că $a[i] + b[i] = v$ (obs: este admis să se utilizeze tablouri adiţionale). (b) Justificaţi că algoritmul propus are complexitate liniară în raport cu m şi n.
4.(2.5p) Fie $x_1, x_2, \ldots x_n$ un şir de valori întregi nenule. Se pune problema determinării, folosind tehnica programării dinamice, sumei maxime a valorilor absolute ale elementelor a unui subşir cu semne alternate (un exemplu de subşir de elemente cu semne alternate ale şirului $x = [2, 5, -1, -4, -3, 1, -4, 4, 6, -3, 1]$ este $[2, -1, 1, -4, 4, -3, 1]$). a) Scrieţi relaţia de recurenţă corespunzătoare sumei maxime; b) Scrieţi un algoritm care dezvoltă relaţia de recurenţă şi determină suma maximă.