

NOTĂ IMPORTANTĂ:

În lipsa altor precizări:

- Presupunem că toate operațiile aritmetice se efectuează pe tipuri de date nelimitate (nu există *overflow* / *underflow*).
- Numerotarea indicilor tuturor șirurilor începe de la 1.
- Toate restricțiile se referă la valorile parametrilor actuali la momentul apelului inițial.
- O subsecvență a unui vector este formată din elemente care ocupă poziții consecutive în vector.

1. Se consideră algoritmul $\text{ceFace}(a, b)$, unde a și b sunt numere naturale ($0 \leq a, b \leq 10^4$).

```
Algorithm ceFace(a, b):
  c ← 0
  bc ← b
  While bc ≠ 0 execute
    c ← c * 10 + bc MOD 10
    bc ← bc DIV 10
  EndWhile
  If c ≠ a then
    Return ceFace(a - 1, b - 1)
  EndIf
  Return a
EndAlgorithm
```

Care este efectul apelului $\text{ceFace}(a, a)$?

- A. Algoritmul returnează cel mai mic palindrom mai mare sau egal cu a .
☒ B. Algoritmul returnează cel mai mare palindrom mai mic sau egal cu a .
 C. Algoritmul returnează cel mai mic palindrom mai mare decât a .
 D. Algoritmul returnează cel mai mare număr par mai mic sau egal cu a .

2. Se consideră algoritmul $\text{creareTablou}(n, m, x)$, unde n, m sunt numere naturale ($1 \leq n, m \leq 100$), iar x este un tablou bidimensional cu $n * m$ elemente numere întregi ($x[1][1], x[1][2], \dots, x[n][m]$, $0 \leq x[i][j] \leq 10^4$, pentru $i = 1, 2, \dots, n; j = 1, 2, \dots, m$).

```
Algorithm creareTablou(n, m, x):
  k ← 0
  For i ← 1, n execute
    For j ← 1, m execute
      If k MOD 2 ≠ 0 then
        x[i][j] ← k * k
      EndIf
      Write x[i][j], " "
      k ← k + 1
    EndFor
    Write new line
  EndFor
EndAlgorithm
```

Ce afișează acest algoritm dacă elementele tabloului x sunt inițializate cu 0?

- ☒ A. Algoritmul afișează elementele tabloului bidimensional x , în care se află valori egale cu 0 și primele $(n * m) \text{ DIV } 2$ pătrate perfecte impare.
☒ B. Algoritmul afișează elementele tabloului bidimensional x , în care se află valori egale cu 0 și primele pătrate perfecte pare.
 C. Algoritmul afișează elementele tabloului bidimensional x , în care se află șirul primelor $(n * m) \text{ DIV } 2$ pătrate perfecte pare.
☒ D. Algoritmul afișează elementele tabloului bidimensional x , în care – dacă am așeza elementele linie după linie – pătratele perfecte impare ar apărea în ordine crescătoare, eventual precedate și/sau urmate de valori egale cu 0.

3. Se consideră algoritmul $\text{something}(n, x)$, unde n este număr natural ($1 \leq n \leq 10^4$), iar x este un vector de n numere naturale ($x[1], x[2], \dots, x[n]$, $1 \leq x[i] \leq 10^6$, pentru $i = 1, 2, \dots, n$).

```
Algorithm something(n, x):
  s ← 0
  For i ← 1, n execute
    nr ← 1
    While x[i] > 9 execute
      nr ← nr + 1
      x[i] ← x[i] DIV 10
    EndWhile
    s ← s + nr
  EndFor
  Return s
EndAlgorithm
```

Ce returnează apelul $\text{something}(5, [222, 2043, 29, 2, 20035])$?

- A. 16
 B. 10
 C. 11
☒ D. 15

4. Fie algoritmul $\text{ceFace}(n, v, a)$, unde n și v sunt două numere naturale ($1 \leq n, v \leq 10^4$), iar a este un șir de numere naturale cu n elemente ($a[1], a[2], \dots, a[n]$).

```
Algorithm ceFace(n, v, a):
  For i ← 1, n execute
    d ← v
    If a[i] ≠ 0 then
      găsit ← False
      While (d ≤ v * a[i]) AND (NOT găsit) execute
        If ((d DIV a[i]) * a[i] = d) AND ((d DIV v) * v = d) then
          găsit ← True
        Else
          d ← d + 1
        EndIf
      EndWhile
    EndIf
    v ← d
  EndFor
  Return v
EndAlgorithm
```

Care este valoarea returnată de algoritm, dacă $n = 4$, $v = 3$ și $a = [5, 4, 2, 10]$?

- A. 20
 B. 120
☒ C. 60
 D. 15

5. Se consideră algoritmul $\text{calcul}(v, n)$, unde n este număr natural ($1 \leq n \leq 10^4$), iar v este un vector cu n elemente numere naturale ($v[1], v[2], \dots, v[n]$, $1 \leq v[i] \leq 10^4$, pentru $i = 1, 2, \dots, n$).

```
Algorithm calcul(v, n):
  i ← 1
  While i ≤ n DIV 2 execute
    p ← 0
    While v[i] ≠ 0 execute
      p ← p + 1
      v[i] ← v[i] DIV 10
    EndWhile
    q ← 0
    While v[n + 1 - i] ≠ 0 execute
      q ← q + 1
      v[n + 1 - i] ← v[n + 1 - i] DIV 10
    EndWhile
    If p ≠ q then
      Return False
    EndIf
    i ← i + 1
  EndWhile
  Return True
EndAlgorithm
```

În care din următoarele situații algoritmul returnează *True*?

- A. Dacă vectorul v este format din valorile $[12, 12, 2, 5466, 3, 111, 1, 3, 44]$ și $n = 9$.
☒ B. Dacă vectorul v este format din valorile $[12, 345, 2, 5466, 3, 111, 10]$ și $n = 7$.
☒ C. Dacă elementele vectorului v au același număr de cifre. 12227
☒ D. Dacă vectorul format din numărul cifrelor elementelor vectorului v formează un palindrom; de exemplu, din $v = [8, 37, 3]$ se formează vectorul $[1, 2, 1]$, care este palindrom.

6. Se consideră algoritmul $\text{alg}(n)$, unde n este număr natural ($0 \leq n \leq 10^4$).

```
Algorithm alg(n):
  If n = 0 then
    Return 0
  Else
    If n MOD 2 = 0 then
      Return alg(n DIV 10) + n MOD 10
    Else
      Return alg(n DIV 10)
    EndIf
  EndIf
EndAlgorithm
```

Care dintre următoarele afirmații sunt adevărate?

- A. Apelul $\text{alg}(123)$ returnează 6.
 B. Algoritmul calculează suma cifrelor aflate pe poziții pare ale numărului dat.
☒ C. Algoritmul calculează suma cifrelor pare ale numărului dat.
 D. Algoritmul calculează suma cifrelor numărului dat.

7. Se consideră algoritmul $f(x)$, unde x este un număr natural nenul ($1 \leq x \leq 10^5$).

```

Algorithm f(x):
  If x > 0 then
    x ← x DIV 2
    f(x)
    Write x, " "
    x ← x DIV 2
    f(x)
  EndIf
EndAlgorithm

```

Precizați ce se afișează în urma apelului $f(10)$.

- ☒ A. 0 1 2 0 5 0 1
☐ B. 0 1 2 5 1 0
☐ C. 1 2 1 5 2 1
☐ D. 1 2 1 1 5 1 2

8. Se consideră matricea pătratică M de dimensiune n care conține numere naturale, unde n este număr natural nenul ($1 \leq n \leq 10^4$, $M[1][1], \dots, M[1][n], M[2][1], \dots, M[2][n], \dots, M[n][1], \dots, M[n][n]$, $1 \leq M[i][j] \leq 10^4$, pentru $i = 1, 2, \dots, n, j = 1, 2, \dots, n$). Se consideră următorul algoritim:

```

Algorithm what(M, n):
  up ← 1
  down ← n
  left ← 1
  right ← n
  While left ≤ right AND up ≤ down execute
    For i ← left, right execute
      Write M[up][i], " "
    EndFor
    up ← up + 1
    For i ← up, down execute
      Write M[i][right], " "
    EndFor
    right ← right - 1
    For i ← right, left, -1 execute
      Write M[down][i], " "
    EndFor
    down ← down - 1
    For i ← down, up, -1 execute
      Write M[i][left], " "
    EndFor
    left ← left + 1
  EndWhile
EndAlgorithm

```

Ce se afișează pentru următoarea matrice M ?

1	2	3
8	9	4
7	6	5

- ☐ A. 1 2 3 4 9 8 7 6 5
☒ B. 1 2 3 4 5 6 7 8 9
☐ C. 1 2 3 4 5 8 9 7 6
☐ D. 1 8 7 6 5 4 3 2 9

9. Fie algoritmul $ce_face(a, b)$, unde a și b sunt numere naturale ($1 \leq a, b \leq 10^4$).

```

Algorithm ce_face(a, b):
  If a = 1 then
    Return 1
  Else
    If a MOD b = 0 then
      Return ce_face(a DIV b, b)
    Else
      Return 0
    EndIf
  EndIf
EndAlgorithm

```

Precizați afirmațiile adevărate:

- ☒ A. În cazul apelului $ce_face(1, 2)$ algoritmul returnează 1.
☒ B. În cazul apelului $ce_face(24, 2)$ algoritmul returnează 0.
☐ C. În cazul apelului $ce_face(2024, 4)$ algoritmul returnează 4.
☐ D. În cazul apelului $ce_face(8, 3)$ algoritmul returnează 2.

10. Fie algoritmi $decide(n)$ și $compute(m)$, unde n și m sunt numere naturale nenule ($1 \leq n, m \leq 10^4$):

```

Algorithm decide(n):
  result ← -1
  m ← 0
  While n > 0 execute
    m ← m * 10 + n MOD 10
    n ← n DIV 10
  EndWhile
  If m MOD 3 = 0 then
    result ← 1
  EndIf
  Return result
EndAlgorithm

```

```

Algorithm compute(m):
  cnt ← 0
  For k ← 0, m - 1 execute
    cnt ← cnt + decide(k)
  EndFor
  Return cnt
EndAlgorithm

```

Pentru ce valori ale lui m algoritmul $compute(m)$ va returna -33?

- ☐ A. 100
☒ B. 99
☐ C. 98
☒ D. 101

3

11. Se consideră algoritmul $f(n, x)$, unde n și x sunt numere naturale ($1 \leq n \leq 10^5$, $2 \leq x \leq 10$):

```

Algorithm f(n, x):
  If n > 0 then
    f(n DIV x, x)
    Write n MOD x
  EndIf
EndAlgorithm

```

Care din următoarele afirmații sunt adevărate?

- ☒ A. Algoritmul afișează reprezentarea numărului n în baza de numeratie x .
☐ B. Algoritmul afișează restul împărțirii întregi a numărului x la numărul n .
☐ C. Algoritmul afișează numărul de cifre al reprezentării în baza x a numărului n .
☐ D. Algoritmul verifică dacă numărul n este divizibil cu x .

12. Se consideră algoritmul $ceFace(n)$, unde n este număr natural ($1 \leq n \leq 10^9$).

```

Algorithm ceFace(n):
  If n ≤ 9 then
    If n MOD 2 = 0 then
      Return n
    Else
      Return -1
    EndIf
  EndIf
  x ← n MOD 10
  y ← ceFace(n DIV 10)
  If x MOD 2 ≠ 0 then
    Return y
  EndIf
  If x > y then
    Return x
  EndIf
  Return y
EndAlgorithm

```

Care dintre următoarele afirmații sunt adevărate?

- ☒ A. Algoritmul returnează un număr format dintr-o singură cifră, sau -1.
☒ B. Algoritmul returnează un număr impar.
☒ C. Algoritmul returnează cifra impară maximă a numărului n , sau -1.
☒ D. Algoritmul returnează cifra pară maximă a numărului n , sau -1.

13. Se consideră algoritmul $decide(n, x)$, unde n este număr natural ($1 \leq n \leq 10^4$), iar x este un vector cu n elemente numere întregi ($x[1], x[2], \dots, x[n]$, $-100 \leq x[i] \leq 100$, pentru $i = 1, 2, \dots, n$):

```

Algorithm decide(n, x):
  b ← True
  i ← 1
  While b = True AND i < n execute
    If x[i] < x[i + 1] then
      b ← True
    Else
      b ← False
    EndIf
    i ← i + 1
  EndWhile
  Return b
EndAlgorithm

```

În care din următoarele situații algoritmul returnează *True*?

- ☒ A. Dacă vectorul $x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$ și $n = 10$
☒ B. Dacă $n > 1$ și elementele vectorului x sunt în ordine strict crescătoare
☐ C. Dacă vectorul x nu are elemente negative
☐ D. Dacă vectorul x are elemente pozitive situate înaintea celor negative

$$x = 2^m \quad y = 3^m$$

14. Fie x și y două numere naturale pozitive cu proprietățile: x este putere a lui 2 și y este multiplu de 3. Fie expresia logică $((x * y + 3) \text{ DIV } 6 = 10) \text{ OR } (((x * y) \text{ MOD } 6 = 0) \text{ AND } ((x + y) \text{ MOD } 4 = 0))$

Care dintre următoarele afirmații sunt adevărate pentru perechi de numere care respectă proprietățile din enunț:

- ☒ A. Există o pereche (x, y) pentru care expresia este adevărată.
☐ B. Există o pereche (x, y) pentru care expresia este falsă.
☒ C. Există perechile (x_1, y_1) și (x_2, y_2) , cu $x_1 \neq x_2$ și $y_1 \neq y_2$ în așa fel încât expresia este adevărată pentru ambele perechi.
☒ D. Expresia este falsă pentru orice pereche (x, y) .

15. Se consideră două numere naturale n și m ($1 \leq n, m \leq 256$) respectiv șirurile de caractere a , având n caractere ($a[1], a[2], \dots, a[n]$) și șirul b având m caractere ($b[1], b[2], \dots, b[m]$).

Care dintre următorii algoritmi returnează *True* dacă șirul a poate fi format pornind de la șirul b prin eliminarea unor caractere, fără a modifica poziția relativă a caracterelor rămase, și *False* în caz contrar. De exemplu, șirul "ace" poate fi format prin eliminarea de caractere din șirul "abcde", dar șirul "aec" nu poate fi obținut prin acest procedeu.

4

$$m = 101$$

$$[0, 100] \rightarrow 33 \times 1 = 34$$

$$-67 + 34 = 33$$

A.

```

Algorithm hasProperty(a, b, n, m):
  If n = 0 then
    Return True
  EndIf
  If m = 0 then
    Return False
  EndIf
  If a[n] = b[m] then
    Return hasProperty(a, b, n - 1, m - 1)
  EndIf
  Return hasProperty(a, b, n, m - 1)
EndAlgorithm

```

B.

```

Algorithm hasProperty(a, b, n, m):
  i ← 1
  j ← 1
  While i ≤ n AND j ≤ m execute
    If a[i] = b[j] then
      i ← i + 1
    EndIf
    j ← j + 1
  EndWhile
  If i > n then
    Return True
  Else
    Return False
  EndIf
EndAlgorithm

```

C.

```

Algorithm hasProperty(a, b, n, m):
  i ← n
  j ← m
  While i + j > 0 execute
    If a[i] = b[j] then
      i ← i - 1
    EndIf
    j ← j - 1
  EndWhile
  If i = 0 then
    Return True
  Else
    Return False
  EndIf
EndAlgorithm

```

D.

```

Algorithm hasProperty(a, b, n, m):
  If n > m then
    Return False
  EndIf
  i ← 1
  j ← 1
  While i < n execute
    If a[i] = b[j] then
      i ← i + 1
    EndIf
    j ← j + 1
  EndWhile
  If i > m then
    Return True
  Else
    Return False
  EndIf
EndAlgorithm

```

16. Se consideră algoritmul ceva(x , n , e), unde x este un vector cu n elemente distincte întregi ($x[1], x[2], \dots, x[n]$, $1 \leq n \leq 10^3$ și $x[i] \neq x[j]$, pentru $1 \leq i < j \leq n$) și e este un număr întreg. Algoritmul caută elementul e în vectorul x , și dacă îl găsește, mută elementul pe prima poziție din vector și returnează *True*, nemodificând ordinea relativă a celorlalte elemente. Dacă e nu se găsește în x , algoritmul returnează *False* și nu modifică conținutul vectorului. De exemplu, pentru vectorul x cu elementele $[-100, 2, 71, 31, -62, 51]$ și $e = 31$, algoritmul va returna *True* și vectorul x va deveni $[31, -100, 2, 71, -62, 51]$. Care dintre următoarele variante este o implementare corectă pentru algoritmul ceva(x , n , e) și are complexitate timp $O(n)$?

A.

```

Algorithm ceva(x, n, e):
  index ← 1
  While index ≤ n execute
    If x[index] = e then
      tmp ← x[index]
      x[index] ← x[1]
      x[1] ← tmp
      Return True
    EndIf
    index ← index + 1
  EndWhile
  Return False
EndAlgorithm

```

rețineți ordinea

B.

```

Algorithm ceva(x, n, e):
  index ← 2
  tmp ← x[1]
  While index ≤ n execute
    If x[index] = e then
      x[1] ← e
      x[index] ← tmp
      Return True
    EndIf
    tmp2 ← x[index]
    x[index] ← tmp
    tmp ← tmp2
    index ← index + 1
  EndWhile
  Return False
EndAlgorithm

```

C.

```

Algorithm ceva(x, n, e):
  index ← n
  While index > 1 execute
    If x[index] = e then
      index2 ← index
      While index2 > 1 execute
        x[index2] ← x[index2 - 1]
        index2 ← index2 - 1
      EndWhile
      x[index2] ← e
    EndIf
    index ← index - 1
  EndWhile
  If x[1] = e then
    Return True
  Else
    Return False
  EndIf
EndAlgorithm

```

D. Niciuna dintre variantele A, B, C

17. Se consideră algoritmul expresie(x , y , z), unde x, y, z sunt numere naturale ($0 \leq x, y, z \leq 10^4$):

```

Algorithm expresie(x, y, z):
  If x = 0 then
    Return z
  Else
    Return expresie(x - 1, y, x * x + y * y + z)
  EndIf
EndAlgorithm

```

Precizați expresia a cărei valoare o calculează și returnează algoritmul:

- A. $\sum_{i=1}^x i^2 + \sum_{j=1}^y x \cdot y + \sum_{k=1}^z 1$
 B. $\sum_{i=1}^x i^2 + \sum_{j=1}^y j^2 + z$
 C. $\sum_{i=1}^x i^2 + x \cdot y^2 + z$
 D. $\sum_{i=1}^x i^2 + \sum_{j=1}^y j^2 + \sum_{k=1}^z k$

$A = C$

18. Se consideră algoritmul ceFace(v , a , b), unde v este un vector cu n elemente din mulțimea $\{0, 1\}$, ($1 \leq n \leq 10^4$, $v[1], \dots, v[n]$), iar a și b sunt numere naturale nenule. Vectorul v este ordonat crescător.

```

Algorithm ceFace(v, a, b):
  If b - a + 1 = 0 then
    Return 0
  EndIf
  If v[a] = 1 then
    Return b - a + 1
  EndIf
  If v[b] = 0 then
    Return 0
  EndIf
  c ← (a + b) DIV 2
  Return ceFace(v, a, c) + ceFace(v, c + 1, b)
EndAlgorithm

```

Care dintre următoarele afirmații sunt adevărate, considerând că apelul inițial este ceFace(v , 1, n)?

- A. Dacă vectorul v conține cel puțin o valoare de 1, atunci se returnează lungimea vectorului.
 B. Dacă vectorul v conține doar valori de 1, atunci se returnează valoarea lui n .
 C. Dacă vectorul v conține doar valori de 0, atunci se returnează 0.
 D. Se returnează numărul de valori 1 conținute de vectorul v .

19. Se știe că numărul total de șiruri binare (care conțin doar caracterele 0 și 1) de lungime n este 2^n . De exemplu, pentru $n = 2$ acestea sunt 00, 01, 10 și 11, numărul lor fiind $2^2 = 4$. Șirul 100011 are lungimea 6 și conține ca subsecvență toate cele 4 șiruri posibile de lungime $n = 2$, fiindcă începând cu prima poziție apare 10, începând cu a doua poziție apare 00, începând cu a patra poziție apare 01 și începând cu a cincea poziție apare 11.

Care este lungimea minimă a unui șir, care conține ca subsecvență toate cele 2^n șiruri binare posibile pentru $n = 4$?

- A. 18 B. 19 C. 20 D. 21

20. Se consideră algoritmul $t(q, x, y)$, unde q este un caracter oarecare, iar x și y sunt numere naturale nenule ($1 \leq x, y \leq 100$).

```

Algorithm t(q, x, y):
  If x ≤ y then
    Write q
  Else
    If x MOD y = 0 then
      t(q, x + 1, y - 2)
    Else
      If (x DIV y) MOD 2 ≠ 0 then
        t(q, x - 1, y + 2)
        Write 'c'
      Else
        t(q, x - 1, y - 1)
        Write "cc"
      EndIf
    EndIf
  EndIf
EndAlgorithm

```

Precizați care dintre următoarele afirmații sunt adevărate:

- A. În urma apelurilor $t('c', 33, 28)$, $t('c', 10, 6)$ și $t('c', 22, 16)$ se afișează aceleași caractere.
 B. În urma apelurilor $t('c', 33, 28)$ și $t('c', 45, 40)$ nu se afișează aceleași caractere.
 C. În urma apelului $t('c', 11, 8)$ se afișează "cc".
 D. În urma apelului $t('c', 25, 16)$ nu se afișează "cccc".

21. Se consideră algoritmul $hIndex(x, n)$, unde x este un vector cu n ($1 \leq n \leq 10^5$) elemente numere naturale nenule ($x[1], x[2], \dots, x[n]$). Definim $hIndex$ -ul vectorului x , ca fiind cea mai mare valoare v pentru care este adevărat că există cel puțin v valori în x care sunt mai mari sau egale cu v . De exemplu, pentru $x = [3, 10, 2, 7, 10, 8, 50, 1, 1]$ $hIndex$ -ul este 5.

```

1. Algorithm hIndex(x, n):
2.   h ← 1
3.   cont ← True
4.   While cont = True AND h ≤ n execute
5.     pos ← h
6.     For i ← h + 1, n execute
7.       If x[i] > x[pos] then
8.         pos ← i
9.       EndIf
10.    EndFor
11.    If pos ≠ h then
12.      tmp ← x[pos]
13.      x[pos] ← x[h]
14.      x[h] ← tmp
15.    EndIf
16.    If x[h] ≥ h then
17.      h ← h + 1
18.    Else
19.      cont ← False
20.    EndIf
21.  EndWhile
22.  ...
23. EndAlgorithm

```

Care dintre următoarele afirmații sunt adevărate?

- A. În momentul în care s-ar executa linia 22 vectorul x este sortat descrescător.
 B. Algoritmul $hIndex(x, n)$ returnează $hIndex$ -ul vectorului x dacă pe linia 22 scriem instrucțiunea Return h.
 C. Algoritmul $hIndex(x, n)$ returnează $hIndex$ -ul vectorului x dacă pe linia 22 scriem instrucțiunea Return h - 1.
 D. Dacă algoritmul $hIndex(x, n)$ se apelează pentru un vector x sortat strict descrescător, atunci algoritmul nu returnează $hIndex$ -ul vectorului x , indiferent ce instrucțiune adăugăm pe linia 22.

h index - cel mai mare index cu x[h] ≥ h

22. Se consideră algoritmul $ceFace(n, k, x, p)$, unde n, k și p sunt numere naturale nenule ($1 \leq n, k, p \leq 10, p \leq n$), iar x este un vector cu $p + 1$ elemente numere naturale ($x[0], x[1], \dots, x[p]$). Presupunem că $x[0]$ este inițializat cu 0.

```

Algorithm ceFace(n, k, x, p):
  If k > p then
    For i ← 1, p execute
      Write x[i]
    EndFor
    Write " " //un singur spațiu
  Else
    For i ← x[k - 1] + 1, n execute
      x[k] ← i
      ceFace(n, k + 1, x, p)
    EndFor
  EndIf
EndAlgorithm

```

Precizați care dintre următoarele variante de răspuns sunt corecte.

- A. După ce algoritmul se apelează sub forma $ceFace(3, 1, x, 3)$ acesta se va mai autoapela de 6 ori.
 B. Dacă $x[0]$ se inițializează cu o valoare diferită de 0, în urma apelului $ceFace(5, 1, x, 3)$ numărul de spații afișate este diferit de 10.
 C. Dacă algoritmul se apelează sub forma $ceFace(5, 1, x, 4)$ se afișează numerele 1245 1234 1345 1235 2345, dar în altă ordine.
 D. Dacă algoritmul se apelează sub forma $ceFace(5, 1, x, 3)$ rezultatul afișat este 123 124 125 134 135 145 234 235 în această ordine.

Combi mari

7

245

23. Se consideră algoritmul $f(sir, s, d, p)$, unde sir este un șir de caractere, iar s, d, p sunt numere naturale nenule ($0 < s, d, p < 10^9$). Operatorul "+" reprezintă operatorul de concatenare a două șiruri de caractere. Algoritmul $print(a)$ afișează șirul de caractere a , apoi trece la linie nouă.

```

1. Algorithm f(sir, s, d, p):
2.   If s = p AND d = p then
3.     print(sir)
4.   EndIf
5.   If s < p then
6.     f(sir + "-1 ", s + 1, d, p)
7.   EndIf
8.   If s > d then
9.     f(sir + " 1 ", s, d + 1, p)
10.  EndIf
11. EndAlgorithm

```

Precizați care dintre următoarele afirmații sunt adevărate în urma apelului $f("", 0, 0, 2)$:

- A. Se afișează două șiruri de caractere pe linii separate, fiecare conținând 4 numere a căror sumă este 0 (de exemplu, suma numerelor din șirul de caractere "-1 1 -1 1" este 0)
 B. Se afișează doar "-1 -1 1 1".
 C. Se afișează doar "-1 -1 1 1", dar algoritmul nu își termină execuția din cauza unei erori.
 D. Dacă pe linia 2 s-ar înlocui operatorul AND cu OR, atunci s-ar afișa doar "-1 -1".

24. Se consideră algoritmul $ceFace(a, i, n)$, unde i și n sunt numere naturale ($1 \leq i, n \leq 100$), iar a este un vector cu n elemente numere întregi ($a[1], a[2], \dots, a[n]$, $-100 \leq a[i] \leq 100$). În șirul a se află cel puțin un număr pozitiv. Algoritmul $max(x, y, z)$ returnează maximul dintre trei numere întregi x, y și z ($-10^4 \leq x, y, z \leq 10^4$). Algoritmul $ceFace(a, i, n)$ apelează algoritmul intermediar(a, i, m, n), unde parametrii a, i și n au semnificația de mai sus, iar m este un număr natural ($1 \leq m \leq n$).

```

Algorithm intermediar(a, i, m, n):
  s ← 0
  left ← a[m]
  For k ← m, i, -1 execute
    s ← s + a[k]
    If s > left then
      left ← s
    EndIf
  EndFor
  right ← a[m]
  For i ← m, n execute
    s ← s + a[i]
    If s > right then
      right ← s
    EndIf
  EndFor
  Return max(left, right, left + right - a[m])
EndAlgorithm

```

```

Algorithm ceFace(a, i, n):
  If i ≥ n then
    Return a[i]
  EndIf
  m ← (i + n) DIV 2
  v1 ← ceFace(a, i, m - 1)
  v2 ← ceFace(a, m + 1, n)
  v3 ← intermediar(a, i, m, n)
  Return max(v1, v2, v3)
EndAlgorithm

```

Precizați care dintre următoarele afirmații sunt adevărate dacă algoritmul se apelează sub forma $ceFace(a, i, n)$:

- A. Algoritmul identifică o poziție m a vectorului a astfel încât fie suma elementelor de pe pozițiile 1, 2, ..., m , fie suma elementelor de pe pozițiile $m, m + 1, \dots, n$ să fie maximă care se poate obține pentru orice $1 \leq m \leq n$, și returnează suma maximă obținută astfel.
 B. Algoritmul returnează suma maximă care se poate obține însumând elementele unei submulțimi a valorilor vectorului a .
 C. Algoritmul returnează suma maximă care se poate obține pentru o subsecvență a vectorului a .
 D. În cazul în care toate elementele vectorului a sunt pozitive, algoritmul returnează suma tuturor elementelor vectorului a .

