

Concurs de admitere – 19 iulie 2023  
Proba scrisă la Informatică

NOTĂ IMPORTANTĂ:  
În lipsa altor precizări:

- Presupuneți că toate operațiile aritmetice se efectuează pe tipuri de date nelimitate (nu există *overflow* / *underflow*).
- Numerotarea indicilor tuturor sirurilor 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  $F(x)$ , unde  $x$  este număr natural ( $1 \leq x \leq 10^6$ ):

```
Algorithm F(x):
  If x = 0 then
    Return 0
  Else
    If x MOD 3 = 0 then
      Return F(x DIV 10) + 1
    Else
      Return F(x DIV 10)
    EndIf
  EndIf
EndAlgorithm
```

Pentru care dintre următoarele apeluri se returnează 4?

A. F(21369)  
 B. F(6933)  
 C. F(4)  
 D. F(16639)

2. Se consideră algoritmul  $ceFace(a, b)$ , unde  $a$  și  $b$  sunt numere naturale ( $1 \leq a, b \leq 10^4$ ) care nu conțin cifra 0.

```
Algorithm ceFace(a, b):
  p ← 0
  While a ≠ 0 execute
    c ← a MOD 10
    p ← p * 10 + c
    a ← a DIV 10
  EndWhile
  If p = b then
    Return True
  Else
    Return False
  EndIf
EndAlgorithm
```

Algoritmul  $ceFace(a, b)$  returnează *True* dacă și numai dacă:

A. numerele  $a$  și  $b$  sunt egale  
 B.  $a$  și  $b$  sunt numere palindrom  
 C. numărul  $a$  este oglinditul numărului  $b$   
 D. ultima cifră a lui  $a$  este egală cu ultima cifră a lui  $b$

3. Se consideră algoritmul  $ceFace(n)$ , unde  $n$  este un număr natural ( $1 \leq n \leq 10^3$ ). Operatorul „ $\wedge\wedge\wedge$ ” reprezintă împărțirea reală, de exemplu:  $3 / 2 = 1.5$ .

```
Algorithm ceFace(n):
  s ← 0
  For i ← 1, n execute
    p ← (i + 1) * (i + 2)
    s ← s + (i / p)
  EndFor
  Return s
EndAlgorithm
```

Precizați expresia a cărei valoare este returnată de algoritm.

- A.  $\frac{1}{1} + \frac{1}{1+2} + \dots + \frac{1}{1+2+\dots+n}$
- B.  $\frac{1}{2+3} + \frac{2}{3+4} + \dots + \frac{n}{(n+1)+(n+2)}$
- C.  $\frac{1}{1} + \frac{1}{1+2} + \dots + \frac{1}{1+2+\dots+n}$
- D.  $\frac{1}{2+3} + \frac{2}{3+4} + \dots + \frac{n-1}{n+(n+1)}$

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

```
Algorithm f(n, x):
  k ← 0
  For i ← 1, n - 1 execute
    If k = 0 then
      If x[i] = x[i + 1] then
        Return False
      EndIf
      If x[i] < x[i + 1] then
        k ← 1
      EndIf
    EndIf
    If k = 1 then
      If x[i] ≥ x[i + 1] then
        Return False
      EndIf
    EndIf
  EndFor
  If x[n - 1] ≥ x[n] then
    Return False
  EndIf
  Return True
EndAlgorithm
```

Pentru care din următoarele apeluri algoritmul va returna *True*?

A.  $f(6, [1000, 512, 23, 22, 1, 2])$   
 B.  $f(6, [6, 4, 1, 1, 2, 3])$   
 C.  $f(8, [3000, 2538, 799, 424, 255, 256, 299, 1001])$   
 D.  $f(3, [3, 2, 1])$

5. Se dă algoritmul  $calcul(a, b, c, d)$ , unde  $a, b, c, d$  sunt numere naturale nenule ( $1 \leq a, b, c, d \leq 100$ ).

```
Algorithm calcul(a, b, c, d):
  x ← a * b
  y ← c * d
  While y ≠ 0 execute
    z ← x MOD y
    x ← y
    y ← z
  EndWhile
  Return x
EndAlgorithm
```

Care dintre următoarele afirmații sunt adevărate?

- A. Algoritmul returnează cel mai mare divizor comun al numerelor  $a, b, c, d$ .  
 ✓ B. Algoritmul returnează cel mai mare divizor comun al numerelor  $a * b$  și  $c * d$ .  
 ✓ C. Algoritmul returnează cel mai mic multiplu comun al numerelor  $a, b, c, d$ .  
 D. Algoritmul returnează cel mai mic multiplu comun al numerelor  $a * b$  și  $c * d$ .

6. Se consideră algoritmul  $p(na, a, nb, b)$ , unde  $na$  și  $nb$  sunt numere naturale ( $0 \leq na, nb \leq 10^4$ ),  $a$  și  $b$  sunt vectori cu  $na$ , respectiv  $nb$  numere naturale ( $a[1], a[2], \dots, a[na]$ ,  $1 \leq a[i] \leq 10^4$ , pentru  $i = 1, 2, \dots, na$  și  $b[1], b[2], \dots, b[nb]$ ,  $1 \leq b[i] \leq 10^4$ , pentru  $i = 1, 2, \dots, nb$ ). Variabila locală  $c$  este un vector.

```
Algorithm p(na, a, nb, b):
  i ← 1
  j ← 1
  nc ← 0
  While i ≤ na AND j ≤ nb execute
    nc ← nc + 1
    If a[i] < b[j] then
      c[nc] ← a[i]
      i ← i + 1
    Else
      c[nc] ← b[j]
      j ← j + 1
    EndIf
  EndWhile
  Return nc
EndAlgorithm
```

Care dintre următoarele afirmații sunt adevărate?

- ✓ A. Dacă  $na = 0$  și  $nb = 0$ , atunci valoarea returnată prin  $nc$  este egală cu 0.  
 ✓ B. Dacă elementele din  $a$  și  $b$  sunt sortate crescător, atunci elementele depuse în  $c$  sunt sortate crescător.  
 C. Valoarea returnată prin  $nc$  este întotdeauna egală cu  $na + nb$ .  
 ✓ D. Dacă  $na, nb > 0$  și cel mai mare element din  $a$  este mai mic decât toate elementele din  $b$ , atunci  $c$  va avea exact aceleași elemente ca și  $a$ .

7. Se dă algoritmul  $\text{suma}(n, a, m, b)$ , unde  $n$  și  $m$  sunt numere naturale ( $1 \leq n, m \leq 10^5$ ), iar  $a$  și  $b$  sunt două şiruri ordonate crescător cu  $n$ , respectiv  $m$  elemente numere naturale ( $a[1], a[2], \dots, a[n]$  și  $b[1], b[2], \dots, b[m]$ ):

```
Algorithm suma(n, a, m, b):
    s ← 0
    For i ← 1, n, 2 execute
        j ← 1
        While j ≤ a[i] AND j ≤ m execute
            s ← s + b[j]
            j ← j + 1
        EndWhile
    EndFor
    Return s
EndAlgorithm
```

- Ce valoare va returna algoritmul, dacă  $n = 4$ ,  $a = [1, 3, 4, 7]$ ,  $m = 6$  și  $b = [2, 4, 6, 8, 10, 12]$ ?
- 42
  - 22 ✓
  - 20
  - Nu se poate determina ce valoare va returna

8. Se consideră algoritmul  $\text{verifica}(n, p_1, p_2)$ , unde  $n, p_1$  și  $p_2$  sunt numere naturale ( $1 \leq n, p_1, p_2 \leq 10^6$ ):

```
Algorithm verifica(n, p1, p2):
    bt ← (p1 + p2) DIV 2
    If p1 > p2 then
        Return False
    EndIf
    If bt * bt = n then
        Return True
    EndIf
    If bt * bt > n then
        Return verifica(n, p1, bt - 1)
    EndIf
    Return verifica(n, bt + 1, p2)
EndAlgorithm
```

- Care dintre următoarele afirmații sunt adevărate?
- Dacă numerele  $p_1, p_2$  și  $n$  sunt prime între ele, atunci apelul  $\text{verifica}(n, p_1, p_2)$  returnează *True*.
  - Algoritmul folosește metoda căutării binare și dacă numărul  $n$  este prim, apelul  $\text{verifica}(n, 1, n)$  returnează *True*.
  - Pentru apelul  $\text{verifica}(n, 1, n)$  algoritmul returnează *True* dacă și numai dacă numărul  $n$  este pătrat perfect.
  - Dacă  $p_1 \leq n \leq p_2$  și în intervalele  $[p_1, n]$  și  $[n, p_2]$  există cel puțin câte un pătrat perfect, atunci apelul  $\text{verifica}(n, p_1, p_2)$  returnează *True*.

9. Se consideră algoritmul  $\text{ceFace}(n)$ , unde  $n$  este un număr natural ( $1 \leq n \leq 3000$ ).

```
Algorithm ceFace(n):
    s ← 0
    i ← 1
    While s < n execute
        s ← s + i
        If s = n then
            Return True
        Else
            i ← i + 2
        EndIf
    EndWhile
    Return False
EndAlgorithm
```

- Care dintre următoarele afirmații sunt adevărate?
- Dacă  $n = 36$ , algoritmul returnează *True*.
  - Dacă  $n$  este egal cu o sumă de numere impare consecutive începând de la 1, algoritmul returnează *True*.
  - Dacă  $n$  este pătrat perfect, algoritmul returnează *True*, altfel returnează *False*.
  - Dacă  $n = 64$ , algoritmul returnează *False*.

10. Se consideră algoritmul  $\text{ceFace}(a)$ , unde  $a$  este număr natural ( $1 \leq a \leq 10^4$ ).

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

- Precizați efectul algoritmului.
- Algoritmul returnează cel mai mic palindrom mai mare sau egal cu  $a$ .
  - Algoritmul returnează cel mai mare palindrom mai mic sau egal cu  $a$ .
  - Algoritmul returnează cel mai mic palindrom mai mare decât  $a$ .
  - Algoritmul returnează cel mai mic număr par mai mare decât  $a$ .

11. 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 ← 2
    x ← 0
    If v[1] MOD 2 ≠ 0 then
        Return False
    EndIf
    While i ≤ n execute
        If x = 0 AND v[i] MOD 2 = 0 then
            Return False
        Else
            If x = 1 AND v[i] MOD 2 = 1 then
                Return False
            Else
                i ← i + 1
                x ← (x + 1) MOD 2
            EndIf
        EndWhile
        Return True
EndAlgorithm
```

- În care din următoarele situații algoritmul returnează *True*?
- Dacă vectorul  $v$  este format din valorile  $[2, 3, 10, 7, 20, 5, 18]$  și  $n = 7$
  - Dacă vectorul  $v$  are valori după următorul model: impar, par, impar, par...
  - Dacă vectorul  $v$  este format din valorile  $[3, 8, 17, 20, 15, 10]$  și  $n = 6$
  - Dacă vectorul  $v$  are valori după următorul model: par, impar, par, impar...

✓ 1 2 3  
✓ 1 2 3

Precizați care dintre următoarele afirmații sunt adevărate:

- Algoritmul returnează suma tuturor elementelor din vectorul  $a$ .
- Algoritmul returnează suma subsecvenței de lungime maximă care conține doar elemente pozitive din vectorul  $a$ .
- Algoritmul returnează suma tuturor elementelor pozitive din vectorul  $a$ .
- Algoritmul returnează suma unei subsecvențe cu suma maximă din vectorul  $a$ .

✓ D

13. Se consideră o matrice  $A$  de numere întregi cu  $n$  linii și  $m$  coloane ( $1 \leq n, m \leq 10^4$ ). În condițiile în care  $n * m = p * q$ , dorim să redimensionăm această matrice într-o matrice  $B$  de numere întregi cu  $p$  linii și  $q$  coloane ( $1 \leq p, q \leq 10^4$ ), conform exemplului de mai jos, unde  $n = 4, m = 6, p = 3$  și  $q = 8$ . Linile și coloanele sunt numerotate începând de la 1.

A: 1	1	2	3	4	5	6	7	8
2	7	8	9	10	11	12		
3	13	14	15	16	17	18		
4	19	20	21	22	23	24		

B: 1	1	2	3	4	5	6	7	8
2	9	10	11	12	13	14	15	16
3	17	18	19	20	21	22	23	24
4								

Care din următoarele variante prezintă un algoritm care pentru perechea de numere naturale  $i$  și  $j$  ( $1 \leq i \leq n, 1 \leq j \leq m$ ) reprezentând indicii în matricea  $A$  va returna perechea de indici din matricea  $B$  corespunzătoare valorii  $A[i][j]$ ?

A: Algorithm reshape(i, j, n, m, p, q):
 Return (i \* m + j) DIV q, (i \* m + j) MOD q
EndAlgorithm

B: Algorithm reshape(i, j, n, m, p, q):
 i ← i - 1
 j ← j - 1
 Return (i \* m + j) DIV q, (i \* m + j) MOD q
EndAlgorithm

C: Algorithm reshape(i, j, n, m, p, q):
 i ← i - 1
 j ← j - 1
 Return (i \* m + j) DIV q + 1,
 (i \* m + j) MOD q + 1
EndAlgorithm

D: Algorithm reshape(i, j, n, m, p, q):
 Return (i \* m + j - 1) DIV q + 1,
 (i \* m + j - 1) MOD q + 1
EndAlgorithm

14. Se consideră algoritmul  $\text{ceFace}(n, m)$ , unde  $n$  este număr natural ( $1 \leq n \leq 10^4$ ), iar  $m$  este o matrice cu  $n$  linii și  $n$  coloane, iar elementele sunt numere naturale ( $m[1][1], \dots, m[1][n], m[2][1], \dots, m[2][n], \dots, m[n][1], \dots, m[n][n]$ ). Considerăm că elementele matricei  $m$  sunt inițial egale cu 0.

```
Algorithm ceFace( $n, m$ ):
    a  $\leftarrow 0$ 
    b  $\leftarrow 1$ 
    For  $j \leftarrow 1, n$  execute
        i  $\leftarrow 1$ 
        While  $i + j \leq n - 1$  execute
            If  $(i \bmod 2 = 1) \text{ AND } (j \bmod 2 = 1)$  then
                 $m[i][j] \leftarrow b$ 
                c  $\leftarrow a + b$ 
                a  $\leftarrow b$ 
                b  $\leftarrow c$ 
            EndIf
            i  $\leftarrow i + 1$ 
        EndWhile
    EndFor
EndAlgorithm
```

15. Algoritmi de mai jos prelucrează un vector  $x$  ordonat crescător, având  $n$  elemente numere naturale ( $1 \leq n \leq 10^4$ ,  $x[1], x[2], \dots, x[n]$ ). Parametrii  $first$  și  $last$  sunt numere naturale ( $1 \leq first \leq last \leq n$ ).

Alegeți algoritmi care au complexitatea timp cea mai scăzută, dacă se apelează sub forma  $A(x, 1, n, n)$ .

A. *log n*

```
Algorithm A( $x, first, last, n$ ):
    If  $first > last$  then
        Return 0
    EndIf
    m  $\leftarrow (first + last) \bmod 2$ 
    If  $x[m] = n$  then
        Return m
    Else
        If  $x[m] > n$  then
            Return A( $x, first, m - 1, n$ )
        Else
            If  $x[m] < n$  then
                Return A( $x, m + 1, last, n$ )
            EndIf
        EndIf
    EndIf
EndAlgorithm
```

B. *log n*

```
Algorithm A( $x, first, last, n$ ):
    While  $first < last$  execute
        m  $\leftarrow (first + last) \bmod 2$ 
        If  $x[m] = n$  then
            Return m
        Else
            If  $x[m] > n$  then
                last  $\leftarrow m - 1$ 
            Else
                first  $\leftarrow m + 1$ 
            EndIf
        EndIf
    EndWhile
Return 0
EndAlgorithm
```

D. *log<sub>3</sub> n*

```
Algorithm A( $x, first, last, n$ ):
    For  $i \leftarrow first, last$  execute
        If  $x[i] = n$  then
             $x[i] \leftarrow 3 * n$ 
        EndIf
    EndFor
EndAlgorithm
```

16. Andrei se joacă cu următorul algoritm, unde  $n$  și  $m$  sunt numere naturale nenule ( $1 \leq n, m \leq 10^4$ ). Algoritmul  $\text{abs}(x)$  returnează valoarea absolută a lui  $x$ .

```
Algorithm problema( $n, m$ ):
    b  $\leftarrow \text{abs}(m - n)$ 
    c  $\leftarrow n - m$ 
    If  $b - c = 0$  then
        a  $\leftarrow n \bmod m$ 
    Else
        a  $\leftarrow (m + 2) \bmod n$ 
    EndIf
    Return a
EndAlgorithm
```

El observă că indiferent de valoarea variabilei  $n$  corespunzătoare specificației, există cel puțin două valori ale lui  $m$  în cazul cărora algoritmul  $\text{problema}(n, m)$  returnează 0. Care sunt aceste valori ale lui  $m$ ?

- A. 1 și  $n$
- B. 1 și  $n + 2$
- C.  $n$  și  $n + 2$
- D. 1 și  $n - 2$

5

Care dintre următoarele afirmații sunt FALSE?

- A. Dacă  $n = 11$ , valoarea lui  $m[6][4]$  este 21
- B. Dacă  $n = 7$ , valoarea lui  $m[3][5]$  este 4
- C. Dacă  $n = 10$ , valoarea lui  $m[6][4]$  este 21
- D. Dacă  $n = 7$ , valoarea maximă din matrice este 8

17. Un elev dorește să genereze, folosind metoda backtracking, toate numerele impare cu câte trei cifre, cifre care iau valori din vectorul  $[4, 3, 8, 5, 7, 6]$ , în ordinea dată. Știind că primele 5 numere generate sunt, în această ordine: 443, 445, 447, 433, 435, care va fi cel de-al zecelea număr generat?

A. 487      B. 453      C. 457      D. 455

18. Se consideră algoritmul  $f(k, n, x)$ , unde  $k, n$  sunt numere naturale ( $1 \leq k, n \leq 10^3$ ) și  $x$  este un vector de  $n$  numere naturale ( $x[1], x[2], \dots, x[n]$ ,  $1 \leq x[i] \leq 10^4$ , pentru  $i = 1, 2, \dots, n$ ).

**Algorithm** f( $k, n, x$ ):

```
If  $n = 0$  then
    Return 0
Else
    d  $\leftarrow 0$ 
    For  $i \leftarrow 2, x[n] \bmod 2$  execute
        If  $(x[n] \bmod i) = 0$  then
            d  $\leftarrow d + 1$ 
        EndIf
    EndFor
    If  $d = k$  then
        Return 1 + f( $k, n - 1, x$ )
    Else
        Return f( $k, n - 1, x$ )
    EndIf
EndIf
EndAlgorithm
```

Care dintre următoarele afirmații sunt adevărate?

- A. Pentru  $x = [4, 9, 26, 121]$  rezultatul apelului  $f(1, 4, x)$  va fi 3.
- B. Pentru  $x = [4, 8, 6, 144]$  rezultatul apelului  $f(2, 4, x)$  va fi 3.
- C. Pentru  $x = [4, 9, 25, 144]$  rezultatul apelului  $f(1, 4, x)$  va fi 3.
- D. Pentru  $x = [8, 27, 25, 121]$  rezultatul apelului  $f(2, 4, x)$  va fi 3.

19. Fiș algoritmul  $\text{check}(n)$ , unde  $n$  este număr natural ( $1 \leq n \leq 10^5$ ).

```
Algorithm check( $n$ ):
    While  $n > 0$  execute
        If  $n \bmod 3 > 1$  then
            Return False
        EndIf
        n  $\leftarrow n \bmod 3$ 
    EndWhile
    Return True
EndAlgorithm
```

Precizați efectul algoritmului.

- A. Algoritmul returnează *True* dacă  $n$  este o putere a lui 3 și *False* în caz contrar.
- B. Algoritmul returnează *True* dacă scrierea în baza 3 a lui  $n$  conține doar cifre 0 și 1 și *False* în caz contrar.
- C. Algoritmul returnează *True* dacă  $n$  poate fi scris ca o putere a lui 3 sau ca sumă de puteri distincte ale lui 3 și *False* în caz contrar.
- D. Algoritmul returnează *True* dacă scrierea în baza 3 a lui  $n$  conține doar cifra 2 și *False* în caz contrar.

20. Un eveniment trebuie să aibă loc într-o anumită sală I, dar trebuie mutat în sala II, unde numerotarea scaunelor diferă. În ambele săli există  $L$  rânduri de scaune ( $2 \leq L \leq 50$ ), fiecare rând fiind împărțit la mijloc de un culoar și având  $K$  scaune ( $2 \leq K \leq 50$ ) în fiecare parte a culoarului (deci, sala conține în total  $2 * K * L$  scaune).

În sala I fiecare loc este identificat printr-un singur număr. Locurile din stânga culoarului au numere pare, iar numerotarea scaunelor începe pe rândul din fața scenei. Deci scaunele din primul rând au numerele (pornind dinspre culoar spre marginea sălii) 2, 4, 6 etc. După ce toate scaunele de pe un rând au fost numerotate, pe rândul următor se continuă numerotarea, reîncepând cu scaunul de lângă culoar cu următorul număr par. Locurile din partea dreaptă a culoarului sunt numerotate la fel, dar folosind numere impare. Deci scaunele din primul rând au numerele (pornind dinspre culoar spre marginea sălii) 1, 3, 5 etc.

În sala II fiecare loc este identificat prin trei valori. Numărul rândului (o valoare între 1 și  $L$ , rândul 1 fiind cel din fața scenei), direcția locului față de culoar (valoarea "stânga" sau "dreapta") și numărul scaunului în cadrul rândului (o valoare între 1 și  $K$ , scaunul 1 fiind cel de lângă culoar).

Din cauza mutării spectacolului, locurile de pe bilete din sala I (reprazentate prin *rând, loc, direcție*) trebuie transformate în locuri valabile în sala II (reprazentate prin *rând, loc, direcție*).

Care dintre următorii algoritmi, având ca date de intrare  $L, K, nrLoc$  conform enunțului execută în mod corect transformarea? O transformare este corectă dacă fiecare spectator va avea un loc unic în sala II.

```

Algorithm transforma(L, K, nrLoc):
    If nrLoc MOD 2 = 1 then
        directie ← "dreapta"
        nrLoc ← nrLoc + 1
    Else
        directie ← "stanga"
    EndIf
    If nrLoc MOD (2 * K) = 0 then
        rand ← nrLoc DIV (2 * K)
    Else
        rand ← nrLoc DIV (2 * K) + 1
    EndIf
    loc ← (nrLoc - (rand - 1) * 2 * K) DIV 2
    Return rand, loc, directie
EndAlgorithm

```

```

C.
Algorithm transforma(L, K, nrLoc):
    If nrLoc MOD 2 = 1 then
        directie ← "dreapta"
        nrLoc ← nrLoc + 1
    Else
        directie ← "stanga"
    EndIf
    rand ← nrLoc DIV (2 * K) + 1
    loc ← (nrLoc - (rand - 1) * 2 * K) DIV 2
    Return rand, loc, directie
EndAlgorithm

```

21. Se consideră algoritmul  $p(x, n, k, \text{final})$ , unde  $x$  este un vector de  $n + 1$  numere naturale ( $x[0], x[1], x[2], \dots, x[n]$ ). Inițial  $x[i] = 0$ , pentru  $i = 0, 1, 2, \dots, n$ . Variabilele  $n$  și  $k$  sunt numere naturale nenule ( $1 \leq n, k \leq 20$ ), iar  $\text{final}$  este de tip boolean. Algoritmul  $\text{Afis}(x, 1, n)$  afișează elementele  $x[1], x[2], \dots, x[n]$ .

```

Algorithm p(x, n, k, final):
    While final = False execute
        While x[k] < n execute
            x[k] ← x[k] + 1
            If OK(x, k) = True then
                If k = n then
                    Afis(x, 1, n)
                Else
                    k ← k + 1
                    x[k] ← 0
                EndIf
            EndWhile
        EndWhile
    EndAlgorithm

```

// aici trebuie completat algoritmul

```

Algorithm transforma(L, K, nrLoc):
    If nrLoc MOD 2 = 1 then
        directie ← "dreapta"
    Else
        directie ← "stanga"
    EndIf
    If nrLoc MOD (2 * K) = 0 then
        rand ← nrLoc DIV (2 * K)
    Else
        rand ← nrLoc DIV (2 * K) + 1
    EndIf
    loc ← (nrLoc - (rand - 1) * 2 * K) DIV 2
    Return rand, loc, directie
EndAlgorithm

```

D.

```

Algorithm transforma(L, K, nrLoc):
    If nrLoc MOD 2 = 1 then
        directie ← "dreapta"
        nrLoc ← nrLoc + 1
    Else
        directie ← "stanga"
    EndIf
    If nrLoc MOD (2 * K) = 0 then
        rand ← nrLoc DIV (2 * K)
    Else
        rand ← nrLoc DIV (2 * K) + 1
    EndIf
    loc ← (nrLoc - (rand - 1) * 2 * K) DIV 2 + 1
    Return rand, loc, directie
EndAlgorithm

```

22. Se dau algoritmii problema( $n$ ) și calcul( $a, b$ ), unde  $n, a, b$  sunt numere naturale ( $0 \leq n, a, b \leq 9$ ).

```

Algorithm problema(n):
    rezultat ← 0
    For k ← 0, n execute
        For j ← 0, n execute
            For p ← 0, n execute
                If p MOD 2 = 0 then
                    rezultat ← rezultat + 1
                EndIf
            EndFor
        EndFor
    EndFor
    Return rezultat
EndAlgorithm

```

Care din următoarele afirmații sunt adevărate?

- A. În urma apelului  $\text{calcul}(1, 8)$  se afișează 1095.
- B. În urma apelului  $\text{calcul}(1, 8)$  se afișează 1094.
- C. În urma apelului  $\text{calcul}(0, 9)$  se afișează 1095.
- D. În urma apelului  $\text{calcul}(0, 9)$  se afișează 1595.

23. Se consideră algoritmul  $\text{checkAcc}(n, f, w, lw)$ , unde  $n$  este un număr natural nenul ( $1 \leq n \leq 10^4$ ),  $f$  este număr natural,  $w$  este un sir de  $lw$  ( $1 \leq lw \leq 10^4$ ) numere naturale ( $w[1], w[2], \dots, w[lw]$ , unde  $0 \leq w[p] \leq 10^4$ , pentru  $p = 1, 2, \dots, lw$ ). Algoritmul  $\text{checkAcc}(n, f, w, lw)$  apelează algoritmul  $t(i, j, k)$ , unde  $i, j$  și  $k$  sunt numere naturale. Algoritmul  $t(i, j, k)$  returnează rezultat boolean.

```

Algorithm checkAcc(n, f, w, lw):
    acc ← True
    If lw = 0 AND f ≠ 1 then
        acc ← False
    Else
        index ← 1
        q ← 1
        While (acc = True) AND (index ≤ lw) execute
            crt ← 1
            changed ← False
            While (changed = False) AND (crt ≤ n) execute
                If t(q, w[index], crt) then
                    q ← crt
                    changed ← True
                Else
                    crt ← crt + 1
                EndIf
            EndWhile
            If changed = False then
                acc ← False
            Else
                index ← index + 1
            EndIf
        EndWhile
        If (index > lw) AND (acc = True) AND (q ≠ f) then
            acc ← False
        EndIf
    EndIf
    Return acc
EndAlgorithm

```

În care dintre situațiile de mai jos algoritmul  $\text{checkAcc}(2, f, w, lw)$  va returna  $\text{True}$ , știind că algoritmul  $t(i, j, k)$  returnează  $\text{True}$  în cazurile din tabel, altfel returnează  $\text{False}$ ?

i	j	k
1	0	1
1	1	2
2	1	2

- A.  $w = [0, 0, 1, 1], lw = 4$  și  $f = 1$
- B.  $w = [1, 1, 1, 0], lw = 4$  și  $f = 2$
- C.  $w = [0, 0, 1, 1], lw = 4$  și  $f = 2$
- D.  $w = [0, 0, 0, 0], lw = 4$  și  $f = 1$

Cu ce se poate face să se completeze algoritmul, astfel încât în urma apelului  $p(x, n, 1, \text{False})$  să se afișeze toate permutările de ordin  $n$ , fiecare o singură dată.

```

Algorithm OK(x, k):
    For i ← 1, k - 1 execute
        If x[k] = x[i] then
            Return False
        EndIf
    EndFor
    Return True
EndAlgorithm

```

Cu ce se poate face să se completeze algoritmul, astfel încât în urma apelului  $p(x, n, 1, \text{False})$  să se afișeze toate permutările de ordin  $n$ , fiecare o singură dată.

A. If  $k > 1$  then  
     $k ← k - 1$   
Else  
    final ← True  
EndIf

B. If  $k > 0$  then  
     $k ← k - 1$   
Else  
    final ← True  
EndIf

C. final ← True

D. If  $k > 1$  then  
     $k ← k - 1$   
    final ← True  
EndIf

24. Se consideră vectorul de cifre  $a = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$ . Cu scopul de a afișa elementele vectorului  $a$  într-o altă ordine, se construiește vectorul  $b$  (inițial vid). La fiecare pas, se poate alege una din următoarele două operații:

- Adaugă – se adaugă primul element din vectorul  $a$  la finalul vectorului  $b$  și se elimină din vectorul  $a$ .
- Sterge – se afișează, apoi se sterge ultimul element din vectorul  $b$ .

Observații:

- Elementele vectorului  $a$  se prelucrează în ordinea dată.
- Nu se poate folosi operația Adaugă dacă vectorul  $a$  este vid și nu se poate folosi operația Sterge, dacă vectorul  $b$  este vid.
- Prelucrarea se termină când vectorii  $a$  și  $b$  sunt vizi.

Respectând regulile de mai sus, în ce ordine NU pot fi afișate cifrele?

- A. 0 1 2 3 4 5 6 7 8 9 ✓ adaug sterg ...
- C. 2 4 6 5 3 7 0 1 9 8
- B. 9 8 7 6 5 4 3 2 1 0 adaug sterg ...
- D. 2 3 1 4 5 0 8 9 7 6