

Concurs de admitere – 19 iulie 2022  
Proba scrisă la Informatică

NOTĂ IMPORTANTĂ:

În lipsa altor precizări, presupunem că toate operațiile aritmetice se efectuează pe tipuri de date nelimitate (nu există *overflow / underflow*).

De asemenea, numerotarea indicilor tuturor sirurilor/vectorilor începe de la 1.

1. Se consideră algoritmul `ceFace(a, b)`, unde  $a$  și  $b$  sunt numere naturale ( $1 \leq a, b \leq 10000$  la momentul apelului).

```
Algorithm ceFace(a, b):
    While (a MOD 10 = b MOD 10) AND (a ≠ 0) AND (b ≠ 0) execute
        a ← a DIV 10
        b ← b DIV 10
    EndWhile
    If ((a = 0) AND (b = 0)) then
        return True
    else
        return False
    EndIf
EndAlgorithm
```

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

- A. numerele  $a$  și  $b$  au același număr de cifre
- B.  $a$  și  $b$  sunt egale
- C.  $a$  și  $b$  sunt formate din aceleași cifre, dar așezate în altă ordine
- D. ultima cifră a lui  $a$  este egală cu ultima cifră a lui  $b$

2. Se consideră algoritmul `f(a, n)` unde  $n$  este număr natural nenul ( $2 \leq n \leq 10000$ ) și  $a$  este un vector cu  $n$  numere întregi ( $a[1], a[2], \dots, a[n]$ ,  $-100 \leq a[i] \leq 100$ , pentru  $i = 1, 2, \dots, n$ ). Variabila locală  $b$  este vector.

```
Algorithm f(a, n):
    i ← 2
    b[1] ← a[1]
    While i ≤ n execute
        b[i] ← b[i - 1] + a[i]
        i ← i + 1
    EndWhile
    return b[n]
EndAlgorithm
```

Precizați care dintre următoarele afirmații sunt adevărate:

- A. Algoritmul returnează suma tuturor elementelor din vectorul  $a$ .
- B. Algoritmul returnează suma ultimelor două elemente din vectorul  $a$ .
- C. Algoritmul returnează ultimul element din vectorul  $a$ .
- D. Algoritmul returnează suma ultimelor  $n - 1$  elemente din vectorul  $a$ .

3. Care dintre algoritmii următori returnează numărul factorilor primi distincți ai unui număr natural  $n$  dat ( $5 < n < 10^5$  la momentul apelului).

A.

```
// Vectorul prime are lungimea n
// prime[i] are valoarea True, dacă
// numărul i este prim și False altfel
Algorithm nrFactoriPrimi_A(n, prime):
    d ← 2
    nr ← 0
    p ← 0
    While n > 0 execute
        While n MOD d = 0 execute
            p ← p + 1
            n ← n DIV d
        EndWhile
        If p ≠ 0 then
            nr ← nr + 1
        EndIf
        d ← d + 1
    While prime[d] = False execute
        d ← d + 1
    EndWhile
    p ← 0
EndWhile
return nr
EndAlgorithm
```

B.

```
Algorithm nrFactoriPrimi_B(n):
    d ← 2
    nr ← 0
    While n > 1 execute
        p ← 0
        While n MOD d = 0 execute
            p ← p + 1
            n ← n DIV d
        EndWhile
        If p > 0 then
            nr ← nr + 1
        EndIf
        If d = 2 then
            d ← d + 1
        else
            d ← d + 2
        EndIf
    EndWhile
    return nr
EndAlgorithm
```

C.

```
Algorithm nrFactoriPrimi_C(n):
    nr ← 0
    For d ← 2, n execute
        If n MOD d = 0 then
            nr ← nr + 1
        EndIf
        While n MOD d = 0 execute
            n ← n DIV d
        EndWhile
    EndFor
    return nr
EndAlgorithm
```

D.

```
Algorithm nrFactoriPrimi_D(n):
    nr ← 0
    d ← 2
    While d * d ≤ n execute
        If n MOD d = 0 then
            nr ← nr + 1
        EndIf
        While n MOD d = 0 execute
            n ← n DIV d
        EndWhile
        d ← d + 1
    EndWhile
    return nr
EndAlgorithm
```

4. Se consideră algoritmul `ceFace(n, m)`, unde  $n$  este număr natural ( $0 \leq n \leq 1000$ ) cu ultima cifră diferită de 0.

```
Algorithm ceFace(n, m):
    If n = 0 then
        return m
    else
        return ceFace(n DIV 10, m * 10 + n MOD 10)
    EndIf
EndAlgorithm
```

Care este rezultatul apelului `ceFace(n, 0)`?

- A. 0 (indiferent de valoarea lui  $n$ )
- B.  $n$  (indiferent de valoarea lui  $n$ )
- C. Suma cifrelor numărului  $n$
- D. Oglinditul numărului  $n$

5. Se consideră algoritmul  $f(x, n)$  unde  $n$  este număr natural ( $2 \leq n \leq 10000$ ), iar  $x$  este un sir de  $n$  numere naturale ( $x[1], x[2], \dots, x[n]$ ,  $1 \leq x[i] \leq 10000$ , pentru  $i = 1, 2, \dots, n$ ).

```

Algorithm f(x, n):
    For i = 1, n - 1 execute
        If x[i] = x[i + 1] then
            return False
        EndIf
    EndFor
    return True
EndAlgorithm

```

Precizați care dintre următoarele afirmații sunt adevărate:

- A. Algoritmul returnează *False* dacă două elemente oarecare din sirul  $x$  sunt distincte.
- B. Algoritmul returnează *False* dacă două elemente oarecare din sirul  $x$  sunt egale.
- C. Algoritmul returnează *False* dacă două elemente consecutive din sirul  $x$  sunt egale.
- D. Algoritmul returnează *False* dacă primele două elemente din sirul  $x$  sunt egale.

6. Se consideră algoritmul  $f(x, n)$  unde  $x$  și  $n$  sunt numere naturale ( $0 \leq n \leq 10000$ ,  $0 < x \leq 10000$ ).

```

1. Algorithm f(x, n):
2.     If n = 0 then
3.         return 1
4.     EndIf
5.     m ← n DIV 2
6.     p ← f(x, m)
7.     If n MOD 2 = 0 then
8.         return p * p
9.     EndIf
10.    return x * p * p
11.EndAlgorithm

```

*ridicare la putere  
logaritmică*

Care dintre următoarele afirmații sunt adevărate?

- (A) Algoritmul returnează  $x$  la puterea  $n$ .
- B. Dacă pe linia 7, în loc de  $n \text{ MOD } 2$  ar fi  $m \text{ MOD } 2$ , atunci algoritmul ar returna  $x$  la puterea  $n$ .
- C. Din cauza autoapelului de pe linia 6, liniile 7, 8, 9, 10 nu se vor executa niciodată.
- D. Algoritmul returnează 1 dacă  $n$  este număr par sau  $x$  dacă  $n$  este număr impar.

7. Considerând că toate operațiile de înmulțire și împărțire se realizează în timp constant, ce putem spune despre complexitatea timp a algoritmului din enunțul subiectului 6?

- A. Complexitatea timp depinde de parametrii  $x$  și  $n$ .
- (B) Complexitatea timp nu depinde de parametrul  $x$ .
- C. Complexitatea timp este  $O(\log \log n)$ .
- D. Complexitatea timp este logaritmică în raport cu parametrul  $n$  ( $O(\log n)$ ).

8. Se consideră algoritmul  $afisare(n)$ , unde  $n$  este număr natural ( $1 \leq n \leq 10000$ ).

```

Algorithm afisare(n):
    If n ≤ 4000 then
        Write n, " "
        afisare(2 * n)
        Write n, " "
    EndIf
EndAlgorithm

```

Ce se afișează pentru apelul  $afisare(1000)$ ?

- A. 1000 2000 4000
- (B) 1000 2000 4000 4000 2000 1000
- C. 1000 2000 4000 2000 1000
- D. 1000 2000 2000 1000

9. Care ar putea fi elementele unui vector astfel încât, aplicând metoda căutării binare pentru valoarea 36, aceasta să fie comparată succesiv cu valorile 12, 24, 36, 50:

- (A) [2, 4, 7, 12, 24, 36, 50]
- (B) [2, 4, 8, 9, 12, 16, 20, 24, 36, 67]
- (C) [4, 8, 9, 12, 16, 24, 36]
- (D) [12, 24, 36, 42, 54, 66]

10. Care dintre următoarele expresii sunt echivalente cu  $x \text{ MOD } y$  pentru toate numerele strict pozitive  $x$  și  $y$  ( $0 < x, y \leq 10000$ )?

- (A)  $x \text{ DIV } y$
- (B)  $x - (y * (x \text{ DIV } y))$
- C.  $x - (x * (x \text{ DIV } y))$
- D.  $x \text{ DIV } y + y \text{ DIV } x$

11. Fie variabila  $n$  care memorează un număr natural. Care dintre expresiile de mai jos are valoarea *True* dacă și numai dacă  $n$  este divizibil cu 2 și cu 3?

- A.  $(n \text{ DIV } 2 = 0) \text{ OR } (n \text{ DIV } 3 \neq 0)$
- B.  $(n \text{ MOD } 3 = 2) \text{ OR } (n \text{ MOD } 2 = 3)$
- (C)  $(n \text{ MOD } 2 \neq 1) \text{ AND } (n \text{ MOD } 3 = 0)$
- D.  $(n \text{ MOD } 2 = 0) \text{ AND } (n \text{ MOD } 3 \neq 1)$

12. Fie variabila  $n$  care memorează un număr natural. Care dintre expresiile de mai jos are valoarea *True* dacă și numai dacă  $n$  este divizibil cu 2 și cu 3?

- (A)  $(n \text{ MOD } 2) - (n \text{ MOD } 3) = 0$
- B.  $(n \text{ MOD } 2) - (n \text{ MOD } 3) < 0$
- C.  $(n \text{ MOD } 2) + (n \text{ MOD } 3) > 0$
- (D)  $(n \text{ MOD } 2) + (n \text{ MOD } 3) = 0$

13. Se consideră algoritmul  $f(n)$ , unde  $n$  este număr natural ( $1 \leq n \leq 100$ ). Operatorul "/" reprezintă împărțirea reală (ex.  $3 / 2 = 1,5$ ). Precizați efectul algoritmului.

```

Algorithm f(n):
    s ← 0; p ← 1;
    For i ← 1, n execute
        s ← s + i
        p ← p * (1 / s)
    EndFor
    return p
EndAlgorithm

```

- A. Evaluează expresia  $1/1 * 1/2 * 1/3 * \dots * 1/n$
- B. Evaluează expresia  $1/1 * 1/(1*2) * 1/(1*2*3) * \dots * 1/(1*2*3*\dots*n)$
- (C) Evaluează expresia  $1/1 * 1/(1+2) * 1/(1+2+3) * \dots * 1/(1+2+3+\dots+n)$
- D. Evaluează expresia  $1/1 + 1/(1*2) + 1/(1*2*3) + \dots + 1/(1*2*3*\dots*n)$

14. Se consideră algoritmul  $prelucrare(s1, lung1, s2, lung2)$ , unde  $s1$  și  $s2$  sunt două siruri de caractere de lungime  $lung1$ , respectiv  $lung2$  ( $1 \leq lung1, lung2 \leq 1000$ ). Cele două siruri conțin doar caractere, având codul ASCII din intervalul  $[1, 125]$ . Variabila locală  $x$  este vector. Considerăm algoritmul  $ascii(s, i)$  care returnează codul ASCII al celui de-al  $i$ -lea caracter al sirului de caractere  $s$ .

```

Algorithm prelucrare(s1, lung1, s2, lung2):
    For i = 1, 125 execute
        x[i] ← 0
    EndFor
    For i = 1, lung1 execute
        x[ascii(s1, i)] ← x[ascii(s1, i)] + 1
    EndFor
    For i = 1, lung2 execute
        x[ascii(s2, i)] ← x[ascii(s2, i)] - 1
    EndFor
    ok ← True
    For i = 1, 125 execute
        If x[i] ≠ 0 then
            ok ← False
        EndIf
    EndFor
    return ok
EndAlgorithm

```

Precizați efectul algoritmului.

- A. Algoritmul returnează *True* dacă șirurile de caractere *s1* și *s2* au aceeași lungime și *False* în caz contrar.
- B. Algoritmul returnează *True* dacă șirurile de caractere *s1* și *s2* sunt formate din același caractere având același frecvență corespunzătoare, și *False* în caz contrar.
- C. Algoritmul returnează *True* dacă în fiecare dintre cele două șiruri de caractere *s1* și *s2* apar toate caracterele având codul ASCII din intervalul [1, 125] și *False* în caz contrar.
- D. Algoritmul returnează *True* dacă cele două șiruri de caractere *s1* și *s2* sunt formate din caractere diferite și *False* în caz contrar.

15. Care este rezultatul conversiei numărului binar 100101100111 în baza 10?

- A. 2407      B. 2408      C. 1203      D. Niciunul dintre răspunsurile A., B., C.

16. Se consideră un vector *a* cu *n* numere naturale (*a*[1], *a*[2], ..., *a*[*n*]), numărul natural *n* ( $1 \leq n \leq 10000$ ) și un număr natural *x*. Care din următoarele secvențe de cod afișează poziția cu indicele minim unde se află valoarea *x* în vectorul *a*, sau afișează -1 dacă *x* nu apare în vectorul *a*?

- A.
 

```

i ← 1
While (i ≤ n) AND (a[i] = x) execute
    i ← i + 1
EndWhile
If i ≤ n then
    Write i
else
    Write -1
EndIf

```
- B.
 

```

i ← 1
While (i ≤ n) AND (a[i] = x) execute
    i ← i + 1
EndWhile
If i = n + 1 then
    Write i
else
    Write -1
EndIf

```
- C.
 

```

i ← 1
While (i ≤ n) AND (a[i] = x) execute
    i ← i + 1
EndWhile
If i = n + 1 then
    Write i
else
    Write -1
EndIf

```
- D.
 

```

i ← 1
While (i ≤ n) AND (a[i] ≠ x) execute
    i ← i + 1
EndWhile
If i ≤ n then
    Write i
else
    Write -1
EndIf

```

17. Se consideră algoritmul *f(x)*, unde *x* este număr întreg:

```

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 ce valoare a lui *x* algoritmul va returna valoarea 4?

- A. 13369      B. 21369      *1234*      C. 4      D. 1233

18. Se consideră algoritmul *f(n, i, j)* unde *n*, *i* și *j* sunt numere naturale ( $1 \leq n, i, j \leq 10000$  la momentul apelului inițial).

```

Algorithm f(n, i, j):
    If i > j then
        Write '*'
    else
        If n MOD i = 0 then
            f(n, i - 1, j)
        else
            If n DIV i ≠ j then
                f(n, i + 1, j - 1)
                Write '0'
            else
                f(n, i + 2, j - 2)
                Write '#'
            EndIf
        EndIf
    EndIf
EndAlgorithm

```

Ce se afișează în urma execuției apelului *f(15, 3, 10)*?

- B. \*0#000
   
C. \*#0#000
   
D. \*0000000

19. Se consideră algoritmul *ceFace(n, x)*, unde *n* este număr natural ( $1 \leq n \leq 100$ ) și *x* este un vector cu *n* elemente numere naturale (*x*[1], *x*[2], ..., *x*[*n*]).

```

Algorithm ceFace(n, x):
    For i = 1, n execute
        c ← x[i]
        x[i] ← x[n - i + 1]
        x[n - i + 1] ← c
    EndFor
EndAlgorithm

```

Care va fi nouă conținutul vectorului *x* după executarea algoritmului dat dacă *n* = 6 și *x* = [5, 3, 2, 1, 1, 1]?

A. [1, 1, 2, 1, 3, 5]

B. [1, 1, 1, 2, 3, 5]

D. Niciuna dintre variantele anterioare nu este corectă.

C. [5, 3, 2, 1, 1, 1]

20. Se consideră algoritmul what(n), unde  $n$  este număr natural ( $1 \leq n \leq 1000$  la apelul inițial).

```

Algorithm what(n):
  If n = 0 then
    return True
  Endif
  If (n MOD 10 = 3) OR (n MOD 10 = 7) then
    return what(n DIV 10)
  else
    return False
  Endif
EndAlgorithm

```

Care dintre următoarele afirmații sunt adevărate?

A. Algoritmul returnează *True* dacă și numai dacă fie  $n$  este format doar din cifre de 3, fie  $n$  este format doar din cifre de 7B. Algoritmul returnează *False* dacă  $n$  conține cel puțin o cifră parăC. Algoritmul returnează *False* dacă și numai dacă  $n$  conține cel puțin o cifră  $c$  unde  $c \neq 3$  și  $c \neq 7$ D. Algoritmul returnează *True* dacă și numai dacă  $n$  nu conține nicio cifră din mulțimea {0, 1, 2, 4, 5, 6, 8, 9}21. Se consideră algoritmul calcul(x, n), unde  $x$  și  $n$  sunt numere naturale ( $1 \leq x \leq 10000$ ,  $1 \leq n \leq 10000$ ), și  $x \leq n$ .

```

Algorithm calcul(x, n):
  b ← 1
  For i ← 1, n - x execute
    b ← b * i → 6
  EndFor
  a ← b
  For i ← n - x + 1, n execute
    a ← a * i → 6 · 5
  EndFor
  return a DIV b → 6 · 5
EndAlgorithm

```

$$A_5^2 = \frac{5!}{3!} = 5 \cdot 4$$

Precizați care dintre următoarele afirmații sunt adevărate:

- A. Dacă  $x = 2$  și  $n = 5$ , atunci algoritmul returnează 10.  
 B. Algoritmul returnează numărul acelor submulțimi ale mulțimii  $\{1, 2, \dots, n\}$  care au câte  $x$  elemente.  
 C. Algoritmul returnează numărul aranjamentelor de  $n$  elemente, luate câte  $x$ .  
 D. Algoritmul returnează numărul combinărilor de  $n$  elemente, luate câte  $x$ .

22. O fermă crește găini și iepuri, fiecare găină având două picioare și fiecare iepure patru picioare. Numărul total de capete este  $n$  și numărul total de picioare al animalelor din fermă este  $m$  ( $0 \leq n, m \leq 10^4$ ). Care dintre următorii algoritmi returnează *True* și afișează toate perechile de numere posibile pentru numărul găinilor și al iepurilor din fermă, sau returnează *False* dacă nu există soluție?

(A)

```

Algorithm ferma_A(n, m):
  found = False
  For i ← 0, n execute
    j ← n - i
    If 2 * i + 4 * j = m then
      found ← True
      Write i, ', ', j
      Write newline
    EndIf
  EndFor
  return found
EndAlgorithm

```

(B)

```

Algorithm ferma_B(n, m):
  found ← False
  For i ← 0, n execute
    For j ← 0, n execute
      If 2 * i + 4 * j = m AND
          i + j = n then
        found ← True
        Write i, ', ', j
        Write newline
      EndIf
    EndFor
  EndFor
  return found
EndAlgorithm

```

(C)

```

Algorithm ferma_C(n, m):
  found ← False
  For i ← 0, n execute
    For j ← 0, n - i execute
      If 2 * i + 4 * j = m AND
          i + j = n then
        found ← True
        Write i, ', ', j
        Write newline
      EndIf
    EndFor
  EndFor
  return found
EndAlgorithm

```

(D)

```

Algorithm ferma_D(n, m):
  found ← False
  For i ← 0, n execute
    For j ← 0, i execute
      If 2 * i + 4 * j = m AND
          i + j = n then
        found ← True
        Write i, ', ', j
        Write newline
      EndIf
    EndFor
  EndFor
  return found
EndAlgorithm

```

23. Se dă un număr natural  $n$ , care poate fi scris ca produs de trei numere naturale  $a, b, c$ , ( $n = a * b * c$ ). Care dintre următoarele expresii arăta valoarea restului împărțirii lui  $n$  la numărul natural  $d$  ( $1 \leq n, a, b, c, d \leq 10000$ )?

$$n = 4 \cdot 5 \cdot 6 = 120 \quad d = 15$$

- A.  $(a \text{ MOD } d) * b * c$   
 B.  $((a \text{ MOD } d) * (b \text{ MOD } d) * (c \text{ MOD } d)) \text{ MOD } d$   
 C.  $(a \text{ MOD } d) * (b \text{ MOD } d) * (c \text{ MOD } d)$   
 D.  $(a \text{ DIV } d) * (b \text{ DIV } d) * (c \text{ DIV } d)$

24. Se consideră algoritmul det(a, n, m), unde  $a$  este un sir de  $n$  numere naturale ( $a[1], a[2], \dots, a[n]$ ) dacă  $n \geq 1$  sau sir vid dacă  $n = 0$ .  $n$  și  $m$  sunt numere naturale ( $0 \leq n \leq 100$ ,  $0 \leq m \leq 10^6$ ).

1. Algorithm det(a, n, m):  
 2. For i ← 1, n - 1 execute  
 3. For j ← i + 1, n execute  
 4. If a[i] > a[j] then  
 5. tmp ← a[i]  
 6. a[i] ← a[j]  
 7. a[j] ← tmp  
 8. EndIf  
 9. EndFor  
 10. EndFor  
 11. i ← 1  
 12. j ← n  
 13. b ← False  
 14. While i < j execute  
 15. If a[i] + a[j] = m then  
 16. b ← True  
 17. EndIf

} ord. cresc.

```

18.     If a[i] + a[j] < m then
19.         i ← i + 1
20.     else
21.         j ← j - 1
22.     EndIf
23. EndWhile
24. return b
25. EndAlgorithm

```

Precizați care dintre următoarele afirmații sunt adevărate:

- A) Algoritmul returnează *True* dacă în sirul *a* există o pereche de numere care au suma egală cu *m*.
- B) Algoritmul returnează întotdeauna *False*.
- C) Algoritmul returnează *False* dacă *n* = 0.
- D) În liniile 2, ..., 10 algoritmul sortează crescător sirul *a*.

25. Se consideră algoritmul *magic(n, a)*, unde *a* este un vector cu *n* numere naturale (*a[1], a[2], ..., a[n]*,  $1 \leq n \leq 10000$ ).

```

Algorithm magic(n, a):
    If n < 2 then
        return False
    EndIf
    For i ← 2, n execute
        If a[i - 1] = a[i] then
            return True
        EndIf
    EndFor
    return False
EndAlgorithm

```

Precizați care dintre următoarele afirmații sunt adevărate:

- A) Pentru *magic(5, [2, 5, 4, 5, 4])* algoritmul returnează *False*.
- B) Algoritmul indică dacă există duplicate în sirul *a*, dacă și numai dacă vectorul *a* este sortat crescător/descrescător.
- C) Pentru *magic(9, [1, 2, 3, 4, 4, 5, 6, 7, 9])* algoritmul returnează *True*.
- D) Pentru *magic(5, [9, 5, 5, 2, 4])* algoritmul returnează *True*.

26. Fie algoritmul *f(n, a, b, c)* unde *n* este număr natural ( $n \leq 20$ ) și *a, b, c* trei numere întregi.

```

Algorithm f(n, a, b, c):
    If n = 0 then
        return 1
    else
        return f(n - 1, a * a, b + 1, c * 2) + f(n - 1, a - 1, b * b, c + 1) + 1
    EndIf
EndAlgorithm

```

Care este rezultatul returnat la apelul *f(n, 1, 1, 2)*?

- A)  $2^{n+1} - 1 = 7$
- B.  $n = 2$
- C)  $2^0 + 2^1 + 2^2 + \dots + 2^n = 7$
- D.  $2^{n+1} = 8$

27. Se consideră algoritmii *f(n, p)* și *g(n)*, unde *n* și *p* sunt inițial numere naturale ( $1 \leq n, p \leq 10^6$  la momentul apelului initial).

```

Algorithm g(n):
    If n < 2 then
        return False
    EndIf
    i ← 2
    While i * i ≤ n execute
        If n MOD i = 0 then
            return False
        EndIf
        i ← i + 1
    EndWhile
    return True
EndAlgorithm

```

```

Algorithm f(n, p):
    If n = 0 then
        return 1
    EndIf
    If n > 0 AND n ≥ p then
        c ← 0
        If g(p) = True then
            c ← c + f(n - p, p + 1)
        EndIf
        return c + f(n, p + 1)
    EndIf
    return 0
EndAlgorithm

```

Precizați care dintre următoarele afirmații sunt adevărate:

- A) Algoritmul *g(n)* returnează *True* dacă numărul *n* este prim și *False* în caz contrar.
- B) Pentru apelul *f(n, 2)* se returnează numărul de moduri diferite în care numărul *n* poate fi scris ca sumă de cel puțin un termen de numere prime distincte în ordine strict crescătoare.
- C. Pentru apelul *f(n, 2)* se returnează suma divizorilor primi ai numărului *n*.
- D) Apelurile *f(n, 1)* și *f(n, 2)* vor returna același rezultat, oricare ar fi *n*.

28. Se consideră algoritmul *AlexB(value, n, k, p)*, unde *value* este un sir cu *n* numere naturale (*value[1], value[2], ..., value[n]*), iar *n, k* și *p* sunt numere naturale. Inițial sirul *value* are *n* elemente egale cu zero. Algoritmul *afişare(value, n)* afişează pe o linie sirul *value*.

```

Algorithm AlexB(value, n, k, p):
    p ← p + 1
    value[k] ← p
    If p = n then
        afişare(value, n)
    else
        For i ← 1, n execute
            If value[i] = 0 then
                AlexB(value, n, i, p)
            EndIf
        EndFor
    EndIf
    p ← p - 1
    value[k] ← 0
EndAlgorithm

```

Precizați sirul afișat pe a zecea linie, dacă *n* = 5 și algoritmul se apelează sub forma *AlexB(value, 5, 1, 0)*.

- A. 1 5 2 3 4
- B. 1 5 4 0 4
- C. 5 5 5 5 5
- D. 1 2 5 4 3

permute? den sare?

29. Se consideră algoritmul *f(n)* unde *n* este număr natural ( $1 \leq n \leq 10000$  la momentul apelului).

```

Algorithm f(n):
    c ← 0
    If n ≠ 0 then
        c ← c + 1
        n ← n & (n - 1) // and pe biți
        While n ≠ 0 execute
            c ← c + 1
            n ← n & (n - 1) // and pe biți
        EndWhile
    EndIf
    return c
EndAlgorithm

```

Operatorul  $\&$  este operatorul AND pe biți; tabelul de adevăr este următorul:

	0	1
0	0	0
1	0	1

Exemplu:

2 & 7 convertit în binar: 010 & 111 = 010 care este 2 în baza 10.  
6 & 1 convertit în binar: 110 & 001 = 000 care este 0 în baza 10.

Care dintre afirmațiile de mai jos NU sunt adevărate?

- A. Dacă  $n$  este o putere a lui 2, atunci  $f(n)$  returnează valoarea 1.
- B. Dacă  $n > 16$  și  $n < 32$ , atunci valoarea returnată de  $f(n)$  aparține mulțimii  $\{2, 3, 4, 5\}$ .
- C. Algoritmul returnează numărul de numere pare strict mai mici decât  $n$ .
- D. Algoritmul returnează numărul de numere impare strict mai mici decât  $n$ .

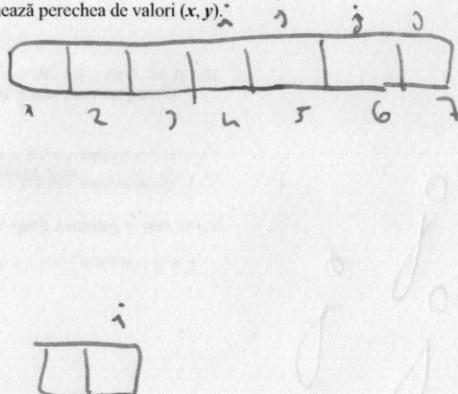
30. Se consideră algoritmul  $\text{calcul}(v, n)$ , unde  $n$  este număr natural nenul ( $1 \leq n \leq 10000$ ) și  $v$  este un sir cu  $n$  numere întregi ( $v[1], v[2], \dots, v[n]$ ). Instrucțiunea  $\text{return } x, y$  returnează perechea de valori  $(x, y)$ .

Algorithm  $\text{calcul}(v, n)$ :

```

i ← n DIV 2 + 1
j ← i + 1
k ← i
p ← j
While j ≤ n execute
    While (j ≤ n) AND (v[i] = v[j]) execute
        j ← j + 1
    EndWhile
    If j - i > p - k then
        k ← i
        p ← j
    EndIf
    i ← j
    j ← j + 1
EndWhile
If j - i > p - k then
    k ← i
    p ← j
EndIf
return p - k, k
EndAlgorithm

```



Precizați care dintre următoarele afirmații sunt adevărate:

- A. Dacă sirul are un singur element, algoritmul returnează valorile 0, -1
- B. Dacă  $n = 2$  și cele două elemente ale sirului sunt simetrice față de 0 (de ex. -5, 5), rezultatul va fi -1, 1
- C. Dacă  $n = 2$  și cele două elemente ale sirului au valori consecutive (de ex. 3, 4), se returnează întotdeauna valorile 1, 2
- D. Unul dintre numerele returnate de algoritm reprezintă lungimea celei mai lungi secvențe cu valori egale, din a doua jumătate a sirului, pentru orice  $n > 1$  număr par

$\&$	0	1
0	0	0
1	0	1