**BABEŞ-BOLYAI TUDOMÁNYEGYETEM**
**MATEMATIKA-INFORMATIKA KAR**

# Felvételi verseny – 2024 július 19
## Informatika írásbeli

FONTOS MEGJEGYZÉS:
Más pontosítások hiányában:

- Az aritmetikai műveleteket végtelen adattípusokon végezzük (nincs túlcsordulás és alulcsordulás).
- Minden vektort, mátrixot és karakterláncot 1-től sorszámozunk (indexelünk).
- Az aktuális paraméterek értékeire vonatkozó megszorítások a kezdeti hívás pillanatában érvényesek.
- Egy vektor tömbszakaszát a vektor olyan elemei alkotják, amelyek egymás utáni pozíciókon találhatók.
- Egy vektor részsorozatát a vektor olyan elemei alkotják, amelyek nem kötelezően egymás utáni pozíciókon találhatók az adott vektorban, de az elemek eredeti sorrendje megmarad.
- Ha ugyanabban a sorban több egymásutáni értékadó utasítás található, ezek "; "-vel vannak elválasztva.

**1.** Legyen a `ceFace(A, m, n)` algoritmus, ahol $\boldsymbol{m}$ természetes szám ($1 \leq \boldsymbol{m} \leq 100$), $\boldsymbol{A}$ egy $\boldsymbol{m}$ elemű, egész számokat tároló vektor ($\boldsymbol{A}[1], \boldsymbol{A}[2], ..., \boldsymbol{A}[\boldsymbol{m}]$, $-10^5 \leq \boldsymbol{A}[\boldsymbol{i}] \leq 10^5$, $\boldsymbol{i} = 1, 2, ..., \boldsymbol{m}$) és $\boldsymbol{n}$ természetes szám ($\boldsymbol{n} \leq \boldsymbol{m}$):

```
Algorithm ceFace(A, m, n):
    For i ← 1, n execute
        min_idx ← i
        For j ← i + 1, m execute
            If A[min_idx] > A[j] then
                min_idx ← j
            EndIf
        EndFor
        aux ← A[i]
        A[i] ← A[min_idx]
        A[min_idx] ← aux
    EndFor
EndAlgorithm
```

Állapítsátok meg, hogy a következő állítások közül melyek igazak?

A. Ha $\boldsymbol{n} = \boldsymbol{m}$, akkor a `ceFace(A, m, n)` algoritmus végrehajtása után a vektor elemei növekvő sorrendbe lesznek rendezve.
B. Ha $\boldsymbol{n} = \boldsymbol{m}$, akkor a `ceFace(A, m, n)` algoritmus végrehajtása után a vektor elemei csökkenő sorrendbe lesznek rendezve.
C. Ha $\boldsymbol{A} = [4, 64, 1, 25, 12, 22, 2, 11]$, $\boldsymbol{n} = 2$ és $\boldsymbol{m} = 8$, akkor a `ceFace(A, m, n)` algoritmus végrehajtása után az $\boldsymbol{A}$ vektornak legkevesebb első 3 eleme növekvő sorrendbe lesz rendezve.
D. Ha $\boldsymbol{n} < \boldsymbol{m}$, akkor a `ceFace(A, m, n)` algoritmus végrehajtása után az $\boldsymbol{A}$ vektornak legkevesebb első $\boldsymbol{n} + 1$ eleme növekvő sorrendbe lesz rendezve.

**2.** Legyen a `h(n, a)` algoritmus, ahol $\boldsymbol{n}$ természetes szám ($1 \leq \boldsymbol{n} \leq 10^3$) és $\boldsymbol{a}$ egy $\boldsymbol{n}$ elemű, egész számokat tároló vektor ($\boldsymbol{a}[1], \boldsymbol{a}[2], ..., \boldsymbol{a}[\boldsymbol{n}]$), ahol $-100 \leq \boldsymbol{a}[\boldsymbol{i}] \leq 100$, $\boldsymbol{i} = 1, 2, ..., \boldsymbol{n}$):

```
Algorithm h(n, a):
    If n = 1 then
        Return a[n]
    Else
        If a[n] > a[n - 1] then
            a[n - 1] ← a[n] - a[n - 1]
        Else
            a[n - 1] ← a[n] + a[n - 1]
        EndIf
        Return h(n - 1, a)
    EndIf
EndAlgorithm
```

Az $\boldsymbol{n}$ szám és az $\boldsymbol{a}$ vektor mely értékeire térít vissza a `h(n, a)` algoritmus 1-et?

A. $\boldsymbol{n} = 6$, $\boldsymbol{a} = [1, 2, 3, 4, 5, 6]$
B. $\boldsymbol{n} = 6$, $\boldsymbol{a} = [6, 5, 4, 3, 2, 1]$
C. $\boldsymbol{n} = 5$, $\boldsymbol{a} = [1, 5, 4, 2, 3]$
D. $\boldsymbol{n} = 2$, $\boldsymbol{a} = [1, 2]$

**3.** Legyen az `E = (x MOD 3 = 0) OR ((y < x) OR NOT ((y * 3) MOD 7 ≤ 3))` kifejezés.

Mi lesz az értéke a kifejezésnek, ha $\boldsymbol{x} = 10$ és $\boldsymbol{y} = 41$?

A. *True*
B. *False*
C. Ugyanaz az érték, mint az `E1 = NOT ((y MOD 3 = 0) OR ((x < y) OR NOT ((x * 3) MOD 7 ≤ 3)))` kifejezés esetében
D. Ugyanaz az érték, mint az `E2 = (x MOD 3 = 0) OR ((x < y) AND ((y * x) MOD 3 ≤ 7))` kifejezés esetében

**4.** Ion a következő algoritmust implementálta azért, hogy ellenőrizze, hogy az ***nr*** természetes szám ($0 < nr < 10^6$) prím-e.

```
Algorithm prim(nr):
    If nr < 2 then
        Return False
    EndIf
    If (nr > 2) AND (nr MOD 2 = 0) then
        Return False
    EndIf
    d ← 3
    While d * d < nr execute
        If nr MOD d = 0 then
            Return False
        EndIf
        d ← d + 2
    EndWhile
    Return True
EndAlgorithm
```

Ion az algoritmus helyességét az ***M*** = {2, 3, 4, 5, 10, 11, 13} halmazhoz tartozó számokra teszteli. A következő állítások közül melyek igazak?

A. Az algoritmus helyes és helyes eredményt térít vissza úgy az ***M***-hez tartozó számokra, mint bármely más, a specifikációnak megfelelő számra.
B. Az algoritmus helytelen, de helyes eredményt térít vissza az ***M***-hez tartozó számokra.
C. Az algoritmus helytelen, és helytelen eredményt térít vissza az ***M***-hez tartozó összes számra.
D. Az algoritmus helytelen, de helyes eredményt térít vissza az ***M***-hez tartozó számok közül legalább egyre, és helytelen eredményt az ***M***-hez tartozó számok közül legalább egy másik számra.

**5.** Legyen az `f(n, x)` algoritmus, ahol ***n*** természetes szám ($1 \le n \le 10^4$) és ***x*** egy ***n*** elemű, egész számokat tároló vektor ($x[1], x[2], ..., x[n], -200 \le x[i] \le 200, i = 1, 2, ..., n$):

```
Algorithm f(n, x):
    a ← True
    i ← 1
    While a AND (i < n) execute
        a ← (x[i] > x[i + 1])
        i ← i + 1
    EndWhile
    Return a
EndAlgorithm
```

A következő bemeneti adatok közül melyekre térít vissza *True*-t az `f(n, x)` algoritmus?

A. Bármely vektor esetében, amely a pozitív elemek után tartalmazza a negatív elemeket
B. Bármely, szigorúan csökkenő vektor esetében
C. Bármely vektor esetében, amely nem tartalmaz pozitív elemeket
D. Ha ***x*** = [5, 4, 3, 2, 1, 0, -1, -2, -3, -4, -5] és ***n*** = 11

**6.** Legyen az $E = AB_{(16)} + 120_{(3)} - 120_{(4)}$ kifejezés, ahol $x_{(b)}$-vel a ***b*** számrendszerben írt ***x*** számot jelöljük.

Melyik érték felel meg az ***E*** kifejezésnek?

A. $162_{(10)}$ B. $278_{(8)}$ C. $1000101_{(2)}$ D. $242_{(8)}$

**7.** Legyen az `f(a, b)` algoritmus, ahol ***a*** és ***b*** nullától különböző természetes szám ($0 < a, b < 10^4$).

```
Algorithm f(a, b):
    If a = 0 then
        Return b
    EndIf
    x ← f(a - 1, b + 1)
    Return f(a - 1, x - 2)
EndAlgorithm
```

Melyik az a legkisebb ***a*** természetes szám, amelyre az `f(a, 15)` alakban meghívott algoritmus egy szigorúan negatív számot térít vissza?

A. 3 B. 4 C. 5 D. 6

**8.** Legyen a `compute(n)` algoritmus, ahol ***n*** természetes szám ($1 < n \le 10^4$).

```
Algorithm compute(n):
    x ← 0
    While n > 0 execute
        If n MOD 2 = 1 then
            x ← x + 1
        EndIf
        n ← n DIV 2
    EndWhile
    Return x
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha ***n*** páratlan, a `compute(n)` algoritmus egy 1-nél nagyobb értéket térít vissza.
B. A `compute(n)` algoritmus a 2-es számrendszerben felírt ***n*** szám számjegyeinek összegét téríti vissza.
C. A `compute(n)` algoritmus az ***n*** szám páratlan osztóinak (valódi és nem valódi) darabszámát téríti vissza.
D. A `compute(n)` algoritmus a 2-es számrendszerben felírt ***n*** szám 1-es bitjeinek darabszámát téríti vissza.

**9.** Legyen az `f(p, q, r)` algoritmus, ahol ***p***, ***q*** és ***r*** logikai értékek:

```
Algorithm f(p, q, r):
    While (p AND (NOT r)) OR (NOT q) execute
        Write (q AND (p OR r))
        p ← NOT p
        r ← q OR p
    EndWhile
EndAlgorithm
```

A következő állítások közül melyek igazak az `f(True, False, True)` hívás esetében?

A. Az algoritmus végtelen ciklusba jut és ismételten *False*-t ír ki.
B. Az algoritmus nem ír ki semmit.
C. Az algoritmus kiírja egyszer a *False* értéket.
D. Az algoritmus a *False True False* értékeket írja ki.

**10.** Legyen a következő bináris fa:

```
      1
     / \
    2   3
   / \
  4   5
```

A következő csomópont-sorozatok közül melyik felel meg a fa preorderben történő bejárásának?

A. 1, 2, 4, 5, 3
B. 4, 2, 5, 1, 3
C. 1, 2, 3, 4, 5
D. 4, 5, 2, 3, 1

**11.** Legyen a `mark(n, m, a)` algoritmus, ahol ***n*** és ***m*** nullától különböző természetes számok ($1 \leq \boldsymbol{n}, \boldsymbol{m} \leq 10$) és ***a*** egy ***n*** elemű, természetes számokat tároló vektor (***a***[1], ***a***[2], ..., ***a***[***n***]). A `tuple(i, j, k)` algoritmus, ahol ***i***, ***j*** és ***k*** nullától különböző természetes számok ($1 \leq \boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k} \leq 10$) *True*-t vagy *False*-t térít vissza.

```
Algorithm mark(n, m, a):
   a[1] ← 1
   For i ← 2, n execute
      a[i] ← 0
   EndFor
   ready ← False
   While NOT ready execute
      ready ← True
      For i ← 1, n execute
         For j ← 1, n execute
            For s ← 1, m execute
               If a[i] = 1 AND tuple(i, s, j) AND a[j] = 0 then
                  a[j] ← 1
                  ready ← False
               EndIf
            EndFor
         EndFor
      EndFor
   EndWhile
EndAlgorithm
```

Feltételezzük, hogy az alábbi számhármasok mindegyikének esetében a `tuple(i, j, k)` algoritmus *True*-t térít vissza. Melyek azok a számhármas-párok, amelyeknek esetében a `mark(3, 3, a)` hívás eredményeként az ***a*** vektor minden elemének értéke 1 lesz?

A. (1, 1, 2) și (2, 2, 3)
B. (1, 1, 2) și (3, 2, 2)
C. (1, 2, 2) și (1, 3, 3)
D. (1, 2, 2) și (3, 3, 1)

**12.** Legyen az ***n*** soros és ***n*** oszlopos ***mat*** mátrix ($1 \leq \boldsymbol{n} \leq 200$, ***mat***[1][1], ..., ***mat***[1][***n***], ***mat***[2][1], ..., ***mat***[2][***n***], ..., ***mat***[***n***][1], ..., ***mat***[***n***][***n***]) és a `matrice(mat, n)` algoritmus.

```
Algorithm matrice(mat, n):
    k ← 1
    For i ← 1, n execute
        For j ← 1, n execute
            mat[i][j] ← k
            k ← k * -1
        EndFor
    EndFor
    Return mat
EndAlgorithm
```

A következő állítások közül melyek igazak a visszatérített mátrixra a `matrice(mat, n)` hívás végrehajtása után?

A. Ha ***n*** = 31, a főátló elemeinek szorzata 1.
B. Ha ***n*** = 32, az első sor elemeinek szorzata 1.
C. Ha ***n*** = 127, az utolsó sor utolsó oszlopában található elem értéke -1.
D. Ha ***n*** = 128, az első oszlop elemeinek összege 1.

**13.** Legyen a `modifica(n, a)` algoritmus, ahol ***n*** természetes szám ($1 \leq n \leq 10^3$) és ***a*** egy ***n*** elemű, egész számokat tároló vektor (***a***[1], ***a***[2], ..., ***a***[***n***], -100 ≤ ***a***[***i***] ≤ 100, ***i*** = 1, ..., ***n***):

```
Algorithm modifica(n, a):
    x ← a[n]
    i ← 0
    For j ← 1, n - 1 execute
        If a[j] ≤ x then
            i ← i + 1
            t ← a[i]
            a[i] ← a[j]
            a[j] ← t
        EndIf
    EndFor
    t ← a[i + 1]
    a[i + 1] ← a[n]
    a[n] ← t
    Return a
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha az ***a*** vektor növekvően rendezett, az algoritmus végrehajtása után is növekvően rendezett marad.
B. Ha az ***a*** vektor szigorúan csökkenően rendezett, akkor az algoritmus által visszatérített vektorban a maximális elem az utolsó pozíción lesz.
C. Az algoritmus által visszatérített vektorban a maximális elem mindig az utolsó pozíción lesz.
D. Ha ***n*** = 100, és az ***a*** vektor elemeire érvényes, hogy ***a***[***i***] = ***i*** **MOD** 2, (***i*** = 1, 2, ..., ***n***). akkor az algoritmus végrehajtása után a vektor növekvően rendezett lesz.

**14.** Legyen az `f(v, n)` algoritmus, ahol ***n*** természetes szám ($2 \leq n \leq 10^4$) és ***v*** egy ***n*** elemű, természetes számokat tartalmazó vektor (***v***[1], ***v***[2], ..., ***v***[***n***], $1 \leq v[i] \leq 10^3$, ***i*** = 1, 2, ..., ***n***).

```
Algorithm f(v, n):
    a ← 0; b ← n; i ← 1
    While i < n execute
        If v[i] MOD 3 = 0 then
            a ← a + v[i]
            b ← b + 1
        EndIf
        i ← i + 1
        b ← b - 1
    EndWhile
    If b = 0 then
        Return 0
    EndIf
    i ← 0
    While a ≥ b execute
        a ← a – b
        i ← i + 1
    EndWhile
    Return i
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Az algoritmus visszatéríti a ***v*** vektor azon elemeinek a számtani középarányosát, amelyek 3-nak többszörösei, vagy 0-át, ha a vektorban nincs olyan elem, ami 3-nak többszöröse.
B. Az algoritmus visszatéríti a ***v*** vektor azon elemeinek legnagyobb közös osztóját, amelyek 3-nak többszörösei, vagy 0-át, ha a vektorban nincs olyan elem, ami 3-nak többszöröse.
C. Az algoritmus visszatéríti a ***v*** vektor azon elemeinek a darabszámát, amelyek 3-nak többszörösei, vagy 0-át, ha a vektorban nincs olyan elem, ami 3-nak többszöröse.
D. Az A., B., C válaszok közül egyik sem igaz.

**15.** Abból a célból, hogy meghatározza minden részhalmazát az ***A*** = {4, 8, 9, 12, 15} 5 elemű halmaznak, egy tanuló megírta a `generare(i, n, x, A)` algoritmust. A halmazt az ***n*** elemű, természetes számokat tároló ***A*** vektor ábrázolja. A generált részhalmazokat az `afis(m, x, A)` algoritmus íratja ki, ahol ***x*** egy 0-tól indexelt segédvektor, és az ***m*** természetes szám az aktuális ***x*** vektor mérete. A `generare(1, 5, x, A)` hívás előtt az ***x***[0] elem 0-val volt inicializálva.

```
Algorithm generare(i, n, x, A):
    For j ← n, x[i - 1] + 1, -1 execute
        x[i] ← j
        afis(i, x, A)
        generare(i + 1, n, x, A)
    EndFor
EndAlgorithm
```

```
Algorithm afis(m, x, A):
    Write "{", a[x[1]]
    For i ← 2, m execute
        Write ", ", a[x[i]]
    EndFor
    Write "}", newline
EndAlgorithm
```

Tudva, hogy az első 4 kiírt részhalmaz, ebben a sorrendben: {15}, {12}, {12, 15}, {9}, melyik lesz a 8-dik generált részhalmaz (az üres részhalmazt nem vesszük figyelembe)?

A. {9, 12}  B. {8}  C. {9, 12, 15}  D. {8, 15}

**16.** Legyen az `f(x, n, k)` algoritmus, ahol ***n*** és ***k*** természetes számok ($3 \leq \boldsymbol{n} \leq 10^4$, $1 \leq \boldsymbol{k} \leq 10^4$), ***x*** egy ***n*** elemű, természetes számokat tároló vektor (***x***[1], ***x***[2], ..., ***x***[***n***], $1 \leq \boldsymbol{x}[\boldsymbol{i}] \leq 10^4$, ***i*** = 1, 2, ..., ***n***):

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

A következő hívások közül melyek esetén térít vissza az algoritmus 10-et?

A. `f([1, 4, 6], 3, 3)`
B. `f([1, 2, 3, 4, 5], 5, 3)`
C. `f([1, 2, 3, 4], 4, 4)`
D. `f([10, 15, 25], 3, 1)`

**17.** Legyen a `decide(n)` algoritmus, ahol ***n*** természetes szám ($10^4 \leq \boldsymbol{n} \leq 10^7$):

```
Algorithm decide(n):
    m ← 10
    abc ← n DIV m
    While abc ≥ 1000 execute
        m ← m * 10
        abc ← n DIV m
    EndWhile
    bc ← abc MOD 100
    f ← (bc < 2)
    i ← 2
    While i ≤ bc DIV 2 execute
        If bc MOD i = 0 then
            f ← True
            i ← bc
        EndIf
        i ← i + 1
    EndWhile
    Return f
EndAlgorithm
```

A következő hívások közül melyek esetén térít vissza az algoritmus *True*-t?

A. `decide(865756)`
B. `decide(72387)`
C. `decide(103983)`
D. `decide(10405)`

**18.** Legyen a `ceFace(n)` algoritmus, ahol ***n*** nullától különböző természetes szám ($1 \leq \boldsymbol{n} < 10^3$).

```
Algorithm ceFace(n):
    Return ceFaceRecursiv(n, 1, 1)
EndAlgorithm

Algorithm ceFaceRecursiv(n, a, b):
    If n = 0 then
        Return 1
    Else
        If n < 0 OR b > n then
            Return 0
        Else
            Return ceFaceRecursiv(n, a + b, a) + ceFaceRecursiv(n - a, a + b, a)
        EndIf
    EndIf
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. A [11, 16] intervallumban egyetlen olyan ***x*** érték van, amelyre a `ceFace(x)` algoritmus 1-et térít vissza.
B. Bármely ***n*** szám esetében a `ceFace(n)` algoritmus 0-át vagy 1-et térít vissza.
C. A `ceFace(n)` algoritmus azoknak az összegeknek a darabszámát téríti vissza, amelyekben az ***n*** szám, egymásutáni számok összege.
D. A `ceFace(n)` algoritmus azoknak a különböző halmazoknak a darabszámát téríti vissza, amelyeknek az elemei 0-tól különböző Fibonacci-számok és amelyeknek összege ***n***.

**19.** Legyen a `ceFace(x, n)` algoritmus, ahol ***n*** természetes szám ($1 \le \boldsymbol{n} \le 10^4$), ***x*** egy ***n*** elemű, számjegyeket tároló vektor (***x***[1], ***x***[2], ..., ***x***[***n***], $1 \le \boldsymbol{x}[\boldsymbol{i}] \le 9$, ***i*** = 1, 2, ..., ***n***), és a `Zero(k)` algoritmus, amely egy ***k*** elemű, 0-kat tároló vektort térít vissza:

```
Algorithm ceFace(x, n):
    f ← Zero(9)
    For i ← 1, n execute
        f[x[i]] ← f[x[i]] + 1
    EndFor
    i ← 9
    nr ← 0
    While i > 0 execute
        If f[i] = 0 then
            nr ← nr * 10 + i
        EndIf
        i ← i - 1
    EndWhile
    Return 10 * nr
EndAlgorithm
```

Mit térít vissza az adott algoritmus?

A. Egy számot, amelyet az ***x*** vektor számjegyei alkotnak
B. Egy számot, amelyet az ***x*** vektor számjegyei alkotnak, minden számjegy csak egyszer fordul elő
C. Azt a lehetséges legnagyobb számot, amelyet azok a különböző számjegyek alkotnak, amelyek nem fordulnak elő az ***x*** vektorban
D. Azt a lehetséges legkisebb számot, amelyet azok a különböző számjegyek alkotnak, amelyek nem fordulnak elő az ***x*** vektorban

**20.** Legyenek az ***n*** és ***m*** természetes számok ($1 \le \boldsymbol{n}, \boldsymbol{m} \le 100$) és az ***n*** soros és ***m*** oszlopos ***matrix*** mátrix, amelynek elemei 0 vagy 1. Legyenek a `prelucrare(matrix, row, col, n, m)` és `num(matrix, n, m)` algoritmusok, ahol ***row*** és ***col*** természetes számok ($1 \le \boldsymbol{row} \le \boldsymbol{n}$, $1 \le \boldsymbol{col} \le \boldsymbol{m}$).

```
Algorithm prelucrare(matrix, row, col, n, m):
    If row ≥ 1 AND row ≤ n AND col ≥ 1 AND col ≤ m AND matrix[row][col] = 1 then
        matrix[row][col] ← 0
        prelucrare(matrix, row - 1, col, n, m)
        prelucrare(matrix, row + 1, col, n, m)
        prelucrare(matrix, row, col – 1, n, m)
        prelucrare(matrix, row, col + 1, n, m)
    EndIf
EndAlgorithm

Algorithm num(matrix, n, m):
    c ← 0
    For row ← 1, n execute
        For col ← 1, m execute
            If matrix[row][col] = 1 then
                c ← c + 1
                prelucrare(matrix, row, col, n, m)
            EndIf
        EndFor
    EndFor
    Return c
EndAlgorithm
```

Szigetnek nevezzük azt a részét a mátrixnak, ahol a vízszintesen vagy függőlegesen szomszédos elemek azonosak. A következő állítások közül melyek igazak?

A. Ha $\boldsymbol{n} \ne \boldsymbol{m}$ a `num(matrix, n, m)` algoritmus nem vizsgálja meg a mátrix összes elemét.
B. A következő 5 soros és 5 oszlopos mátrix esetében:
```
matrix =
 1  1  0  0  0
 1  1  0  0  0
 0  0  1  0  0
 0  0  0  1  1
 0  0  0  1  1
```
a `num(matrix, 5, 5)` hívás 3-at térít vissza.
C. A `num(matrix, n, m)` algoritmus az adott mátrix 0-kat tároló szigeteinek a darabszámát téríti vissza.
D. A `num(matrix, n, m)` algoritmus az adott mátrix 1-eseket tároló szigeteinek a darabszámát téríti vissza.

**21.** Legyen az ***r*** és az ***s*** két karakterlánc, amelyeknek a hossza ***Lung*** ($1 \leq Lung \leq 256$). Adottak a következő algoritmusok:

- A `copiere(a, primul, ultimul)` algoritmus visszatéríti azt a karakterláncot, amely az ***a*** karakterláncnak a ***primul*** pozíción kezdődő és az ***ultimul*** pozíción záródó részét tartalmazza.
- Az `egale(a, b, k)` algoritmus *True*-t térít vissza, ha az ***a*** és ***b*** karakterlánc, amelyeknek hossza ***k***, azonosak, és *False*-t különben.
- A `lungime(a)` algoritmus visszatéríti az ***a*** karakterlánc hosszát.
- A `concatenare(a, b)` algoritmus azt a karakterláncot téríti vissza, amelyet az ***a*** és ***b*** karakterláncok összeragasztása révén kapunk, ebben a sorrendben.

Állapítsátok meg, hogy a következő algoritmusok közül melyik térít vissza *True*-t, ha az ***r*** karakterlánc megkapható az ***s*** karakterláncból, ha 0, 1, vagy többször elforgatjuk az ***s***-t. Például, a `"abcde"` karakterlánc előállítható a `"cdeab"` karakterlánc elforgatása által.

A.
```
Algorithm check(s, r, Lung):
    For i ← 1, Lung execute
        If egale(s, r, Lung) then
            Return True
        EndIf
        aux ← s[1]
        For j ← 2, Lung execute
            s[j - 1] ← s[j]
        EndFor
        s[Lung] ← aux
    EndFor
    Return False
EndAlgorithm
```

B.
```
Algorithm check(s, r, Lung):
    ss ← concatenare(s, s)
    i ← 1
    sf ← Lung + 1
    While i ≤ sf execute
        k ← i
        j ← 1
        While j ≤ Lung AND ss[k] = r[j] execute
            j ← j + 1
            k ← k + 1
        EndWhile
        If j > Lung then
            Return True
        EndIf
        i ← i + 1
    EndWhile
    Return False
EndAlgorithm
```

C.
```
Algorithm check(s, r, Lung):
    ss ← concatenare(r, s)
    i ← 1
    While i ≤ Lung execute
        k ← i
        j ← 1
        While j ≤ Lung AND ss[k] = r[j] execute
            j ← j + 1
            k ← k + 1
        EndWhile
        If j > Lung then
            Return True
        EndIf
        i ← i + 1
    EndWhile
    Return False
EndAlgorithm
```

D.
```
Algorithm check(s, r, Lung):
    pos1 ← 1
    ok ← False
    While r[pos1] ≠ s[1] execute
        pos1 ← pos1 + 1
    EndWhile
    If pos1 > 0 then
        ok ← egale(s, r, Lung)
    EndIf
    If NOT ok then
        pos2 ← Lung - pos1 + 1
        ok ← (r[1] = s[pos2])
        ss ← copiere(s, pos2, Lung)
        rr ← copiere(r, 1, pos1)
        ok ← ok AND egale(rr, ss, lungime(ss))
    EndIf
    Return ok
EndAlgorithm
```

**22.** Legyen a `ceFace(a, n)` algoritmus, ahol ***n*** természetes szám ($2 < n \leq 10^4$) és ***a*** egy ***n*** természetes számot tároló vektor ($a[1], a[2], ..., a[n]$, $0 \leq a[i] \leq 10^4$, $i = 1, 2, ..., n$). Adva van a `nrPalindromuri(b, p, r)` algoritmus, ahol ***b*** egy ***m*** természetes számot tároló vektor ($b[1], b[2], ..., b[m]$, $0 \leq b[j] \leq 10^4$, $j = 1, 2, ..., m$, $2 < m < 10^4$). A ***p*** és ***r*** paraméter természetes szám és $1 \leq p < r \leq m$. A `nrPalindromuri(b, p, r)` algoritmus a ***b*** vektor $b[p], ..., b[r]$ tömbszakaszában található palindromszámok darabszámát téríti vissza.

```
Algorithm ceFace(a, n):
    b ← 0; c ← b; e ← 0; d ← 0
    For i ← 1, n - 2 execute
        If nrPalindromuri(a, i, i + 2) > 1 then
            If c = 0 then
                d ← i
            EndIf
            c ← c + 1
        Else
            If c > b then
                b ← c; e ← d
            EndIf
            c ← 0
        EndIf
    EndFor
    If c > b then
        b ← c; e ← d
    EndIf
    If b = 0 then
        Write 0, " ", 0
    Else
        Write e, " ", e + b + 1
    EndIf
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha egy adott, $10^4$ elemű ***a*** vektor esetében az algoritmus a 7381 7384 számokat írja ki, következik, hogy a [7381, ..., 7384] intervallumhoz tartozó pozíciókon található 4 szám között pontosan két palindromszám található.

B. Ha ***n*** = 12 és ***a*** = [11, 33, 45, 103, 121, 343, 33, 99, 100, 22, 44, 45] a `ceFace(a, n)` algoritmus 5 8-at ír ki.

C. Ha az algoritmus végrehajtása után ***b*** értéke 0, következik, hogy az ***a*** vektorban nincs egyetlen palindromszám sem.

D. Ha ***n*** = 12 és ***a*** = [11, 33, 45, 103, 121, 343, 33, 99, 100, 22, 44, 45] a `ceFace(a, n)` algoritmus 4 12-őt ír ki.

**23.** Legyen a `fun(a, b, len)` algoritmus, ahol ***len*** természetes szám ($1 \leq$ ***len*** $\leq 100$), ***a*** és ***b*** két ***len*** elemű vektor (***a***[1], ***a***[2], ..., ***a***[***len***], ***b***[1], ***b***[2], ..., ***b***[***len***], 1 ≤ ***a***[***i***], ***b***[***i***] ≤ ***len***, i = 1, 2, …, ***len***).

```
Algorithm fun(a, b, len):
    For i ← 1, len execute
        k ← a[b[i]]
        a[b[i]] ← b[a[i]]
        b[a[i]] ← k
    EndFor
EndAlgorithm
```

Legyen ***len*** = 7, ***a*** = [6, 2, 5, 4, 1, 3, 4] és ***b*** = [1, 2, 3, 5, 6, 4, 4]. A `fun(a, b, len)` algoritmus végrehajtása előtt a két vektor két olyan azonos elemet tartalmaz, amelyek azonos pozíciókon találhatók (***a***[2] = ***b***[2] és ***a***[7] = ***b***[7]).

A következő állítások közül melyek igazak, ha az algoritmust `fun(a, b, len)` alakban hívjuk meg?

A. Az ***a*** és ***b*** vektor 3. és 6. pozícióján azonos elemek lesznek.
B. Az ***a*** és ***b*** vektor 3 azonos pozícióján lesz azonos értékű elem.
C. A ***b*** vektor az [1, 2, 3, 4, 6, 5, 4] értékeket fogja tárolni.
D. Az ***a*** vektor a [4, 2, 6, 3, 6, 1, 4] értékeket fogja tárolni.

**24.** Legyen a `calculeaza(v, b, n, i)` algoritmus, ahol ***b***, ***n***, ***i*** nullától különböző természetes számok ($1 \leq$ ***b***,***n***,***i*** $\leq 10^3$) és ***v*** egy ***n*** elemű, természetes számokat tároló vektor (***v***[1], ***v***[2], ..., ***v***[***n***], $0 \leq$ ***v***[***i***] $\leq 10^3$, ***i*** = 1, 2, ..., ***n***):

```
Algorithm calculeaza(v, b, n, i):
    If b = 0 then
        Return True
    EndIf
    If i = n then
        Return False
    EndIf
    Return calculeaza(v, b - v[i], n, i + 1) OR calculeaza(v, b, n, i + 1)
EndAlgorithm
```

A következő bemeneti adatok közül melyekre tért vissza az algoritmus *True*-t?

A. ***v*** = [3, 1, 7, 4, 2], ***b*** = 10, ***n*** = 5, ***i*** = 1
B. ***v*** = [2, 6, 4, 8, 12], ***b*** = 12, ***n*** = 5, ***i*** = 1
C. ***v*** = [3, 1, 7, 4, 2], ***b*** = 10, ***n*** = 5, ***i*** = 2
D. ***v*** = [2, 6, 4, 8, 12], ***b*** = 12, ***n*** = 5, ***i*** = 3

**BABEŞ-BOLYAI TUDOMÁNYEGYETEM**
**MATEMATIKA ÉS INFORMATIKA KAR**

Felvételi verseny – 2024 július 19.

Informatika írásbeli

JAVÍTÁSI KULCS & MEGOLDÁSOK

**HIVATALBÓL**: 10 pont

| | | |
|---|---|---|
| **1.** | ACD | 3.75 pont |
| **2.** | BD | 3.75 pont |
| **3.** | AD | 3.75 pont |
| **4.** | B | 3.75 pont |
| **5.** | BD | 3.75 pont |
| **6.** | AD | 3.75 pont |
| **7.** | C | 3.75 pont |
| **8.** | ABD | 3.75 pont |
| **9.** | A | 3.75 pont |
| **10.** | A | 3.75 pont |
| **11.** | AC | 3.75 pont |
| **12.** | AB | 3.75 pont |
| **13.** | ABD | 3.75 pont |
| **14.** | D | 3.75 pont |
| **15.** | B | 3.75 pont |
| **16.** | CD | 3.75 pont |
| **17.** | AD | 3.75 pont |
| **18.** | AD | 3.75 pont |
| **19.** | C | 3.75 pont |
| **20.** | BD | 3.75 pont |
| **21.** | AB | 3.75 pont |
| **22.** | AD | 3.75 pont |
| **23.** | BCD | 3.75 pont |
| **24.** | ABD | 3.75 pont |