BABEŞ-BOLYAI TUDOMÁNYEGYETEM
MATEMATIKA-INFORMATIKA KAR

MATEK-INFO verseny – 2024 április 20

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.
- Ha ugyanabban a sorban több egymásutáni értékadó utasítás található, ezek "; "-vel vannak elválasztva.

1. Adott a `calcul(v, n)` algoritmus, ahol $\boldsymbol{n}$ természetes szám ($1 \leq \boldsymbol{n} \leq 10^4$) és $\boldsymbol{v}$ egy $\boldsymbol{n}$ természetes számot tartalmazó vektor ($\boldsymbol{v}[1], \boldsymbol{v}[2], ..., \boldsymbol{v}[\boldsymbol{n}]$, $1 \leq \boldsymbol{v}[\boldsymbol{i}] \leq 10^4$, $\boldsymbol{i} = 1, 2, ..., \boldsymbol{n}$):

```
Algorithm calcul(v, n):
    i ← 1; j ← n
    While i < j execute
        While i < j AND v[i] MOD 2 = 1 execute
            i ← i + 1
        EndWhile
        While i < j AND v[j] MOD 2 = 1 execute
            j ← j - 1
        EndWhile
        If v[i] ≠ v[j] then
            Return False
        EndIf
        i ← i + 1
        j ← j - 1
    EndWhile
    Return True
EndAlgorithm
```

Mely esetekben térít vissza az algoritmus *True*-t?

A. Ha a $\boldsymbol{v}$ vektor értékei [1, 11, 2, 4, 3, 4, 7, 6, 4, 21, 23, 25, 2] és $\boldsymbol{n} = 13$
B. Ha a $\boldsymbol{v}$ vektor értékei [1, 11, 2, 4, 3, 7, 6, 4, 21, 23, 25, 2] és $\boldsymbol{n} = 12$
C. Akkor és csakis akkor, ha a $\boldsymbol{v}$ vektor két olyan páros értékű eleme különbségének abszolút értéke egyenlő 2-vel, amelyek között létezik legalább egy páratlan értékű elem.
D. Ha a $\boldsymbol{v}$ vektor páros elemeiből létrehozott vektor balról jobbra bejárva egyenlő a $\boldsymbol{v}$ vektor páros eleimeiből létrehozott vektorral, amelyet jobbról balra járunk be

2. Adott a `g(a, b)` algoritmus, ahol $\boldsymbol{a}$ és $\boldsymbol{b}$ természetes számok ($0 \leq \boldsymbol{a}, \boldsymbol{b} \leq 10^4$):

```
Algorithm g(a, b):
    If a = b then
        Return a
    EndIf
    If a > b then
        Return g(a - b, b)
    Else
        Return g(a, b - a)
    EndIf
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha az algoritmust `g(2, 2)` alakban hívjuk meg, 2-t fog visszatéríteni.
B. Ha $\boldsymbol{a} = \boldsymbol{b}$, az algoritmus nem hívja meg önmagát egyszer sem.
C. Ha $\boldsymbol{a} = 0$ és $0 \leq \boldsymbol{b} \leq 10^4$, az algoritmus egyszer hívja meg önmagát.
D. Ha $\boldsymbol{a} \neq 0$, $\boldsymbol{b} \neq 0$ és $\boldsymbol{a} \neq \boldsymbol{b}$, az algoritmus $(\boldsymbol{a} + \boldsymbol{b} - 1)$-szer hívja meg önmagát.

3. Egy irányított gráfnak 8 csúcsa van, amelyek 1-től 8-ig vannak számozva, és az ívei: (1, 7), (1, 8), (3, 5), (3, 7), (4, 3), (4, 7), (6, 3), (6, 5), (6, 7), (6, 8), (8, 5), (8, 7). Azoknak a csúcsoknak a darabszáma, amelyeknek a kifoka nulla:

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

4. Mi az értéke a `NOT ((x MOD 2 = 0) AND (NOT ((y > x) AND (x MOD 7 ≠ 5))))` kifejezésnek, ha $\boldsymbol{x} = 12$ és $\boldsymbol{y} = 23$?

A. *True*
B. *False*
C. Ugyanaz az érték, mint a `NOT ((x MOD 2 = 0) AND (NOT ((x > y) AND (x MOD 7 ≠ 5))))` kifejezésé
D. Ugyanaz az érték, mint a `NOT ((y MOD 2 = 0) AND (NOT ((x > y) AND (y MOD 7 ≠ 5))))` kifejezésé

5. Adott a `ghici(n)` algoritmus, ahol ***n*** természetes szám ($1 \leq \boldsymbol{n} \leq 10^9$):

```
Algorithm ghici(n):
    f ← 0; y ← -1
    For c ← 0, 9 execute
        x ← n
        k ← 0
        While x > 0 execute
            If x MOD 10 = c then
                k ← k + 1
            EndIf
            x ← x DIV 10
            If k > f then
                f ← k
                y ← c
            EndIf
        EndWhile
    EndFor
    Return y
EndAlgorithm
```

Mit fog visszatéríteni az algoritmus?

A. Az ***n*** szám számjegyeinek darabszámát
B. Az ***n*** szám számjegyei előfordulási számainak maximumát
C. Egyet az ***n*** szám számjegyei közül, amelynek előfordulási száma maximális
D. Egyet az ***n*** szám maximális értékű számjegyei közül

6. Adott a `divizori(n)` algoritmus, ahol ***n*** egész szám ($-10^3 \leq \boldsymbol{n} \leq 10^3$).

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

A következő állítások közül melyek igazak?

A. Ha ***n*** = 5, az algoritmus 2-t térít vissza.
B. Ha ***n*** > 1, az algoritmus visszatéríti az ***n*** szám összes (valódi és nem valódi) osztóinak darabszámát.
C. Ha ***n*** = 0, az algoritmus 0-át térít vissza.
D. Ha ***n*** < 0, az algoritmus visszatéríti az ***n*** szám abszolút értékének összes (valódi és nem valódi) osztóinak darabszámát.

7. Adott a `ceReturneaza(a, b)` algoritmus, ahol ***a*** és ***b*** természetes számok ($0 \leq \boldsymbol{a}, \boldsymbol{b} \leq 10^3$):

```
Algorithm ceReturneaza(a, b):
    If a > b then
        c ← a; a ← b; b ← c
    EndIf
    d ← 0
    For i ← a, b execute
        If i MOD 2 = 0 then
            d ← d + 1
        EndIf
    EndFor
    Return d
EndAlgorithm
```

A következő esetek közül melyekben térít vissza az algoritmus 0-át?

A. ***a*** = 11, ***b*** = 11
B. ***a*** = 4, ***b*** = 8
C. ***a*** = 12, ***b*** = 12
D. ***a*** = 0, ***b*** = 0

8. Adott a `ceFace(n)` algoritmus, ahol ***n*** természetes szám ($1 \leq \boldsymbol{n} \leq 10^4$):

```
Algorithm ceFace(n):
    k ← 0; s ← 0
    While k ≠ n execute
       k ← k + 1
       s ← s + 2 * k - 1
       Write s, " "
    EndWhile
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha ***n*** = 3, az algoritmus kiírja: `0 9`
B. Ha ***n*** = 10, az ***s*** változó utolsó előtti értéke a végrehajtás során `81`
C. Az algoritmus kiírja az 1, 2, …, ***n*** természetes számok négyzetét
D. Ha ***n*** = 4, az algoritmus kiírja: `1 4 8 16`

9. Adottak a `verificare_aux(a, b)` és a `verificare(a, b)` algoritmusok, ahol $\boldsymbol{a}$ és $\boldsymbol{b}$ természetes számok ($1 \leq \boldsymbol{a}, \boldsymbol{b} \leq 10^9$):

```
Algorithm verificare_aux(a, b):
    c ← b
    While a > 0 execute
        While (c > 0) AND (a MOD 10 ≠ c MOD 10) execute
            c ← c DIV 10
        EndWhile
        If c = 0 then
            Return False
        EndIf
        c ← b
        a ← a DIV 10
    EndWhile
    Return True
EndAlgorithm
```

```
Algorithm verificare(a, b):
    Return verificare_aux(a, b) AND verificare_aux(b, a)
EndAlgorithm
```

Az alábbi feltételek közül melyeknek esetében térít vissza a `verificare(a, b)` algoritmus *True*-t?

A. Ha $\boldsymbol{a}$ és $\boldsymbol{b}$ számjegyeinek darabszáma azonos.
B. Ha $\boldsymbol{a}$ =1001 és $\boldsymbol{b}$ = 10.
C. Ha $\boldsymbol{a}$ számjegyeinek frekvenciatömbje azonos $\boldsymbol{b}$ számjegyeinek frekvenciatömbjével.
D. Ha $\boldsymbol{a}$ = 123 és $\boldsymbol{b}$ = 321.

10. Adott a `verifica(n)` algoritmus, ahol $\boldsymbol{n}$ természetes szám ($1 \leq \boldsymbol{n} \leq 10^4$).

```
Algorithm verifica(n):
    a ← n MOD 10
    n ← n DIV 10
    While n > 0 execute
        b ← n MOD 10
        If a ≤ b then
            Return False
        EndIf
        a ← b
        n ← n DIV 10
    EndWhile
    Return True
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. A `verifica(2024)` hívás eredményeként az algoritmus *False*-t térít vissza
B. Az algoritmus akkor és csakis akkor térít vissza *True*-t, ha $\boldsymbol{n}$ számjegyei szigorúan növekvő sorrendben találhatók.
C. Az algoritmus akkor és csakis akkor térít vissza *True*-t, ha $\boldsymbol{n}$ számjegyei szigorúan csökkenő sorrendben találhatók.
D. Az algoritmus akkor és csakis akkor térít vissza *True*-t, ha $\boldsymbol{n}$ legértékesebb számjegye kisebb, mint a legkevésbé értékes számjegye.

11. Adott az `F(x, n, i, S, k)` algoritmus, ahol $\boldsymbol{x}$ egy $\boldsymbol{n}$ egész számot tartalmazó vektor ($1 \leq \boldsymbol{n} \leq 10^4$, $\boldsymbol{x}[1], ..., \boldsymbol{x}[\boldsymbol{n}]$, $-10^3 \leq \boldsymbol{x}[\boldsymbol{i}] \leq 10^3$, $\boldsymbol{i} = 1, 2, ..., \boldsymbol{n}$), $\boldsymbol{S}$ egy valós szám, $\boldsymbol{i}$ és $\boldsymbol{k}$ természetes számok. A „/” operátor a valós osztást jelöli, például: 3 / 2 = 1.5.

```
Algorithm F(x, n, i, S, k):
    If n < i then
        If k = n then
            Return 0
        Else
            Return S / (n - k)
        EndIf
    Else
        If x[i] MOD 2 = 0 then
            Return F(x, n, i + 1, S + x[i], k)
        Else
            Return F(x, n, i + 1, S, k + 1)
        EndIf
    EndIf
EndAlgorithm
```

Ha az algoritmust az `F(x, n, 1, 0.0, 0)` alakban hívjuk meg, a következő állítások közül melyek igazak?

A. Az algoritmus az $\boldsymbol{x}$ vektor páros szám elemeinek összegét elosztja a páratlan számok darabszámával és ezt téríti vissza.
B. Ha $\boldsymbol{n}$ = 10 és $\boldsymbol{x}$ = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], az algoritmus 6.0-t térít vissza.
C. Az algoritmus az $\boldsymbol{x}$ vektor páros szám elemeinek számtani közepét téríti vissza.
D. Ha $\boldsymbol{n}$ = 10 és $\boldsymbol{x}$ = [1, 3, 5, 7, 9, 11, 13, 15, 17, 19], az algoritmus 0-át térít vissza.

12. Adott az $\boldsymbol{n}$ méretű $\boldsymbol{x}$ négyzetes mátrix, amelynek elemei különböző természetes számok ($2 \leq \boldsymbol{n} \leq 50$, $\boldsymbol{x}[1][1], ..., \boldsymbol{x}[1][\boldsymbol{n}]$, $\boldsymbol{x}[2][1], ..., \boldsymbol{x}[2][\boldsymbol{n}], ..., \boldsymbol{x}[\boldsymbol{n}][1], ..., \boldsymbol{x}[\boldsymbol{n}][\boldsymbol{n}]$, $1 \leq \boldsymbol{x}[\boldsymbol{i}][\boldsymbol{j}] \leq 10^4$, $\boldsymbol{i} = 1, 2, ..., \boldsymbol{n}$, $\boldsymbol{j} = 1, 2, ..., \boldsymbol{n}$). Minden sorban és minden oszlopban az elemek növekvő sorrendbe vannak rendezve. A `cauta(n, x, v)` algoritmus keresi a $\boldsymbol{v}$ értéket az $\boldsymbol{x}$ mátrixban és visszatéríti annak a sornak és annak az oszlopnak az indexét, ahol a $\boldsymbol{v}$ érték megtalálható a mátrixban vagy (-1, -1)-et, ha a $\boldsymbol{v}$ érték nem található meg a mátrix elemei között. Feltételezzük, hogy a `cautareBinara(t, n, v)` algoritmus a bináris keresést implementálja abból a célból, hogy megállapítsa a $\boldsymbol{v}$ szám létezését az $\boldsymbol{n}$ elemű, növekvően rendezett $\boldsymbol{t}$ vektorban. Ha $\boldsymbol{v}$ nem található a mátrix $\boldsymbol{i}$-dik sorában, a `cautareBinara(x[i], n, v)` algoritmus a hívás után -1-et térít vissza.
A következő algoritmusok közül melyik a leghatékonyabb a futási idő szempontjából és teljesíti a kért követelményeket?

A.
```
Algorithm cauta(n, x, v):
    a ← -1
    b ← -1
    For i ← 1, n execute
        For j ← 1, n execute
            If x[i][j] = v then
                a ← i
                b ← j
            EndIf
        EndFor
    EndFor
    Return a, b
EndAlgorithm
```

B.
```
Algorithm cauta(n, x, v):
    a ← -1
    b ← -1
    For i ← 1, n execute
        j ← cautareBinara(x[i], n, v)
        If j ≠ -1 then
            a ← i
            b ← j
        EndIf
    EndFor
    Return a, b
EndAlgorithm
```

C.
```
Algorithm cauta(n, x, v):
    a ← -1
    b ← -1
    i ← 1; j ← n
    While i ≤ n AND j > 0 execute
        If x[i][j] = v then
            a ← i
            b ← j
        EndIf
        If x[i][j] > v then
            j ← j - 1
        Else
            i ← i + 1
        EndIf
    EndWhile
    Return a, b
EndAlgorithm
```

D.
```
Algorithm cauta(n, x, v):
    a ← -1
    b ← -1
    i ← 1; j ← 1
    While i ≤ n AND x[i][j] < v execute
        i ← i + 1
    EndWhile
    While j ≤ n AND x[i][j] < v execute
        j ← j + 1
    EndWhile
    If x[i][j] = v then
        a ← i
        b ← j
    EndIf
    Return a, b
EndAlgorithm
```

13. Adott a 3 × 3 elemű ***M*** négyzetes mátrix. A következő programrészletek közül melyik implementálja helyesen a mátrix elemeinek trigonometrikus irányban való elforgatását 90 fokkal a mátrix (2, 2) pozíciója körül?

A.
```
For i ← 0, 1 execute
    X ← M[1][1]
    M[1][1] ← M[1][2]
    M[1][2] ← M[1][3]
    M[1][3] ← M[2][3]
    M[2][3] ← M[3][3]
    M[3][3] ← M[3][2]
    M[3][2] ← M[3][1]
    M[3][1] ← M[2][1]
    M[2][1] ← X
EndFor
```

B.
```
For i ← 0, 2 execute
    X ← M[1][1]
    M[1][1] ← M[1][2]
    M[1][2] ← M[1][3]
    M[1][3] ← M[2][3]
    M[2][3] ← M[3][3]
    M[3][3] ← M[3][2]
    M[3][2] ← M[3][1]
    M[3][1] ← M[2][1]
    M[2][1] ← X
EndFor
```

C.
```
For i ← 1, 2 execute
    X ← M[1][1]
    M[1][1] ← M[1][2]
    M[1][2] ← M[1][3]
    M[1][3] ← M[2][3]
    M[2][3] ← M[3][3]
    M[3][3] ← M[3][2]
    M[3][2] ← M[3][1]
    M[3][1] ← M[2][1]
    M[2][1] ← X
EndFor
```

D.
```
For i ← 1, 3 execute
    X ← M[1][1]
    M[1][1] ← M[1][i]
    M[1][i] ← M[1][3]
    M[1][3] ← M[i][3]
    M[i][3] ← M[3][3]
    M[3][3] ← M[3][i]
    M[3][i] ← M[3][1]
    M[3][1] ← M[i][1]
    M[i][1] ← X
EndFor
```

14. Adott a `rearanjeaza(x, n)` algoritmus, ahol ***n*** természetes szám ($1 \leq n \leq 200$) és ***x*** egy ***n*** különböző egész számot tartalmazó vektor (***x***[1], ***x***[2], ..., ***x***[***n***], $-100 \leq x[i] \leq 100$, ***i*** = 1, 2, ..., ***n***). Az `interschimba(x, i, j)` algoritmus felcseréli az ***x***[***i***] elemet az ***x***[***j***] elemmel.

```
Algorithm rearanjeaza(x, n):
    v ← x[n]
    i ← 0; j ← 1
    While j ≤ n - 1 execute
        If x[j] ≤ v then
            i ← i + 1
            interschimba(x, i, j)
        EndIf
        j ← j + 1
    EndWhile
    i ← i + 1
    interschimba(x, i, n)
    Return i
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Az algoritmus növekvő sorrendbe rendezi az ***x*** vektor elemeit.
B. Ha az ***x*** vektor növekvő sorrendbe rendezett, a vektor elemeinek sorrendje nem változik.
C. Az ***x*** vektor úgy lesz átrendezve, hogy attól az elemtől balra, amely eredetileg az utolsó volt, csak kisebb értékű elemek lesznek, tőle jobbra, csak nagyobb értékűek.
D. Az algoritmus visszatéríti az ***x*** vektor legkisebb elemének eredeti indexét.

15. Adott a `calcul(v, n)` algoritmus, ahol ***n*** természetes szám ($1 \leq n \leq 10^4$) és ***v*** egy ***n*** természetes számot tartalmazó vektor (***v***[1], ***v***[2], ..., ***v***[***n***], $1 \leq v[i] \leq 200$, ***i*** = 1, 2, ..., ***n***):

```
Algorithm calcul(v, n):
    If n = 1 then
        Return v[1]
    EndIf
    If v[1] MOD v[n] = 0 then
        v[1] ← v[n]
        n ← n - 1
        Return calcul(v, n)
    Else
        aux ← v[n]
        v[n] ← v[1] MOD v[n]
        v[1] ← aux
        Return calcul(v, n)
    EndIf
EndAlgorithm
```

A következő értékek közül melyek esetében térít vissza az algoritmus 12-t?

A. ***v*** = [60, 96, 120, 84], ***n*** = 4
B. ***v*** = [75, 24, 12, 84], ***n*** = 4
C. ***v*** = [75, 24, 49, 80], ***n*** = 4
D. ***v*** = [60, 24, 12, 84], ***n*** = 4

16. Adott a `ceFace(n)` algoritmus, ahol ***n*** egész szám ($-10^4 \leq n \leq 10^4$):

```
Algorithm ceFace(n):
    If n = 0 then
        Return "0"
    EndIf
    If n < 0 then
        Return "-" + ceFace(-n)
    EndIf
    If n MOD 3 = 0 then
        Return ceFace(n DIV 3) + "0"
    EndIf
    If n MOD 3 = 1 then
        Return ceFace(n DIV 3) + "1"
    EndIf
    Return ceFace(n DIV 3) + "2"
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha ***n*** a 3-nak valamely hatványa, a visszatérített karakterlánc egyetlen `"1"`-es karaktert tartalmaz.
B. ***n*** = 3-ra és ***n*** = -3-ra a `ceFace(n)` algoritmus azonos értékeket térít vissza.
C. Ha ***n*** = 82, az algoritmus a `"010001"`-et téríti vissza.
D. Ha ***n*** negatív szám, az algoritmus végtelen ciklusba kerül.

17. Adott a `decide(n, x)` algoritmus, ahol ***n*** természetes szám ($1 \le n \le 10^4$) és ***x*** egy ***n*** természetes számot tartalmazó vektor (x[1], x[2], ..., x[n], $0 \le x[i] \le 100$, $i = 1, 2, ..., n$).

```
Algorithm decide(n, x):
    a ← x[1]
    i ← 2; j ← 1
    While i ≤ n execute
        If x[i] = a then
            j ← j + 1
        Else
            If j > 0 then
                j ← j - 1
            Else
                a ← x[i]
                j ← 1
            EndIf
        EndIf
        i ← i + 1
    EndWhile
    i ← 1; j ← 0
    While i ≤ n execute
        If x[i] = a then
            j ← j + 1
        EndIf
        i ← i + 1
    EndWhile
    If j > (n DIV 2) then
        Return a
    Else
        Return -1
    EndIf
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha ***n*** = 5 és ***x*** = [1, 2, 1, 3, 1] az algoritmus 1-et térít vissza.
B. Ha ***n*** = 5 és ***x*** = [1, 2, 2, 3, 1] az algoritmus -1-et térít vissza.
C. Bármely bemeneti vektor esetében az algoritmus -1-et térít vissza.
D. Az algoritmus az ***x*** vektor első elemét téríti vissza.

18. Adott a `ceFace(n)` algoritmus, amely beolvas ***n*** számot, ahol ***n*** természetes szám ($1 \le n \le 10^9$):

```
Algorithm ceFace(n):
    nr ← 0
    Read a
    For i ← 2, n execute
        Read b
        If a ≠ b then
            nr ← nr + 1
        EndIf
        a ← b
    EndFor
    Return nr
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Az algoritmus azoknak a számoknak a darabszámát téríti vissza, amelyek ismétlődnek a beolvasott számok között (például, ha a számok: 3, 34, 34, 7, 3 akkor a visszatérített érték 2).
B. Az algoritmus annak a leghosszabb tömbszakasznak a hosszát téríti vissza, amelyben a beolvasott értékek azonosak (például, ha a számok 2, 34, 34, 34, 5 akkor a visszatérített érték 3).
C. Az algoritmus azoknak az egymásutáni számokból álló pároknak a darabszámát téríti vissza, amelyek különböző értékekből állnak (például, ha a számok 2, 34, 34, 7, akkor a (2, 34), (34, 7) elempárok egymásutániak és különböző értékekből állnak, tehát a visszatérített érték 2).
D. Az algoritmus azoknak az egymásutáni számokból álló pároknak a darabszámát téríti vissza, amelyek azonos értékekből állnak (például, ha a számok 2, 2, 3, 3, akkor a (2, 2), (3, 3) elempárok egymásutániak és azonos értékekből állnak, tehát a visszatérített érték 2).

19. Adott az `f(a)` algoritmus, ahol ***a*** természetes szám ($0 \le a \le 10^4$):

```
Algorithm f(a):
    n ← 0
    While a > 1 execute
        b ← 1
        While b ≤ a execute
            b ← 3 * b
            n ← n + 1
        EndWhile
        a ← a DIV 3
    EndWhile
    Return n
EndAlgorithm
```

Mit térít vissza az algoritmus, ha ***a*** = 81-re hívjuk meg?

A. 0
B. 14
C. 16
D. 9

20. Adott a `h(n, a)` algoritmus, ahol ***n*** természetes szám ($1 \leq \boldsymbol{n} \leq 10^3$) és ***a*** egy ***n*** elemű, egész számokat tartalmazó, növekvően rendezett vektor (***a***[1], ***a***[2], ..., ***a***[***n***], $-10^4 \leq \boldsymbol{a}[\boldsymbol{i}] \leq 10^4$, ***i*** = 1, 2, ..., ***n***).

A következő hívások közül melyik térít vissza 4-et?

A. `h(5, [1, 2, 3, 4, 5])`
B. `h(6, [2, 4, 6, 10, 18, 20])`
C. `h(7, [2, 2, 3, 4, 6, 9, 13])`
D. `h(5, [2, 2, 2, 4, 6])`

```
Algorithm h(n, a):
    t ← 0; i ← n
    While i > 2 execute
        k ← 1; j ← i – 1
        b ← a[i]
        While k < j execute
            If a[k] + a[j] = b then
                t ← t + 1
                k ← k + 1
                j ← j - 1
            Else
                If a[k] + a[j] < b then
                    k ← k + 1
                Else
                    j ← j - 1
                EndIf
            EndIf
        EndWhile
        i ← i – 1
    EndWhile
    Return t
EndAlgorithm
```

21. Adott az `f(x, n, m)` algoritmus, ahol ***n*** és ***m*** természetes számok ($1 \leq \boldsymbol{n}, \boldsymbol{m} \leq 10^4$) és ***x*** egy ***n*** elemű, természetes számokat tartalmazó vektor (***x***[1], ***x***[2], ..., ***x***[***n***], $1 \leq \boldsymbol{x}[\boldsymbol{i}] \leq 10^4$, ***i*** = 1, 2, ..., ***n***):

Mit térít vissza az algoritmus, ha `f(x, 9, 41)`, alakban hívjuk meg, ahol ***x*** = [41, 15, 5, 8, 10, 1, 16, 18, 19]?

A. 1 B. 3 C. 5 D. 7

```
Algorithm f(x, n, m):
    If m = 0 then
        Return 1
    EndIf
    If n = 0 then
        Return 0
    EndIf
    If x[n] > m then
        Return f(x, n - 1, m)
    Else
        Return f(x, n - 1, m) + f(x, n - 1, m – x[n])
    EndIf
EndAlgorithm
```

22. Adott a `select(v, x, n)` algoritmus, ahol ***n*** természetes szám ($1 \leq \boldsymbol{n} \leq 10^4$), ***v*** egy ***n*** elemű, egész számokat tartalmazó vektor (***v***[1], ***v***[2], ..., ***v***[***n***], $-100 \leq \boldsymbol{v}[\boldsymbol{i}] \leq 100$, ***i*** = 1, 2, ..., ***n***) és ***x*** egy egész szám, $-100 \leq \boldsymbol{x} \leq 100$:

A következő állítások közül melyek igazak?

A. A `select([0, 1, 2, 4, 5, 8, 9, 10, 7, 6], 10, 10)` hívás esetén az algoritmus 10-et térít vissza.
B. Az algoritmus akkor és csakis akkor téríti vissza az ***x*** elem pozícióját a ***v*** vektorban, ha a ***v*** vektor növekvően rendezett.
C. Az algoritmus bonyolultsága $O(\log_2 \boldsymbol{n})$.
D. A `select([0, 1, 2, 4, 5, 8, 9, 10, 7, 6], 7, 10)` hívás esetén az algoritmus -1-et térít vissza.

```
Algorithm select(v, x, n):
    i ← 1; j ← n
    While i ≤ j execute
        k ← (i + j) DIV 2
        If v[k] = x then
            Return k
        EndIf
        If v[i] ≤ v[k] then
            If v[i] ≤ x AND x < v[k] then
                j ← k - 1
            Else
                i ← k + 1
            EndIf
        Else
            If v[k] < x AND x ≤ v[j] then
                i ← k + 1
            Else
                j ← k - 1
            EndIf
        EndIf
    EndWhile
    Return -1
EndAlgorithm
```

23. Adott a `maiMare(n)` algoritmus, ahol ***n*** egy nemnulla természetes szám ($1 \le n < 10^6$), amelynek számjegyei különbözők. Az algoritmus azoknak a számoknak a darabszámát kellene visszatérítse, amelyek szigorúan nagyobbak, mint ***n*** és az ***n*** számjegyeiből állnak. Például, `maiMare(213) = 3`. Feltételezzük, hogy ***n*** nem kezdődik 0 számjeggyel és rendelkezésre állnak a következő algoritmusok, a specifikációknak megfelelően:

- `factorial(n)` – az ***n*** természetes szám faktoriálisát ($1 \le n \le 10$) téríti vissza
- `nrCifre(n)` – az ***n*** természetes szám számjegyeinek darabszámát téríti vissza ***n*** ($1 \le n < 10^6$)
- `imparte(n)` – visszatérít egy vektort, amelynek elemei az ***n*** szám számjegyei ($1 \le n < 10^6$), fordított sorrendben. Például: `imparte(1352)` a `[2, 5, 3, 1]` vektort téríti vissza.

```
Algorithm maiMare(n):
    cifre ← imparte(n)
    nrCif ← nrCifre(n)
    Return calculeaza(cifre, nrCif)
EndAlgorithm
```

```
1.  Algorithm calculeaza(v, n):
2.      If n < 2 then
3.          Return 0
4.      EndIf
5.      mm ← 0
6.      For i ← 1, n - 1 execute
7.          If v[i] > v[n] then
8.              mm ← mm + 1
9.          EndIf
10.     EndFor
11.     ...
12. EndAlgorithm
```

A következő utasítások közül melyiket kell a `calculeaza(v, n)` algoritmus 11-dik sorába írni?

A. `Return factorial(n) – ((n - mm - 1) * factorial(n - 1) + calculeaza(v, n - 1))`
B. `Return calculeaza(v, n - 1) * mm + factorial(n - 1)`
C. `Return (mm * factorial(n) + calculeaza(v, n - 1)) DIV n`
D. `Return calculeaza(v, n - 1) + mm * factorial(n - 1)`

24. Egy előadást egy adott I-es teremben kellett volna megtartani, de át kellett költöznie a II-es terembe, ahol a székek számozása különbözik. Mindkét teremben ***L*** széksor található ($2 \le L \le 50$), minden sort kettőbe oszt a sor közepén található folyosó, melynek mindkét oldalán ***K*** szék található ($2 \le K \le 50$) (tehát a teremben összesen 2 * ***K*** * ***L*** szék van). A II-es teremben minden helyet egyetlen szám azonosít be. A folyosó bal oldalán levő helyek páros számúak, és a székek számozása a színpad előtti sorban kezdődik. Tehát az első sorban levő székek számai (a folyosótól a terem széle fele haladva) 2, 4, 6 stb. Miután egy adott sorban megszámoztak minden széket, a következő sorban folytatódik a számozás kezdve a folyosó melletti székkel és a következő páros számmal. A folyosó jobb oldalán található helyeket hasonlóan számozták, de páratlan számokkal. Tehát, az első sorban levő székek számai 1, 3, 5 stb. a folyosótól a terem széle fele haladva.
Az I-es teremben minden helyhez három érték van rendelve. A sor sorszáma (***rand*** – egy 1 és ***L*** közötti szám – beleértve az 1-et és az ***L***-t is –, ahol az 1-es sor a színpad előtti), a folyosóhoz képest a hely iránya (***directie*** – „`stanga`” vagy „`dreapta`”) és a szék sorszáma a soron belül (***loc*** – egy 1 és ***K*** közötti szám, – beleértve az 1-et és a ***K***-t is –, ahol az 1-es szék a folyosó melletti). Mivel az előadásnak át kell költöznie, az I-es teremben érvényes jegyeken található helyeket (ezeket a ***rand***, ***loc***, ***directie*** jelöli) át kell alakítani úgy, hogy a II-es teremben érvényes helyeknek feleljen meg (egyetlen szám).
Melyek azok az algoritmusok, amelyek a ***K***, ***rand***, ***loc***, ***directie*** bemeneti adatok alapján elvégzik a kijelentésnek megfelelően helyesen az átalakítást (egy átalakítás akkor helyes, ha minden néző helye egyedi a II-es teremben)?

A.
```
Algorithm transforma(K, rand, loc, directie):
    If directie = "stanga" then
        rez ← 2 * (loc + K * (rand - 1))
    Else
        rez ← 2 * (loc + K * (rand - 1) + 1)
    EndIf
    Return rez
EndAlgorithm
```

B.
```
Algorithm transforma(K, rand, loc, directie):
    rez ← rand * (K – 1) * 2
    rez ← rez + (loc * 2)
    If directie = "dreapta" then
        rez ← rez – 1
    EndIf
    Return rez
EndAlgorithm
```

C.
```
Algorithm transforma(K, rand, loc, directie):
    rez ← (rand - 1) * K * 2
    rez ← rez + (loc * 2)
    If directie = "dreapta" then
        rez ← rez – 1
    EndIf
    Return rez
EndAlgorithm
```

D.
```
Algorithm transforma(K, rand, loc, directie):
    rez ← (rand - 1) * K * 2
    rez ← rez + (loc * 2)
    If directie = "dreapta" then
        rez ← rez + 1
    EndIf
    Return rez
EndAlgorithm
```

BABEŞ-BOLYAI TUDOMÁNYEGYETEM

MATEMATIKA ÉS INFORMATIKA KAR

MATEK-INFO verseny – 2024 április 20

Informatika írásbeli

JAVÍTÁSI KULCS & MEGOLDÁSOK

HIVATALBÓL: 10 pont

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