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

Felvételi verseny – 2022 szeptember 15.

Informatika írásbeli

FONTOS MEGJEGYZÉS:
Más pontosítások hiányában feltételezhetjük, hogy az aritmetikai műveleteket végtelen adattípusokon végezzük, vagyis nincs túlcsordulás és alulcsordulás.
Minden sorozatot/vektort 1-től számozunk.

1. Adott a `dönt(n, x)` algoritmus, ahol $\boldsymbol{n}$ természetes szám ($1 \leq \boldsymbol{n} \leq 10000$) és $\boldsymbol{x}$ egy $\boldsymbol{n}$ elemű, természetes számokat tároló vektor ($\boldsymbol{x}[1], \boldsymbol{x}[2], ..., \boldsymbol{x}[\boldsymbol{n}]$, $-100 \leq \boldsymbol{x}[\boldsymbol{i}] \leq 100$, ahol $\boldsymbol{i} = 1, 2, ..., \boldsymbol{n}$):

```
Algorithm dönt(n, x):
    b ← True
    i ← 1
    While b = True AND i < n execute
        If x[i] < x[i + 1] then
            b ← True
        else
            b ← False
        EndIf
        i ← i + 1
    EndWhile
    return b
EndAlgorithm
```

Mikor fog az algoritmus *True*-t téríteni vissza?

A. Ha az $\boldsymbol{x}$ vektor a következő elemeket tárolja: 1, 2, 3, …, 10
B. Ha az $\boldsymbol{x}$ vektor szigorúan növekvő.
C. Ha az $\boldsymbol{x}$ vektornak nincsenek negatív elemei.
D. Ha az $\boldsymbol{x}$ vektornak vannak pozitív elemei, amelyek a negatív elemek előtt találhatók.

2. Adott egy természetes szám, amelynek nincs 0-val egyenlő számjegye, és amelynek $\boldsymbol{n}$ darab számjegyét az $\boldsymbol{a}$ ($\boldsymbol{a}[1], \boldsymbol{a}[2], ..., \boldsymbol{a}[\boldsymbol{n}]$) sorozat tárolja ($1 \leq \boldsymbol{n} \leq 10$ az eredeti hívás pillanatában).
A következő algoritmusok közül melyik térít vissza *True*-t, ha a szám, amelyet így adtunk meg palindrom és *False*-t különben. Egy szám akkor palindrom, ha balról jobbra olvasva ugyanaz az értéke, mint ha jobbról balra olvassuk.

A.

```
Algorithm palindrom_1(a, n):
    i ← 1
    j ← n
    k ← True
    While (i ≤ j) AND (k = True) execute
        If a[i] = a[j] then
            i ← i + 1
            j ← j - 1
        else
            k ← False
        EndIf
    EndWhile
    return k
EndAlgorithm
```

B.

```
Algorithm eltolás(a, n):
    For i = 1, n - 1 execute
        a[i] ← a[i + 1]
    EndFor
EndAlgorithm

Algorithm palindrom_2(a, n):
    j ← n
    If (j = 0) OR (j = 1) then
        return True
    EndIf
    If a[1] = a[j] then
        eltolás(a, n)
        return palindrom_2(a, n - 2)
    EndIf
    return False
EndAlgorithm
```

C.

```
Algorithm palindrom_3(a, n):
    i ← n
    j ← 1
    k ← True
    sum1 ← 0
    sum2 ← 0
    While (i > n DIV 2) AND (j ≤ n DIV 2)
                                    execute
        sum1 ← sum1 + a[i]
        sum2 ← sum2 + a[j]
        i ← i - 1
        j ← j + 1
    EndWhile
    If sum1 = sum2 then
        k ← True
    else
        k ← False
    EndIf
    return k
EndAlgorithm
```

D.

```
Algorithm palindrom_4(a, n):
    i ← 1
    j ← n
    k ← True
    While (i ≤ j) AND (k = True) execute
        If (a[i] = a[j]) AND (i MOD 2 = 0)
                      AND (j MOD 2 = 0) then
            i ← i + 1
            j ← j - 1
        else
            k ← False
        EndIf
    EndWhile
    return k
EndAlgorithm
```

3. Adott az `F(n)` algoritmus, ahol ***n*** természetes szám ($1 \leq n \leq 10^9$).

```
Algorithm F(n):
    If n < 10 then
        return n
    EndIf
    u ← n MOD 10
    p ← F(n DIV 10)
    If u MOD 5 ≤ p MOD 5 then
        return u
    EndIf
    return p
EndAlgorithm
```

A következő állítások közül melyek helyesek?

A. Ha ***n*** = 812376, az algoritmus 6-ot térít vissza.
B. Ha ***n*** = 8237631, az algoritmus 1-et térít vissza.
C. Ha ***n*** = 4868, az algoritmus 8-at térít vissza.
D. Ha ***n*** = 51, az algoritmus 0-át térít vissza.

4. Adott az `f(n)` algoritmus, ahol az ***n*** paraméter természetes szám ($1 \leq n \leq 10^9$).

```
Algorithm f(n):
    v ← 0; z ← 0;
    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
        EndWhile
        If k > v then
            v ← k
            z ← c
        EndIf
    EndFor
    return z
EndAlgorithm
```

A következő állítások közül melyek helyesek?

A. Az algoritmus visszatéríti az ***n*** szám számjegyeinek darabszámát.
B. Az algoritmus visszatéríti az ***n*** szám legnagyobb értékkel rendelkező számjegye előfordulásainak darabszámát.
C. Az algoritmus visszatéríti az ***n*** szám leggyakrabban előforduló számjegyei közül az egyiket.
D. Az algoritmus visszatéríti az ***n*** szám leggyakrabban előforduló számjegyeinek számát.

5. A következő algoritmusok közül melyik írja ki a paraméterként megadott ***x*** természetes szám bináris alakját ($0 < x \leq 10^9$ az eredeti hívás pillanatában)?

A.
```
Algorithm imp(x):
    If x = 0 then
        r ← x MOD 2
        imp(x DIV 2)
        write r
    EndIf
EndAlgorithm
```
B.
```
Algorithm imp(x):
    If x ≠ 0 then
        r ← x MOD 2
        imp(x DIV 2)
        write r
    EndIf
EndAlgorithm
```
C.
```
Algorithm imp(x):
    If x = 0 then
        r ← x DIV 2
        imp(x DIV 2)
        write r
    EndIf
EndAlgorithm
```
D.
```
Algorithm imp(x):
    If x ≠ 0 then
        r ← x MOD 2
        imp(x)
        write r
    EndIf
EndAlgorithm
```

6. Melyek igazak az **5.** tétel változataira vonatkozó, következő állítások közül?
A. Az A változat algoritmusa a végrehajtás során nem ír ki semmit.
B. A B változat algoritmusa nem hívódik meg rekurzívan az ***x*** paraméter egyetlen megengedett értékére sem.
C. A C változat algoritmusa helyessé válna, ha az „`=`”-őt „`≠`”-re cserélnénk.
D. A D változat algoritmusa helyessé válna, ha az „`imp(x)`”-et „`imp(x DIV 2)`”-re cserélnénk.

7. Adottak az ***a*** és ***b*** egész számok ($-1000 \leq a, b \leq 1000$) és a `NOT ((a > 0) AND (b > 0))` kifejezés. Az alábbi kifejezések közül melyek ekvivalensek az előbb megadott kifejezéssel?

A. `(NOT (a < 0)) AND (NOT (b < 0))`
B. `(a ≤ 0) AND (b ≤ 0)`
C. `(NOT (a > 0)) OR (NOT (b > 0))`
D. `NOT ((a > 0) OR (b < 0))`

8. Adott az `s(n)` algoritmus, ahol ***n*** természetes szám ($2 \leq n \leq 10$). A / műveleti jel a valós osztást jelöli (például, 3 / 2 = 1,5).

```
Algorithm s(n):
    p ← 1
    x ← 0
    For k = 0, n - 1 execute
        p ← p * (k + 1)
        x ← x + 1 / p
    EndFor
    return x
EndAlgorithm
```

Az alábbi összegek közül, melyiket téríti vissza az algoritmus?

A. $\sum_{k=0}^{n} \frac{1}{k!}$
B. $\sum_{k=0}^{n} \frac{1}{k}$
C. $\sum_{k=0}^{n-1} \frac{1}{k!}$
D. $\sum_{k=1}^{n} \frac{1}{k!}$

9. Adott a `mitCsinál(n)` algoritmus, ahol ***n*** pozitív természetes szám ($1 \leq n \leq 10000$).

```
Algorithm mitCsinál(n):
    m ← 0
    p ← 10
    While p < n execute
        r ← n MOD p
        m ← m + r
        p ← p * 10
    EndWhile
    return m
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha ***n*** = 125 az algoritmus 521-et térít vissza.
B. A `mitCsinál(n)` algoritmus az ***n*** szám tükrözöttjét téríti vissza.
C. Ha ***n*** = 125 az algoritmus 155-öt térít vissza.
D. Ha ***n*** = 340 az algoritmus 40-et térít vissza.

10. Adott az `f(v, n)` algoritmus, ahol ***n*** nullától különböző természetes szám ($1 \leq n \leq 10000$) és ***v*** egy ***n*** elemű, pozitív természetes számokból álló vektor (***v***[1], ***v***[2], ..., ***v***[***n***]). Feltételezzük, hogy a `prim(d)` algoritmus *True*-t térít vissza, ha ***d*** (természetes szám) prímszám és *False*-t különben.

```
Algorithm f(v, n):
    x ← 1
    a ← 0
    For i ← 1, n execute
        For d ← 2, (v[i] DIV 2) execute
            If (prim(d) = True) AND (v[i] MOD d = 0) then
                x ← x * d
            EndIf
        EndFor
    EndFor
    For d ← 2, (x DIV 2) execute
        If (x MOD d = 0) AND (prim(d) = True) then
            a ← a + 1
        EndIf
    EndFor
    return a
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Az algoritmus a ***v*** vektorban található összes szám különböző valódi prímosztóinak darabszámát téríti vissza.
B. Az algoritmus a ***v*** vektorban található számok prímosztóinak szorzatát téríti vissza.
C. Az algoritmus a ***v*** vektorban található prímszámok darabszámát téríti vissza.
D. Az algoritmus a ***v*** vektorban található számok összes osztójának darabszámát térít vissza.

11. Adott az `f(n)` algoritmus, ahol ***n*** természetes szám ($0 < n \leq 10^9$ a hívás pillanatában). A ***v*** lokális változó vektor.

```
Algorithm f(n):
    m ← 0
    While n > 0 execute
        m ← m + 1
        v[m] ← n MOD 10
        n ← n DIV 10
    EndWhile
    x ← 0
    mx ← 0
    While mx > -1 execute
        x ← x * 10 + mx
        mx ← -1
        j ← 1
        For i = 1, m execute
            If v[i] > mx then
                j ← i
                mx ← v[i]
            EndIf
        EndFor
        v[j] ← -1
    EndWhile
    return x
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Az algoritmus azt a legnagyobb számot téríti vissza, amely az ***n*** szám számjegyeiből építhető fel.
B. Az algoritmus 10-nek azt a legnagyobb hatványát téríti vissza, amely osztója az ***n*** számnak.
C. Az algoritmus az ***n*** szám legbaloldalibb számjegyét téríti vissza.
D. Az algoritmus az ***n*** szám számjegyeinek összegét téríti vissza.

12. Adott az `f(n)` algoritmus, ahol az ***n*** paraméter természetes szám ($1 \leq \boldsymbol{n} \leq 1000^2$ a hívás pillanatában).

```
Algorithm f(n):
    z ← 0; p ← 1;
    While n ≠ 0 execute
        c ← n MOD 10
        n ← n DIV 10
        If c MOD 3 = 0 then
            z ← z + p * (9 – c)
            p ← p * 10
        EndIf
    EndWhile
    return z
EndAlgorithm
```

Mit fog visszatéríteni az algoritmus, ha az ***n*** = 103456 értékre hívjuk meg?

A. 639 B. 963 C. 693 D. 369

13. Tekintsük a **12.** tételben szereplő `f(n)` algoritmust, de most az ***n*** paraméter egy kétjegyű természetes szám ($10 \leq \boldsymbol{n} \leq 99$ a hívás pillanatában).
A következő változatok között melyek tartalmaznak csak olyan számokat, amelyeknek esetében az algoritmus a 3-as értéket téríti vissza?

A. 61, 65, 67
B. 62, 66, 68
C. 16, 56, 76
D. 26, 66, 86

14. Adott a `mitCsinál(a, b)` algoritmus, ahol ***a*** és ***b*** pozitív természetes számok (1 ≤ ***a***, ***b*** ≤ 10000).

```
Algorithm mitCsinál(a, b):
    For i ← 2, a, 2 execute
        If a MOD i = 0 then
            If b MOD i = 0 then
                Write i
                Write new line
            EndIf
        EndIf
    EndFor
EndAlgorithm
```

Ha ***a*** = 600, a ***b*** mely értékeire ír ki az algoritmus 4 darab számot, ha végrehajtjuk a `mitCsinál(a, b)` algoritmust?

A. ***b*** = 20 B. ***b*** = 50 C. ***b*** = 12 D. ***b*** = 90

15. A **14.** tételben szereplő algoritmusra vonatkozóan, a következő állítások közül melyek igazak?

A. Az algoritmus az ***a*** és ***b*** számok közös osztóit írja ki.
B. Az algoritmus az ***a*** és ***b*** számok közös valódi osztóit írja ki.
C. Az algoritmus az ***a*** és ***b*** számok közös páratlan osztóit írja ki.
D. Az algoritmus az ***a*** és ***b*** számok közös páros osztóit írja ki.

16. Legyen egy program, amely növekvő sorrendben generál minden, pontosan 5 különböző számjegyből álló természetes számot, amelyek a 2, 3, 4, 5, 6 számjegyekből építhetők fel.

Adjátok meg azt a két számot, amelyet a program közvetlenül az alábbi sorozat előtt és közvetlenül utána generál: 34256, 34265, 34526, 34562.

A. 32645 és 34625 B. 32654 és 34655 C. 32654 és 34625 D. 32645 és 34655

17. Legyen az ***x*** = (1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, ...) sorozat, amely az észrevehető szabálynak megfelelően folytatódik.

Tudva, hogy a sorozat első eleme az 1. pozíción található, az alábbi tömbszakaszok közül melyik tartalmaz csak **11**-eseket?

A. ***x***[100], ..., ***x***[109] B. ***x***[113], ..., ***x***[120] C. ***x***[140], ..., ***x***[152] D. ***x***[123], ..., ***x***[132]

18. A **17.** tételben leírt ***x*** sorozat első 100 eleme között hány prímszám található?

A. 4 B. 34 C. 36 D. 30

19. Adottak az ***a*** és ***n*** természetes számok (1 ≤ ***a***, ***n*** ≤ 1000), valamint az ***n*** elemű, természetes számokat tároló ***V*** vektor (***V***[1], ***V***[2], ..., ***V***[***n***]) és a `one(a, n, V)` és `two(a, n, V)` algoritmusok:

```
Algorithm one(a, n, V):
    p ← 1; i ← 1;
    While (i ≤ n) AND (a > V[p]) execute
        p ← p + 1
        i ← i + 1
    EndWhile
    return p
EndAlgorithm
```

```
Algorithm two(a, n, V):
    p ← 1; i ← 1;
    While i ≤ n execute
        If a > V[i] then
            p ← p + 1
        EndIf
        i ← i + 1
    EndWhile
    return p
EndAlgorithm
```

Mely tulajdonsággal rendelkezhet a ***V*** vektor ahhoz, hogy bármely ***n*** és adott tulajdonságú ***V*** esetében a két algoritmus azonos értéket térítsen vissza tetszőleges ***a*** értékre?

A. A *V* vektor minden eleme egyenlő.
B. A *V* vektor minden eleme különböző és növekvően rendezett.
C. A *V* vektor minden eleme különböző és csökkenően rendezett.
D. A *V* vektor növekvően rendezett, de az elemek nem feltétlenül különbözők.

20. Adott az `összeg(n)` algoritmus, ahol ***n*** természetes szám ($0 < n \leq 10000$ az eredeti hívás pillanatában).

```
Algorithm összeg(n):
    If n = 0 then
        return 0
    else
        return összeg(n - 1) + n DIV (n + 1) + (n + 1) DIV n
    EndIf
EndAlgorithm
```

A következő állítások közül melyek **NEM** igazak?
A. Az algoritmus $n + 1$ értékét téríti vissza.
B. Az algoritmus kiszámolja és visszatéríti az ***n*** szám valódi osztóinak összegét.
C. Az `összeg(1)` hívás a 2-es értéket téríti vissza.
D. Az algoritmus kiszámolja és visszatéríti az első ***n*** természetes szám átlaga egész részének kétszeresét.

21. Legyen a következő algoritmus, ahol az ***a*** és ***b*** természetes számok bemeneti paraméterek ($0 \leq a, b \leq 10000$ az eredeti hívás pillanatában):

```
Algorithm mitCsinál(a, b):
    While a * b ≠ 0 execute
        If a > b then
            return mitCsinál(a MOD b, b)
        else
            return mitCsinál(a, b MOD a)
        EndIf
    EndWhile
    return a + b
EndAlgorithm
```

A következő állítások közül melyek igazak?
A. Az algoritmus az ***a*** és ***b*** számok összegét téríti vissza.
B. Ha az algoritmust `mitCsinál(x, 0)` vagy `mitCsinál(0, x)` formában hívjuk meg, a nullától különböző ***x*** számot téríti vissza, illetve 0-át, ha a hívás `mitCsinál(0, 0)` alakú.
C. Az algoritmus az ***a*** és ***b*** számok legnagyobb közös osztóját téríti vissza.
D. Az algoritmus visszatéríti az ***a*** a ***b.*** hatványon értékét.

22. Adott a `kiÍr(n)` algoritmus, ahol ***n*** természetes szám ($1 \leq n \leq 10^9$):

```
Algorithm kiÍr(n):
    For i = 1, n – 1 execute
        For j = i + 1, n execute
            If (j – i) < (n DIV 2) then
                Write i, " ", j – i
                Write new line
            else
                If (j – i) ≠ (n DIV 2) then
                    Write j – i, " ", i
                    Write new line
                EndIf
            EndIf
        EndFor
    EndFor
EndAlgorithm
```

Hány számpárt fog kiírni az algoritmus, ha $n = 7$?

A. 21 B. 15 C. 11 D. 17

23. Adott az alábbi kódrészlet. Adjátok meg, hogy hányszor lesz kiírva az `'UBB'` karakterlánc, tudva, hogy $\boldsymbol{n} = 3^{k}$, ahol $\boldsymbol{k}$ természetes szám ($1 \le \boldsymbol{k} \le 30$)?

```
j ← n
While j > 1 execute
    i ← 1
    While i ≤ n execute
        i ← 3 * i
        Write 'UBB'
    EndWhile
    j ← j DIV 3
EndWhile
```

A. $\boldsymbol{k}^2$
B. $\boldsymbol{k} * 3^{k}$
C. $\boldsymbol{k} * (\boldsymbol{k} + 1)$
D. $3 * \boldsymbol{k}$

24. Adottak az alábbi kódrészletek és az ***i***, ***j***, ***a***, ***b*** természetes számok ($1 < \boldsymbol{a}, \boldsymbol{b} \le 10^9$).

1. kódrészlet (S1)

```
i ← 1
While i ≠ b execute
    j ← 1
    While j ≠ a execute
        Write '*'
        j ← j + 1
    EndWhile
    i ← i + 1
EndWhile
```

2. kódrészlet (S2)

```
i ← 1
While i ≠ a execute
    j ← 1
    While j ≠ b execute
        Write '*'
        j ← j + 1
    EndWhile
    i ← i + 1
EndWhile
```

A következő állítások közül melyek igazak?

A. Az **S1** kódrészlet által kiírt karakterek száma különbözik az **S2** kódrészlet által kiírt karakterek számától.
B. A két kódrészlet időbonyolultsága azonos.
C. Az **S1** kódrészlet $(\boldsymbol{a} - 1) * (\boldsymbol{b} - 1)$ karaktert ír ki.
D. Az **S2** kódrészlet $\boldsymbol{a} * \boldsymbol{b}$ karaktert ír ki.

25. Adott a `mitCsinál(szám)` algoritmus, ahol ***szám*** természetes szám ($100 \le \textbf{\textit{szám}} \le 2*10^9$ a hívás pillanatában).

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

```
Algorithm mitCsinál(szám):
    s ← 0
    c1 ← szám MOD 10
    szám ← szám DIV 10
    c2 ← szám MOD 10
    szám ← szám DIV 10
    While szám ≠ 0 execute
        c3 ← szám MOD 10
        t ← c3 * 100 + c2 * 10 + c1
        If tulajdonság(t) then
            s ← s + c1 + c2 + c3
        EndIf
        c1 ← c2
        c2 ← c3
        szám ← szám DIV 10
    EndWhile
    return s
EndAlgorithm
```

Adjátok meg azt értéket, amit a `mitCsinál(szám)` algoritmus visszatérít, ha ***szám*** = 1271211312.

A. 31　　B. 32　　C. 33　　D. 34

26. A következő algoritmusok közül melyik határozza meg helyesen és téríti vissza az ***n*** természetes szám ($0 < n < 10^5$) négyzetgyökének értékét a legközelebbi egész számra lefele kerekítve. A / műveleti jel a valós osztást jelöli (például: 3 / 2 = 1,5).

A.
```
Algorithm négyzetgyök_A(n):
    x ← 0
    z ← 1
    While z ≤ n execute
        x ← x + 1
        z ← z + 2 * x
        z ← z + 1
    EndWhile
    return x
EndAlgorithm
```

B.
```
Algorithm négyzetgyök_B(n):
    s ← 1
    d ← n DIV 2
    While s < d execute
        k ← (s + d) DIV 2
        If k * k ≥ n then
            d ← k
        else
            s ← k + 1
        EndIf
    EndWhile
    If s * s ≤ n then
        return s + 1
    else
        return s - 1
    EndIf
EndAlgorithm
```

C.
```
// Az algoritmust először
// négyzetgyök_C(n, n) alakban hívjuk meg
Algorithm négyzetgyök_C(n, x):
    eps ← 0.001
    y ← 0.5 * (x + n / x)
    If x - y < eps then
        // x egész részét térítjük vissza:
        return [x]
    EndIf
    return négyzetgyök_C(n, y)
EndAlgorithm
```

D.
```
Algorithm négyzetgyök_D(n):
    s ← 0
    p ← 0
    k ← 0
    While k < n execute
        k ← k + 3 + p
        p ← p + 2
        s ← s + 1
    EndWhile
    return s
EndAlgorithm
```

27. Tudva, hogy ***x*** természetes szám, az alábbi kifejezések közül melyeknek az értéke akkor és csakis akkor *True*, ha ***x*** olyan páros szám, amely **NEM** tartozik a (10, 20) nyílt intervallumhoz?

A. `NOT((x > 10) AND (x < 20)) AND (NOT (x MOD 2 = 1))`
B. `(x MOD 2 = 0) AND ((x < 10) OR (x > 20))`
C. `NOT(x MOD 2 = 1) AND ((x > 10) AND (x < 20))`
D. `NOT((x MOD 4 = 1) OR (x MOD 4 = 3) OR ((x > 10) AND (x < 20)))`

28. Adott az ***n*** különböző természetes számot tároló, szigorúan növekvő ***a*** sorozat (***a***[1], ***a***[2], …, ***a***[***n***], $2 \leq n \leq 1000$). Egy sorozatban *lokális csúcs*nak nevezünk egy olyan számot, amely szigorúan nagyobb, mint az előző pozíción levő szám, valamint a következő pozíción található szám. A sorozat első és utolsó eleme nem lehet lokális csúcs. Azt szeretnénk, hogy az `átrendezés(a, n)` algoritmus rendezze át a sorozatban található értékeket úgy, hogy a sorozatnak maximális számú lokális csúcsa legyen, majd térítse vissza az új sorozatot. A ***b*** lokális változó sorozat. A következő algoritmusok közül melyek helyesek?

A.

```
Algorithm átrendezés(a, n):
    i ← n
    For p ← 2, n, 2 execute
        b[p] ← a[i]
        i ← i - 1
    EndFor
    For p ← 1, n, 2 execute
        b[p] ← a[i]
        i ← i - 1
    EndFor
    return b
EndAlgorithm
```

B.

```
Algorithm átrendezés(a, n):
    i ← n
    For p ← 2, n, 2 execute
        b[p] ← a[i]
        i ← i - 1
        b[p - 1] ← a[i]
        i ← i - 1
    EndFor
    If n MOD 2 = 1 then
        b[n] ← a[i]
    EndIf
    return b
EndAlgorithm
```

C.

```
Algorithm átrendezés(a, n):
    i ← n
    For p ← 2, n, 2 execute
        b[p] ← a[i]
        i ← i - 1
    EndFor
    i ← 1
    For p ← 1, n, 2 execute
        b[p] ← a[i]
        i ← i + 1
    EndFor
    return b
EndAlgorithm
```

D.

```
Algorithm átrendezés(a, n):
    i ← n
    For p ← 2, n, 3 execute
        b[p] ← a[i]
        i ← i - 1
        b[p - 1] ← a[i]
        i ← i - 1
        If p + 1 ≤ n then
            b[p + 1] ← a[i]
            i ← i - 1
        EndIf
    EndFor
    If n MOD 3 = 1 then
        b[n] ← a[i]
    EndIf
    return b
EndAlgorithm
```

29. Adott az `f(n, p1, p2)` algoritmus. ahol ***n***, ***p*1** és ***p*2** szigorúan pozitív természetes számok ($1 < n, p1, p2 \le 10^4$ a hívás pillanatában).

```
Algorithm f(n, p1, p2):
    c ← 0
    While p1 ≤ n execute
        c ← c + n DIV p1
        p1 ← p1 * p2
    EndWhile
    return c
EndAlgorithm
```

A következő állítások közül melyek igazak?

A. Ha a három paraméter értéke egyenlő ($n = p1 = p2$), az algoritmus mindig 1-et térít vissza.
B. Ha $p1 = 5$ és $p2 = 5$, az algoritmus az $n!$ végén található 0 számjegyek darabszámát téríti vissza.
C. Ha ***p*1** egyenlő ***p*2**-vel és az értékük nagyobb, mint 2, az algoritmus a $[\log_{p1} n]$ értékét téríti vissza.
D. A fenti három állítás közül egyik sem igaz.

30. A következő algoritmusok közül melyik téríti vissza az [***a***, ***b***] ($0 < \boldsymbol{a} < \boldsymbol{b} < 10^6$) intervallumhoz tartozó *szummatív* számok darabszámát? Egy 0-tól különböző ***n*** természetes szám *szummatív*, ha $\boldsymbol{n}^2$ felírható ***n*** darab egymás utáni 0-tól különböző természetes szám összegeként. Például, 1 és 7 *szummatív*, mivel 1 = 1, illetve 49 = 4 + 5 + 6 + 7 + 8 + 9 + 10.

A.

```
Algorithm szummatív(a, b):
    k ← 0
    For i ← a, b execute
        If i MOD 2 ≠ 0 then
            k ← k + 1
        EndIf
    EndFor
    return k
EndAlgorithm
```

B.

```
Algorithm szummatív(a, b):
    return (b - a) DIV 2 + (b - a) MOD 2
                        + (a MOD 2 + b MOD 2) DIV 2
EndAlgorithm
```

C.

```
Algorithm szummatív(a, b):
  k ← 0
  For i ← a, b execute
    i2 ← i * i
    For j ← 2, i – 1 execute
      If i2 = j * i + (i * (i + 1) DIV 2) then
        k ← k + 1
      EndIf
    EndFor
  EndFor
  return k
EndAlgorithm
```

D.

```
Algorithm szummatív(a, b):
  k ← 0
  For i ← a, b execute
    i2 ← i * i
    For j ← 2, i DIV 2 execute
      If i2 = j * i + (i * (i + 1) DIV 2) then
        k ← k + 1
      EndIf
    EndFor
  EndFor
  return k
EndAlgorithm
```