BABEŞ-BOLYAI TUDOMÁNYEGYETEM
MATEMATIKA ÉS INFORMATIKA KAR

Felvételi verseny – 2021 szeptember 9.

Informatika írásbeli

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

```
Algoritmus mitCsinál(n):
   szám ← 0
   Minden d ← 1, n végezd el
      Ha n MOD d = 0 akkor
         szám ← szám + 1
      vége(ha)
   vége(minden)
   Ha szám = 2 akkor
      visszatérít igaz
   különben
      visszatérít hamis
   vége(ha)
Vége(algoritmus)
```

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. Az algoritmus *igaz* értéket térít vissza, ha ***n*** páratlan szám.
B. Az algoritmus *igaz* értéket térít vissza, ha ***n*** páros szám.
C. Az algoritmus *igaz* értéket térít vissza, ha ***n*** prímszám.
D. Az algoritmus *igaz* értéket térít vissza, ha ***n*** négyzetszám.

2. Tudjuk, hogy $x < y$ (x és y valós számok). A következő kifejezések közül melyeknek az értéke akkor és csakis akkor *igaz*, ha a ***t*** változó értéke (***t*** valós szám) NEM tartozik az (x, y) intervallumhoz?

A. `(t > x) VAGY (t < y)`
B. `(t ≤ x) VAGY (t ≥ y)`
C. `(t ≤ x) ÉS (t ≥ y)`
D. `(t > x) ÉS (t < y)`

3. Legyen az `f(n)` algoritmus, ahol ***n*** természetes szám ($1 \leq n \leq 10000$).

```
Algoritmus f(n):
   r ← 0
   Amíg n > 0 végezd el
      r ← r + (n MOD 10) * (n MOD 2)
      n ← n DIV 10
   vége(amíg)
   visszatérít r
Vége(algoritmus)
```

Válasszátok ki, mivel kellene helyettesíteni az alábbi algoritmusban az aláhúzott üres részt ahhoz, hogy a két algoritmus mindig ugyanazt az értéket térítse vissza?

```
Algoritmus fr(n):
   Ha n > 0 akkor
      visszatérít _________________
   vége(ha)
   visszatérít 0
Vége(algoritmus)
```

A. `(n MOD 2) * (n MOD 10) + fr(n DIV 10)`
B. `(n MOD 2) * (n MOD 10) * fr(n DIV 10)`
C. `(n MOD 10) + fr(n DIV 10)`
D. `(n MOD 2) * (n MOD 10) + fr(n MOD 10)`

4. Legyen az `f(n)` algoritmus, ahol ***n*** természetes szám (1 ≤ ***n*** ≤ 10000).

```
Algoritmus f(n):
       Minden i ← 1, n végezd el
           Minden j ← 1, 2 * i - 1 végezd el
              Ki: '*'
           vége(minden)
       vége(minden)
Vége(algoritmus)
```

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. ***n*** = 3-ra az algoritmus 3 csillagot ír ki
B. ***n*** = 3-ra az algoritmus 9 csillagot ír ki
C. Ahhoz, hogy az algoritmus 1154 csillagot írjon ki, ***n*** értéke 34 kell legyen
D. Ahhoz, hogy az algoritmus 289 csillagot írjon ki, ***n*** értéke 17 kell legyen

5. Az alábbi algoritmusnak bemeneti paraméterei az ***n*** egész szám (2 ≤ ***n*** ≤ 10000) és az ***n*** elemű ***v*** sorozat, amelynek elemei természetes számok (***v***[1], ***v***[2], …, ***v***[***n***]). A `/` műveleti jel valós osztást jelent (például `3 / 2 = 1,5`). A ***v*** sorozatban található legalább egy páros és legalább egy páratlan szám.

```
Algoritmus fn(v, n):
   a ← 0
   b ← 0
   Minden i ← 1, n végezd el
      Ha v[i] MOD 2 = 0 akkor
         a ← a + v[i]
         b ← b + 1
      vége(ha)
   vége(minden)
   visszatérít a / b
Vége(algoritmus)
```

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. Az algoritmus visszatéríti a ***v*** sorozat páros értékű elemeinek darabszámát
B. Az algoritmus visszatéríti a ***v*** sorozat páros értékű elemeinek átlagát
C. Az algoritmus visszatéríti a ***v*** sorozat páros értékű elemeinek összegét
D. Az algoritmus visszatéríti a ***v*** sorozat páratlan értékű elemeinek átlagát

6. Az alábbi algoritmusnak bemeneti paraméterei az ***n*** egész szám (1 ≤ ***n*** ≤ 10000) és az ***n*** elemű ***v*** sorozat, amelynek elemei természetes számok (***v***[1], ***v***[2], …, ***v***[***n***]).

```
Algoritmus fn(v, n):
   a ← 0
   Minden i ← 1, n végezd el
      ok ← 1
      b ← v[i]
      Amíg (b ≠ 0) ÉS (ok = 1) végezd el
         Ha b MOD 2 = 0 akkor
            ok ← 0
         vége(ha)
         b ← b DIV 10
      vége(amíg)
      Ha ok = 1 akkor
         a ← a + v[i]
      vége(ha)
   vége(minden)
   visszatérít a
Vége(algoritmus)
```

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. Az algoritmus visszatéríti a ***v*** sorozat páratlan értékű elemeinek összegét
B. Az algoritmus visszatéríti a ***v*** sorozat azon elemeinek összegét, amelyek 2-nek hatványai
C. Az algoritmus visszatéríti a ***v*** sorozat azon elemeinek összegét, amelyek csak páros számjegyekből állnak
D. Az algoritmus visszatéríti a ***v*** sorozat azon elemeinek összegét, amelyek csak páratlan számjegyekből állnak

7. Az alábbi algoritmusok közül melyik számítja ki egy egész szám moduluszát (abszolút értékét)? Feltételezzük, hogy egy logikai kifejezés értéke 1, ha a kifejezés igaz és 0, ha hamis.

A.
```
Algoritmus modulusz(n):
   visszatérít n * (-2 * (n < 0) + 1)
Vége(algoritmus)
```

B.
```
Algoritmus modulusz(n):
   Ha n < 0 akkor
      visszatérít n * (-1)
   vége(ha)
   visszatérít n
Vége(algoritmus)
```

C.
```
Algoritmus modulusz(n):
   Ha n < 0 akkor
      visszatérít n * (-1)
   különben
      visszatérít n
   vége(ha)
Vége(algoritmus)
```

D.
```
Algoritmus modulusz(n):
   Ha n > 0 akkor
      visszatérít n * (-1)
   különben
      visszatérít n
   vége(ha)
Vége(algoritmus)
```

8. Mi lesz az alábbi kifejezés értéke, ha ***x*** = 15 és ***y*** = 17?

```
(NEM (x MOD 10 = 0)) ÉS (y MOD 2 = 0) ÉS (x < y)
```

A. `igaz` B. `hamis` C. Hiba D. A kifejezést nem lehet kiértékelni

9. Legyen a `mitCsinál(n, i)` rekurzív algoritmus, ahol ***n*** természetes szám (2 ≤ ***n*** ≤ 1000).

```
Algoritmus mitCsinál(n, i):
   Ha i = 1 akkor
      visszatérít i
   különben
      Ha n MOD i = 0 akkor
         visszatérít i + mitCsinál (n, i - 1)
      különben
         visszatérít mitCsinál (n, i - 1)
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

Ha az algoritmust a `mitCsinál(n, n)` utasítással hívjuk meg, döntsétek el, hogy a következő állítások közül melyek igazak:

A. Az algoritmus visszatéríti az ***n*** szám legnagyobb osztója után következő számot
B. Az algoritmus visszatéríti a nem prím természetes számoknak az összegét, ***n***-nel bezárólag
C. Az algoritmus visszatéríti az ***n*** szám valódi osztóinak összegét
D. Az algoritmus visszatéríti az ***n*** szám valódi és nem valódi osztóinak összegét

10. A `bűvös(s, n)` algoritmusnak bemeneti paraméterei az ***n*** egész szám (1 ≤ ***n*** ≤ 10000) és az ***n*** elemű ***s*** karakterlánc (***s***[1], ***s***[2], …, ***s***[***n***]).

```
Algoritmus bűvös(s, n):
   i ← 1
   f ← 1
   Amíg i ≤ n DIV 2 végezd el
      Ha s[i] ≠ s[n - i + 1] akkor
         f ← 0
      vége(ha)
      i ← i + 1
   vége(amíg)
   visszatérít f
Vége(algoritmus)
```

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. Az algoritmus 1-et térít vissza, ha ***s*** páros darabszámú karakterből áll.
B. Az algoritmus 1-et térít vissza, ha ***s*** páratlan darabszámú karakterből áll.
C. Az algoritmus 1-et térít vissza, ha ***s*** palindrom.
D. Az algoritmus 1-et térít vissza, ha ***s*** csak különböző karakterekből áll.

11. A következő kifejezések közül, melyeknek az értéke akkor és csakis akkor *igaz*, ha ***x*** páratlan negatív szám? |***x***|-szel az ***x*** szám moduluszát (abszolút értékét) jelöljük.

A. `(|x| MOD 2 = 1) ÉS (x < 0)`
B. `NEM ((|x| MOD 2 = 0) ÉS (x ≥ 0))`
C. `NEM ((|x| MOD 2 = 0) VAGY (x ≥ 0))`
D. `(|x| MOD 2 ≠ 0) VAGY (x < 0)`

12. Legyen a `mitCsinál(n)` algoritmus, ahol ***n*** természetes szám ($0 \leq n \leq 10000$).

```
Algoritmus mitCsinál(n):
   s ← 0
   Amíg n > 0 végezd el
      c ← n MOD 10
      Ha c MOD 2 ≠ 0 akkor
         s ← s + c
      vége(ha)
      n ← n DIV 10
   vége(amíg)
   visszatérít s
Vége(algoritmus)
```

Mit fog visszatéríteni a `mitCsinál(1234)` hívás?

A. 4   B. 10   C. 60   D. 0

13. Adott egy karakterlánc és az ***f*** függvény, amelynek bemeneti paramétere egy karakter és amely 1-et térít vissza, ha az illető karakter számjegy, különben 0-át. A következő megközelítések közül melyik dönti el, hogy a karakterlánc csak számjegyeket tartalmaz-e?

A. Megvizsgáljuk, hogy az ***f*** függvény, a karakterlánc minden karakterére 1-et térít-e vissza.
B. Meghívjuk az ***f*** függvényt a karakterlánc minden karakterére, majd megvizsgáljuk, hogy a visszatérített értékek összege egyenlő-e a karakterlánc hosszával.
C. Meghívjuk az ***f*** függvényt a karakterlánc minden karakterére, majd megvizsgáljuk, hogy a visszatérített értékek között szerepel-e legalább egy 1-es.
D. Meghívjuk az ***f*** függvényt a karakterlánc egy-egy véletlenszerűen kiválasztott karakterére, ameddig annyiszor nem térít vissza 1-et, amennyi a karakterlánc hossza.

14. Az alábbi algoritmusok közül melyek implementálhatók úgy, hogy időbonyolultságuk lineáris legyen ($O(\boldsymbol{n})$)?

A. Egy elem szekvenciális keresése egy ***n*** számot tartalmazó vektorban
B. Egy ***n*** számot tartalmazó egydimenziós tömb rendezése beszúró rendezéssel
C. A legnagyobb érték megkeresése egy ***n*** számot tartalmazó nem rendezett vektorban
D. Egy ***n*** soros és ***n*** oszlopos négyzetes mátrix főátlóján található elemek összegének kiszámítása

15. Legyen az `f(a, b)` algoritmus, ahol ***a*** és ***b*** természetes számok ($1 \leq \boldsymbol{a}, \boldsymbol{b} \leq 10000$).

```
Algoritmus f(a, b):
   m ← a
   Amíg b MOD m > 0 végezd el
      m ← m + 1
   vége(amíg)
   visszatérít m
Vége(algoritmus)
```

Melyek azok a hívások, amelyeknek eredményeképpen az **`Amíg`** törzse legtöbb egyszer hajtódik végre?

A. `f(10, 11)` B. `f(10, 10)` C. `f(10, 9)` D. `f(10, 15)`

16. Legyen az `f(a, b)` algoritmus, ahol ***a*** és ***b*** természetes számok ($1 \leq \boldsymbol{a}, \boldsymbol{b} \leq 10000$).

```
Algoritmus f(a, b):
   c ← 0
   d ← 0
   p ← 1
   Amíg a + b + c > 0 végezd el
      c ← a MOD 10 + b MOD 10 + c
      d ← d + (c MOD 10) * p
      p ← p * 10
      a ← a DIV 10
      b ← b DIV 10
      c ← c DIV 10
   vége(amíg)
   visszatérít d
Vége(algoritmus)
```

Mit fog visszatéríteni a `f(493, 1836)` hívás?

A. 2329 B. 2229 C. 2430 D. 3292

17. A `kiIr(M, n)` algoritmusnak bemeneti paramétere az ***n*** elemű ***M*** sorozat ($\boldsymbol{n} \leq 10$), amelynek elemei egész számok (***M***[1], ***M***[2], …, ***M***[***n***]). Az ***M*** sorozat egy halmazt ábrázol.

```
Algoritmus kiIr(M, n):
   szám ← 2^n
   k ← 0
   Amíg k < szám végezd el
      akt ← k
      Ki: '{'
      Minden j = 1, n végezd el
         r ← akt MOD 2
         akt ← akt DIV 2
         Ha r = 1 akkor
            Ki: M[j]
         vége(ha)
      vége(minden)
      Ki: '}'
      Ki: új sorba lépés
      k ← k + 1
   vége(amíg)
Vége(algoritmus)
```

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. Az algoritmus kiírja az ***M*** halmaz minden permutációját.
B. Az algoritmus kiírja az ***M*** halmaz minden ***i*** osztályú (***i*** = 0, 1, ..., ***n***) kombinációját (nem feltétlenül ebben a sorrendben).
C. Az algoritmus kiírja az ***M*** halmaz minden ***i*** osztályú (***i*** = 0, 1, ..., ***n***) variációját (nem feltétlenül ebben a sorrendben).
D. Az algoritmus kiírja az ***M*** halmaz minden részhalmazát.

18. Legyen az `s(a, b, c)` algoritmus, ahol ***a***, ***b***, ***c*** pozitív természetes számok (1 ≤ ***a***, ***b***, ***c*** ≤ 10000).

```
Algoritmus s(a, b, c):
   Ha (a = 1) VAGY (b = 1) VAGY (c = 1) akkor
      visszatérít 1
   különben
      Ha a > b akkor
         visszatérít a * s(a - 1, b, c)
      különben
         Ha a < b akkor
            visszatérít b * s(a, b - 1, c)
         különben
            visszatérít c * s(a - 1, b - 1, c - 1)
         vége(ha)
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

A következő állítások közül melyek igazak, ha ***a*** = ***b*** és ***a*** < ***c***?

A. Az algoritmus visszatéríti a ***c***! értékét
B. Az algoritmus visszatéríti a ***c***! / (***c*** – ***a*** + 1)! értékét
C. Az algoritmus visszatéríti a ***c***! / (***c*** – ***a*** – 1)! értékét
D. Az algoritmus visszatéríti ***c***-nek (***a*** – 1) osztályú variációit

19. Az alábbi algoritmusnak bemeneti paraméterei az ***n*** természetes számot tartalmazó ***A*** sorozat (***A***[1], ***A***[2], …, ***A***[***n***]) és az ***n*** természetes szám (1 ≤ ***n*** ≤ 10000). Az ***x*** és ***y*** természetes számok esetében az ***x***^***y*** jelölés jelentése: ***x*** az ***y***-dik hatványon (x^y).

```
Algoritmus h(A, n):
   Ha n = 0 akkor
      visszatérít 0
   különben
      visszatérít A[n] * (-1)^(1 - A[n] MOD 2) + h(A, n - 1)
   vége(ha)
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az algoritmus visszatéríti két összeg különbségét: az ***A*** sorozat páros pozícióin található elemeinek összegéből kivonja a páratlan pozíciókon található elemeknek az összegét.
B. Az algoritmus visszatéríti két összeg különbségét: az ***A*** sorozat páros értékű elemeinek összegéből kivonja a páratlan értékű elemeknek az összegét.
C. Az algoritmus visszatéríti két összeg különbségét: az ***A*** sorozat páratlan értékű elemeinek összegéből kivonja a páros értékű elemeknek az összegét.
D. A fenti válaszok közül egyik sem helyes.

20. Egy Excel állomány ***n*** bejegyzést tartalmaz, amelyek 1-től ***n***-ig meg vannak számozva. Ezeket a bejegyzéseket egy Word állományba kell másolnunk, amelyben a bejegyzések legtöbb ***r*** sorba és pontosan ***c*** oszlopba lesznek elhelyezve minden oldalon. Garantált, hogy ***n*** értéke mindig lehetővé teszi, hogy a bejegyzéseket pontosan ***c*** oszlopba helyezhessük.

Jelöljük x_1, ..., x_c-vel azoknak a bejegyzéseknek a számát, amelyeket egy oldal egyes oszlopaiba másolunk.
A Word dokumentum első oldalán, mivel itt fejléc is van, a sorok száma r_1, $r_1 < r$ (az első oldalon a sorok száma kisebb, mint a többi oldalon), vagyis $x_p = r_1$, $\forall\ 1 \le p \le c$.

A bejegyzéseket a Word állomány minden oldalán az oszlopokban fentről lefele helyezzük el, és az oszlopokat balról jobbra töltjük fel: ha valamelyik oldalon az első bejegyzés sorszáma i, az $(i + 1)$ sorszámú bejegyzés alatta következik, az $(i + x_1)$ sorszámú bejegyzés a második oszlop első bejegyzése lesz, és így tovább.

Szeretnénk, hogy a Word dokumentum utolsó oldalán az oszlopokba elhelyezett bejegyzések darabszámai legyenek minél közelebbiek, vagyis bármely két oszlop bejegyzéseinek darabszáma között a különbség legyen legtöbb 1 ($|x_j - x_k| \le 1$, $\forall\ 1 \le j, k \le c, j \ne k$).

A többi oldal esetében (kivéve az elsőt és az utolsót): $x_p = r$, $\forall\ 1 \le p \le c$.

Ha $n = 5883$, $r = 46$, $r_1 = 12$ és $c = 2$ a dokumentum adott oldalának melyik sorába kerülhet az utolsó bejegyzés (melynek sorszáma $i = 5883$)?

A. 29 B. 30 C. 31 D. 32

21. Legyen a `feldolgoz(a, b, c, d, e)` algoritmus, amelynek bemeneti paraméterei az a, b, c, d és e egész számok ($1 \le a, b \le 10000$, $2 \le c \le 16$, $1 \le d < c$.

```
Algoritmus feldolgoz(a, b, c, d, e):
   Ha a = 0 ÉS b = 0 akkor
      Ha e = 0 akkor
         visszatérít 1
      különben
         visszatérít 0
      vége(ha)
   vége(ha)
   Ha (a MOD c = d) ÉS (b MOD c = d) akkor
      visszatérít feldolgoz(a DIV c, b DIV c, c, d, e)
   vége(ha)
   Ha a MOD c = d akkor
      visszatérít feldolgoz(a DIV c, b DIV c, c, d, e + 1)
   vége(ha)
   Ha b MOD c = d akkor
      visszatérít feldolgoz(a DIV c, b DIV c, c, d, e - 1)
   különben
      visszatérít feldolgoz(a DIV c, b DIV c, c, d, e)
   vége(ha)
Vége(algoritmus)
```

A következő állítások közül melyek igazak a `feldolgoz(a, b, c, d, 0)` hívás esetében?

A. Ha az a és b számok, felírva a c számrendszerben, azonos darabszámú d-vel egyenlő számjegyet tartalmaznak, az algoritmus 1-et térít vissza, különben 0-át
B. Ha az a szám felírva a c számrendszerben, és a b szám felírva a c számrendszerben tartalmaznak (egyenként) legalább egy darab d-vel egyenlő számjegyet, az algoritmus 1-et térít vissza, különben 0-át
C. Ha vagy az a vagy a b szám, felírva a c számrendszerben, tartalmaz legalább egy d-vel egyenlő számjegyet, az algoritmus 1-et térít vissza, különben 0-át
D. Ha az a és b számok, felírva a c számrendszerben egyáltalán nem tartalmaznak d-vel egyenlő számjegyet, az algoritmus 1-et térít vissza, különben 0-át

22. Legyenek az `érték(p, s, i, n, x)` és a `kifejezésÉrték(p, n, x)` algoritmusok, amelyeknek paraméterei az $\boldsymbol{n}$ elemű, egész számokat tartalmazó $\boldsymbol{p}$ sorozat ($\boldsymbol{p}$[1], $\boldsymbol{p}$[2], …, $\boldsymbol{p}$[$\boldsymbol{n}$]), az $\boldsymbol{s}$, $\boldsymbol{i}$ és $\boldsymbol{n}$ természetes számok ($\boldsymbol{n} \leq 1000$, $\boldsymbol{n} = 2^{\boldsymbol{k}}$, $\boldsymbol{k} < 10$) és az $\boldsymbol{x}$ valós szám. A $\boldsymbol{p}$ sorozat elemei a $\boldsymbol{p}[1] + \boldsymbol{p}[2] \cdot \boldsymbol{x} + \boldsymbol{p}[3] \cdot \boldsymbol{x}^2 + \ldots + \boldsymbol{p}[\boldsymbol{n}] \cdot \boldsymbol{x}^{\boldsymbol{n}-1}$ kifejezés együtthatóit ábrázolják, a kitevők szerinti növekvő sorrendben, ahol a legnagyobb kitevő egyenlő ($\boldsymbol{n}$ – 1)-gyel.
Példa: $\boldsymbol{p}$ = [1, 2, 3, 4] az $\boldsymbol{E}(\boldsymbol{x}) = 1 + 2\boldsymbol{x} + 3\boldsymbol{x}^2 + 4\boldsymbol{x}^3$ kifejezésnek felel meg.

```
Algoritmus érték(p, s, i, n, x):
   Ha s + i ≤ n akkor
      _________________________
   különben
      visszatérít p[s]
   vége(ha)
Vége(algoritmus)

Algoritmus kifejezésÉrték(p, n, x):
   visszatérít érték(p, 1, 1, n, x)
Vége(algoritmus)
```

A következő változatok közül melyek egészítik ki helyesen az aláhúzott üres részt ahhoz, hogy a `kifejezésÉrték(p, n, x)` algoritmus az $\boldsymbol{E}(\boldsymbol{x})$ kifejezés értékét térítse vissza?

A. **visszatérít** `p[s] + x * érték(p, s + i, i * 2, n , x * x)`
B. **visszatérít** `érték(p, s, i * 2, n - i, x * x) + x * érték(p, s + i, i * 2, n, x * x)`
C. **visszatérít** `érték(p, s + i, i * 2, n, x * x) + x * érték(p, s, i * 2, n - i, x * x)`
D. **visszatérít** `p[s] + x * érték(p, s + i, i, n, x)`

23. Legyen az `f(a)` algoritmus, amelynek bemeneti paramétere az $\boldsymbol{a}$ természetes szám ($2 \leq \boldsymbol{a} < 1000000$), és amely *igaz* értéket térít vissza, ha létezik egy ***osztó*** természetes szám, 1 < ***osztó*** < $\boldsymbol{a}$, amely az $\boldsymbol{a}$ számnak osztója, különben *hamis* értéket. Az [$\boldsymbol{x}$] jelölés az $\boldsymbol{x}$ szám egész részét jelöli.

Melyek helyesek az `f(a)` algoritmus következő változatai közül?

A.
```
Algoritmus f(a):
    Ha a = 2 akkor
           visszatérít hamis
    vége(ha)
    Ha a MOD 2 = 0 akkor
           visszatérít igaz
    vége(ha)
    Minden osztó ← 3, [√a] - 1, 2 végezd el
           Ha a MOD osztó = 0 akkor
              visszatérít igaz
           vége(ha)
    vége(minden)
    visszatérít hamis
Vége(algoritmus)
```

B.
```
Algoritmus f(a):
    Minden osztó ← 2,[√a] végezd el
           Ha a MOD osztó = 0 akkor
              visszatérít igaz
           vége(ha)
    vége(minden)
    visszatérít hamis
Vége(algoritmus)
```

C.
```
Algoritmus f(a):
    Ha a ≤ 2 akkor
           visszatérít hamis
    vége(ha)
    Ha a MOD 2 = 0 akkor
           visszatérít igaz
    vége(ha)
    Minden osztó ← 3, [√a], 2 végezd el
           Ha a MOD osztó = 0 akkor
              visszatérít igaz
           vége(ha)
    vége(minden)
    visszatérít hamis
Vége(algoritmus)
```

D.
```
Algoritmus f(a):
    osztó ← a - 1
    Amíg igaz végezd el
           Ha a MOD osztó = 0 akkor
              visszatérít igaz
           vége(ha)
           osztó ← osztó - 1
    vége(amíg)
    visszatérít hamis
Vége(algoritmus)
```

24. Legyen az alábbi kifejezés, ahol $1 < \boldsymbol{A} < 2021$ és $1 < \boldsymbol{n} < 10202110$.

$$\boldsymbol{E}(\boldsymbol{A}, \boldsymbol{n}) = (\boldsymbol{A} + \boldsymbol{A}^2 + A^3 + ... + \boldsymbol{A}^n) \textbf{ MOD } 2021$$

Az alábbi algoritmusok közül melyek számolják ki helyesen az $\boldsymbol{E}(\boldsymbol{A}, \boldsymbol{n})$ értéket a megadott időbonyolultsággal?

Feltételezzük, hogy minden műveletet 32 biten ábrázolható adatokkal végzünk és, hogy $\boldsymbol{x}^{\boldsymbol{k}}$ értékét $O(\log \boldsymbol{k})$ időben számoljuk ki.

A.

```
Algoritmus E(A, n):
   visszatérít (A * (A^n - 1) DIV (A - 1)) MOD 2021
Vége(algoritmus)
```

Időbonyolultság: $O(\log \boldsymbol{n})$

B.

```
Algoritmus E(A, n):
   visszatérít ((A * (A^n - 1)) MOD 2021) DIV ((A - 1) MOD 2021)
Vége(algoritmus)
```

Időbonyolultság: $O(\log \boldsymbol{n})$

C.

```
Algoritmus E1(A, n):
   Ha n = 1 akkor
      visszatérít (A, A) // visszatérít egy értékpárt
   vége(ha)
   Ha n MOD 2 = 1 akkor
      (t1, t2) ← E1(A, n - 1)
      p ← (t1 * A) MOD 2021
      visszatérít (p, (p + t2) MOD 2021)
   különben
      (t1, t2) ← E1(A, n DIV 2)
      p ← (t1 * t1) MOD 2021
      visszatérít (p, ((1 + t1) * t2) MOD 2021)
   vége(ha)
Vége(algoritmus)

Algoritmus E(A, n):
   (aux1, aux2) ← E1(A, n)
   visszatérít aux2
Vége(algoritmus)
```

Időbonyolultság: $O(\log \boldsymbol{n})$

D.

```
Algoritmus E(A, n):
   válasz ← A
   Minden i = 2, n végezd el
      válasz ← válasz + A^i
   vége(minden)
   visszatérít válasz MOD 2021
Vége(algoritmus)
```

Időbonyolultság: $O(\boldsymbol{n} \cdot \log \boldsymbol{n})$

25. Egy körre felírunk minden számot 1-től 1000-ig növekvő sorrendben, az óramutató járásával megegyező irányban. Kezdve az 1-től, az óramutató járásával megegyező irányban kiszínezünk minden $\boldsymbol{k}$-dik számot ($1, \boldsymbol{k} + 1, 2 * \boldsymbol{k} + 1, ...$). Az eljárás folytatódik, amíg egy olyan számhoz érünk, amely már ki van színezve. Az eljárás végén $\boldsymbol{x}$ szám lesz kiszínezve.

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. Ha ***k*** = 15 akkor ***x*** = 300
B. Ha ***k*** = 45 akkor ***x*** = 200
C. Ha ***k*** = 25 akkor ***x*** = 40
D. Ha ***k*** = 30 akkor ***x*** = 150

26. Legyen a `mitCsinál(n, k)` algoritmus, ahol ***n*** és ***k*** természetes számok (1 ≤ ***n***, ***k*** ≤ 1000000).

```
Algoritmus mitCsinál(n, k):
   szám ← 0
   p ← 1
   Amíg (n ≠ 0) ÉS (k ≠ 0) végezd el
      Ha n MOD 2 ≠ 0 akkor
         szám ← szám + ((n DIV 10) MOD 10) * p
         p ← p * 10
      különben
         k ← k - 1
      vége(ha)
      n ← n DIV 10
   vége(amíg)
   visszatérít szám
Vége(algoritmus)
```

A következő híváspárok közül melyek térítenek vissza azonos értékeket?

A. `mitCsinál(32345, 3)` és `mitCsinál(321458, 7)`
B. `mitCsinál(321458, 4)` és `mitCsinál(2314587, 4)`
C. `mitCsinál(2314, 3)` és `mitCsinál(23145, 4)`
D. `mitCsinál(23145, 3)` és `mitCsinál(231458, 4)`

27. Legyenek a következő algoritmusok:

- `hatvány(b, p)` – meghatározza a b^p értéket (***b*** a ***p***-dik hatványon), ahol ***b*** és ***p*** természetes számok (1 ≤ ***b*** ≤ 20, 1 ≤ ***p*** ≤ 20);
- `számjegyekSzáma(szám)` – visszatéríti a nem nulla ***szám*** természetes szám számjegyeinek darabszámát (0 < ***szám*** ≤ 1000000), vagy 0-át, ha ***szám*** = 0;
- Az alábbi `szorzat(bal, jobb)` algoritmus, ahol ***bal***, ***jobb*** természetes számok (100 < ***bal*** < 1000000, 0 ≤ ***jobb*** < 1000000 és ***bal*** olyan szám, amelynek a 10-es számrendszerben legalább két nullától különböző számjegye van).

```
Algoritmus szorzat(bal, jobb):
   Ha bal > 0 akkor
      aktJobb ← ______________________________
      aktBal ← bal DIV 10
      Ha bal * jobb < aktBal * aktJobb akkor
         visszatérít szorzat(aktBal, aktJobb)
      különben
         visszatérít bal * jobb
      vége(ha)
   különben
      visszatérít bal * jobb
   vége(ha)
Vége(algoritmus)
```

A következő változatok közül melyik egészíti ki helyesen az aláhúzott részt ahhoz, hogy a `szorzat(bal, jobb)` algoritmus végrehajtása a következő hívások eredményeképpen

```
Ki: szorzat(1092, 0)
Ki: szorzat(75981, 0)
```

a 920-at és a 73575-öt térítse vissza?

A. `(bal MOD 10) * hatvány(10, számjegyekSzáma(jobb)) + jobb`
B. `(bal MOD 10) * hatvány(10, jobb) + jobb`
C. `(bal MOD 10) * hatvány(10, számjegyekSzáma(jobb))`
D. `(bal MOD 10) * számjegyekSzáma(jobb)`

28. Legyen a `mitCsinál(a, n, i, f)` algoritmus, amelynek bemeneti paraméterei az $\boldsymbol{n}$ elemű $\boldsymbol{a}$, egész számokat tároló sorozat ($\boldsymbol{a}[1]$, $\boldsymbol{a}[2]$, …, $\boldsymbol{a}[\boldsymbol{n}]$), az $\boldsymbol{i}$, $\boldsymbol{f}$ és $\boldsymbol{n}$ egész számok ($2 \le \boldsymbol{n} \le 10000$).

```
Algoritmus mitCsinál(a, n, i, f):
   Ha (i = n) ÉS (f = 2) akkor
      visszatérít igaz
   különben
      Ha (i = n) akkor
         visszatérít hamis
      különben
         Ha (f ≤ 1) ÉS (a[i] < a[i + 1]) akkor
            visszatérít mitCsinál(a, n, i + 1, 1)
         vége(ha)
         Ha (1 ≤ f) ÉS (a[i] > a[i + 1]) akkor
            visszatérít mitCsinál(a, n, i + 1, 2)
         vége(ha)
         visszatérít hamis
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

A következő állítások közül melyek igazak a `mitCsinál(a, n, 1, 0)` hívás esetében?

A. Az algoritmus akkor és csakis akkor térít vissza *igaz* értéket, ha az $\boldsymbol{a}$ sorozat maximális értéke egy olyan $\boldsymbol{i}$ pozíción található, amelyre $1 < \boldsymbol{i} < \boldsymbol{n}$.
B. Az algoritmus akkor és csakis akkor térít vissza *igaz* értéket, ha $\exists\, \boldsymbol{k}$ ($1 < \boldsymbol{k} < \boldsymbol{n}$), amelyre $\boldsymbol{a}[1] < \boldsymbol{a}[2] < \ldots < \boldsymbol{a}[\boldsymbol{k}] > \boldsymbol{a}[\boldsymbol{k} + 1] > \ldots > \boldsymbol{a}[\boldsymbol{n}]$.
C. Az algoritmus *hamis* értéket térít vissza, ha az $\boldsymbol{a}$ sorozat szigorúan növekvő.
D. Az algoritmus akkor és csakis akkor térít vissza *igaz* értéket, ha $\exists\, \boldsymbol{k}$ ($1 < \boldsymbol{k} < \boldsymbol{n}$), amelyre $\boldsymbol{a}[\boldsymbol{k}] > \boldsymbol{a}[\boldsymbol{k} + 1] > \ldots > \boldsymbol{a}[\boldsymbol{n}]$.

29. Legyen a következő algoritmus, amelynek paramétere az $\boldsymbol{n}$, nullától különböző természetes szám és amely egy természetes számot térít vissza.

```
Algoritmus f(n):
  j ← n
  Amíg j > 1 végezd el
    i ← 1
    Amíg i ≤ n^4 végezd el
      i ← 4 * i
    vége(amíg)
    j ← j DIV 2
  vége(amíg)
  visszatérít j
Vége(algoritmus)
```

A következő bonyolultsági osztályok közül melyikhez tartozik hozzá a fenti algoritmus időbonyolultsága?

A. $O(\log_2 \boldsymbol{n}^2)$
B. $O(\log_2^2 \boldsymbol{n}^2)$
C. $O(\log_4^2 \boldsymbol{n})$
D. $O(\log_2 \log_4 \boldsymbol{n})$

30. Adott egy $\boldsymbol{n}$ elemű $\boldsymbol{s}$ karakterlánc ($\boldsymbol{s}[1]$, $\boldsymbol{s}[2]$, …, $\boldsymbol{s}[\boldsymbol{n}]$), ahol az elemek az angol ábécé betűi. Meg szeretnénk határozni a karakterlánc leghosszabb utószeletét (szuffixét), amely palindrom. Egy karakterlánc utószelete olyan tömbszakasza az adott karakterláncnak, amely tartalmazza ennek utolsó karakterét. Például, az *abab* karakterlánc leghosszabb utószelete, amely palindrom: *bab*.

Feltételezzük, hogy létezik a következő algoritmus:

- `ascii(c)` – visszatéríti a ***c*** karakter ASCII kódját.

Feltételezzük, hogy az aritmetikai műveletek nem okoznak túlcsordulást az egész számok halmazán.

A következő algoritmusok közül melyik téríti vissza a kért utószelet hosszát, ha meghívjuk az `utószelet(s, n)` utasítással?

A.
```
Algoritmus utószelet(s, n):
    hf ← 0
    hb ← 0
    válasz ← 1
    Minden i ← n, 1, -1 végezd el
        hf ← ascii(s[i]) + 2021 * hf
        hb ← hb + ascii(s[i]) * 2021^(n - i)
        Ha hf = hb akkor
           válasz ← n – i + 1
        vége(ha)
    vége(minden)
    visszatérít válasz
Vége(algoritmus)
```

B.
```
Algoritmus utószelet(s, n):
    hf ← 0
    hb ← 0
    válasz ← 1
    Minden i ← n, 1, -1 végezd el
        hf ← ascii(s[i]) + 3 * hf
        hb ← hb + ascii(s[i]) * 3^(n - i)
        Ha hf = hb akkor
           válasz ← n – i + 1
        vége(ha)
    vége(minden)
    visszatérít válasz
Vége(algoritmus)
```

C.
```
Algoritmus utószelet(s, n):
    hf ← 0
    hb ← 0
    válasz ← 1
    Minden i ← n, 1, -1 végezd el
        hf ← ascii(s[i]) + 2021 * hb
        hb ← hf + ascii(s[i]) * 2021^(n - i)
        Ha hf = hb akkor
           válasz ← n – i + 1
        vége(ha)
    vége(minden)
    visszatérít válasz
Vége(algoritmus)
```

D. Egyik változat sem helyes.

BABEŞ-BOLYAI TUDOMÁNYEGYETEM
MATEMATIKA ÉS INFORMATIKA KAR

Felvételi verseny – 2021 szeptember 9.
Informatika írásbeli
JAVÍTÁSI KULCS & MEGOLDÁSOK

HIVATALBÓL: 10 pont

1	C	3 pont
2	B	3 pont
3	A	3 pont
4	B, D	3 pont
5	B	3 pont
6	D	3 pont
7	A, B, C	3 pont
8	B	3 pont
9	D	3 pont
10	C	3 pont
11	A, C	3 pont
12	A	3 pont
13	A, B	3 pont
14	A, C, D	3 pont
15	A, B	3 pont
16	A	3 pont
17	B, D	3 pont
18	B, D	3 pont
19	C	3 pont
20	C, D	3 pont
21	A	3 pont
22	B, D	3 pont
23	B, C	3 pont
24	C	3 pont
25	B, C	3 pont
26	A, D	3 pont
27	A	3 pont
28	B, C	3 pont
29	B, C	3 pont
30	A	3 pont