BABEŞ-BOLYAI TUDOMÁNYEGYETEM
MATEMATIKA ÉS INFORMATIKA KAR

Felvételi verseny – 2021 július 19.

Informatika írásbeli

1. Legyen a következő algoritmus, amelynek bemeneti paramétere az $\boldsymbol{n}$ természetes szám és egy természetes számot térít vissza.

```
Algoritmus számol(n):
   E ← 1
   P ← 1
   i ← 2
   Amíg i ≤ n végezd el
      P ← (-1) * P * i
      E ← E + P
      i ← i + 1
   vége(amíg)
   visszatérít E
Vége(algoritmus)
```

Mit fog visszatéríteni az algoritmus, ha tudjuk, hogy $\boldsymbol{n} \geq 1$?

A. $1! - 2! + 3! - 4! + ... + (-1)^{n+1} \cdot \boldsymbol{n}!$
B. $1 - 1! + 2! - 3! + ... + (-1)^{n} \cdot \boldsymbol{n}!$
C. $1 - 1 \cdot 2 + 1 \cdot 2 \cdot 3 - 1 \cdot 2 \cdot 3 \cdot 4 + ... + (-1)^{n+1} \cdot 1 \cdot 2 \cdot ... \cdot \boldsymbol{n}$
D. $1 + 1 \cdot 2 - 1 \cdot 2 \cdot 3 + 1 \cdot 2 \cdot 3 \cdot 4 + ... + (-1)^{n} \cdot 1 \cdot 2 \cdot ... \cdot \boldsymbol{n}$

2. Egy Excel állomány $\boldsymbol{n}$ bejegyzést tartalmaz, amelyek meg vannak számozva 1-től $\boldsymbol{n}$-ig. Ezeket a bejegyzéseket egy Word állományba kell másolni, amelyben a bejegyzések $\boldsymbol{r}$ sorba és $\boldsymbol{c}$ oszlopba lesznek elhelyezve minden oldalon (kivéve az elsőt és az utolsót). A Word dokumentum első oldalán, mivel itt fejléc is van, a sorok száma $\boldsymbol{r_1}$, $\boldsymbol{r_1} < \boldsymbol{r}$ (az első oldalon a sorok száma kisebb, mint a többi oldalon).
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 $\boldsymbol{i}$, az $(\boldsymbol{i} + 1)$ sorszámú bejegyzés alatta következik, az $(\boldsymbol{i} + \boldsymbol{r})$ sorszámú bejegyzés pedig első lesz az illető oldal második oszlopában, és így tovább.
Ha $\boldsymbol{n} = 5000$, $\boldsymbol{r} = 46$, $\boldsymbol{r_1} = 12$ és $\boldsymbol{c} = 2$, a Word dokumentum mely oldalán és mely oszlopában lesz az $\boldsymbol{i} = 3245$ sorszámú bejegyzés?

A. 36. oldal, utolsó oszlop
B. 37. oldal, első oszlop
C. 37. oldal, utolsó oszlop
D. 38. oldal, első oszlop

3. Legyen a következő `mitCsinál(m)` algoritmus, ahol $\boldsymbol{m}$ természetes szám ($10 \leq \boldsymbol{m} \leq 10000$).

```
Algoritmus mitCsinál(m):
   Ha m = 0 akkor
      visszatérít 0
   vége(ha)
   Ha m MOD 9 = 0 akkor
      visszatérít 9
   vége(ha)
   visszatérít m MOD 9
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az algoritmus visszatéríti az ***m*** szám 9-cel való osztási maradékát.
B. Az algoritmus visszatéríti az ***m*** szám azon osztóinak darabszámát, amelyek oszthatók 9-cel.
C. Az algoritmus visszatéríti az ***m*** szám *sorsszámjegyét* (egy szám sorsszámjegye egyenlő az adott szám számjegyeinek összegével, ha ez egy számjegyből áll; különben újból kiszámoljuk az összeg számjegyeinek összegét stb., ameddig a kapott összeg egy számjegyből nem fog állni).
D. Az algoritmus akkor és csakis akkor téríti vissza az ***m*** szám sorsszámjegyét, ha ***m*** osztható 9-cel (egy szám sorsszámjegye egyenlő az adott szám számjegyeinek összegével, ha ez egy számjegyből áll; különben újból kiszámoljuk az összeg számjegyeinek összegét stb., ameddig a kapott összeg egy számjegyből nem fog állni).

4. Ahhoz, hogy azokat az ***n*** számjegyű számokat generáljunk, melyek kizárólag a 0, 2, 9 számjegyeket tartalmazzák, felhasználunk egy algoritmust, amely ***n*** = 2-re a 20, 22, 29, 90, 92, 99 számokat generálja növekvő sorrendben.
Ha ugyanazt az algoritmust használjuk ***n*** = 4-re, melyik szám következik rögtön a 2009 után?

A. 2022
B. 2090
C. 2010
D. Egyik sem a fenti változatok közül.

5. Legyen a `keres(n)` algoritmus, ahol ***n*** természetes szám (0 ≤ ***n*** ≤ 1000000).

```
Algoritmus keres(n):
   v ← 0
   Ha n = 0 akkor
      visszatérít 1
   különben
      m ← n
      Amíg m > 0 végezd el
         Ha m MOD 10 = 0 akkor
            v ← v + 1
         vége(ha)
         m ← m DIV 10
      vége(amíg)
      visszatérít v
   vége(ha)
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az algoritmus meghatározza és visszatéríti az ***n*** számjegyeinek darabszámát.
B. Az algoritmus 1-et térít vissza, ha ***n*** 10-nek hatványa, különben 0-át.
C. Az algoritmus 1-et térít vissza, ha ***n*** utolsó számjegye 0, különben 0-át.
D. Az algoritmus meghatározza és visszatéríti az ***n*** számban található 0 értékű számjegyek darabszámát.

6. Legyen az `abc(a, n, p)` algoritmus, ahol ***n*** természetes szám (1 ≤ ***n*** ≤ 10000), ***p*** egész szám (-10000 ≤ ***p*** ≤ 10000) és ***a*** egy nullától különböző természetes számokból álló ***n*** elemű sorozat (***a***[1], ***a***[2], …, ***a***[***n***]).

```
Algoritmus abc(a, n, p):
   Ha n < 1 akkor
      visszatérít 0
   különben
      Ha (1 ≤ p) ÉS (p ≤ n) akkor
         visszatérít a[p]
      különben
         visszatérít -1
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az algoritmus akkor és csakis akkor térít vissza -1-et, ha ***p*** negatív vagy nagyobb, mint ***n***.
B. Az algoritmus visszatéríti a ***p*** pozíción található elem értékét, ha ***p*** nagyobb, mint 0 és kisebb vagy egyenlő a sorozat hosszával.
C. Az algoritmus soha nem térít vissza 0-át, ha a paraméterek megfelelnek a kijelentésben megadott előfeltételeknek.
D. Az algoritmus visszatéríti a ***p*** pozíción található elem értékét, ha ***p*** nagyobb vagy egyenlő, mint 0 és kisebb, mint a sorozat hossza. Ha ***p*** értéke nem 1 és ***n*** közötti, -1-et térít vissza.

7. A következő algoritmusrészletek közül melyik határozza meg az ***i*** változóban egy olyan karakterláncnak a hosszát, amelyik a '*' (csillag) karakterrel ér véget? Az első karakter indexe 1, és a csillag karakter hozzátartozik a karakterlánchoz.

A.
```
i ← 1
Amíg x[i] ≠ '*' végezd el
   i ← i + 1
vége(amíg)
```
B.
```
i ← 1
Amíg x[i] = '*' végezd el
   i ← i + 1
vége(amíg)
i ← i - 1
```
C.
```
i ← 1
Amíg x[i] ≠ '*' végezd el
   i ← i + 1
vége(amíg)
i ← i + 1
```
D.
```
i ← 1
Amíg x[i] ≠ '*' végezd el
   i ← i + 1
vége(amíg)
i ← i - 1
```

8. Legyen a következő algoritmus, amelynek paramétere az ***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 végezd el
         i ← 2 * i
      vége(amíg)
      j ← j DIV 3
   vége(amíg)
   visszatérít j
Vége(algoritmus)
```

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

A. $O(\log_2 \boldsymbol{n})$
B. $O(\log_2^2 \boldsymbol{n})$
C. $O(\log_3^2 \boldsymbol{n})$
D. $O(\log_2 \log_3 \boldsymbol{n})$

9. A `hány(n, m)` algoritmus bemeneti paraméterei az ***n*** és ***m*** természetes számok.

```
Algoritmus hány(n, m):
   Ha n ≤ m akkor
      Ha (n MOD 2 = 0) ÉS (n MOD 3 ≠ 0) akkor
         visszatérít 1 + hány(n + 1, m)
      különben
         visszatérít hány(n + 1, m)
      vége(ha)
   különben
      visszatérít 0
   vége(ha)
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Ha ***n*** = 0 és ***m*** = 1, az algoritmus 0-át térít vissza.
B. Ha ***n*** = 4 és ***m*** = 21, az algoritmus 6-ot térít vissza.
C. Ha ***n*** = 7 és ***m*** = 120, az algoritmus 36-ot térít vissza.
D. Ha ***n*** = 1 és ***m*** = 215, az algoritmus 72-t térít vissza.

10. Legyen az `ellenőríz(n)` algoritmus, ahol ***n*** természetes szám (1 ≤ ***n*** ≤ 100000).

```
Algoritmus ellenőríz(n):
   Amíg n > 0 végezd el
      Ha (n MOD 3) > 1 akkor
         visszatérít 0
      vége(ha)
      n ← n DIV 3
   vége(amíg)
   visszatérít 1
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az algoritmus 1-et térít vissza, ha ***n*** a 3-nak hatványa, különben 0-át.
B. Az algoritmus 1-et térít vissza, ha ***n***-et felírva 3-as számrendszerben, csak 0 és/vagy 1-es számjegyeket tartalmaz, különben 0-át.
C. Az algoritmus 1-et térít vissza, ha ***n*** felírható a 3-as szám különböző hatványainak összegeként, különben 0-át.
D. Az algoritmus 1-et térít vissza, ha ***n***-et felírva 3-as számrendszerben, csak 2-es számjegyeket tartalmaz, különben 0-át.

11. Adott ***sz*** természetes számra (1000 ≤ ***sz*** ≤ 1000000) a következőképpen definiáljuk a csökkentési műveletet: ha ***sz*** utolsó számjegye nem 0, levonunk ***sz***-ből 1-et, különben elosztjuk ***sz***-et 10-zel és csak az egész részt tartjuk meg. A következő algoritmusok közül melyik téríti vissza, a `csökkent(sz, k)` hívás esetén azt a számot, amelyet úgy kapunk, hogy a csökkentési műveletet ***k***-szor (0 ≤ ***k*** ≤ 100) alkalmazzuk ***sz***-re? Például, ha ***sz*** = 15243 és ***k*** = 10, az eredménynek 151-nek kell lennie.

A.

```
Algoritmus csökkent(sz, k):
   Ha k = 0 akkor
      visszatérít sz
   különben
      Ha sz MOD 10 ≠ 0 akkor
         visszatérít csökkent(sz DIV 10, k - 1)
      különben
         visszatérít csökkent(sz - 1, k - 1)
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

B.

```
Algoritmus csökkent(sz, k):
   Amíg k > 0 végezd el
      Ha sz MOD 10 = 0 akkor
         sz ← sz DIV 10
      különben
         sz ← sz - 1
      vége(ha)
   vége(amíg)
   visszatérít sz
Vége(algoritmus)
```

C.

```
Algoritmus csökkent(sz, k):
   Minden i ← 1, k végezd el
      Ha sz MOD 10 > 0 akkor
         sz ← sz - 1
      különben
         sz ← sz DIV 10
      vége(ha)
   vége(minden)
   visszatérít sz
Vége(algoritmus)
```

D.

```
Algoritmus csökkent(sz, k):
   Ha k = 0 akkor
      visszatérít sz
   különben
      Ha k > sz MOD 10 akkor
         sz1 ← sz DIV 10
         visszatérít csökkent(sz1,  k - sz MOD 10 - 1)
      különben
         visszatérít csökkent(sz - k, 0)
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

12. Adott a következő algoritmus, amelynek bemeneti paraméterei az ***n*** természetes számot tartalmazó ***x*** sorozat (***x***[1], ***x***[2], …, ***x***[***n***]) és az ***n*** egész szám.

```
Algoritmus f(x, n):
   Ha n = 1 akkor
      visszatérít 100
   különben
      Ha x[n] > f(x, n - 1) akkor
         visszatérít x[n]
      különben
         visszatérít f(x, n - 1)
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

Mi lesz az algoritmus végrehajtásának eredménye, ha ***x*** = [101, 7, 6, 3] és ***n*** = 4?

A. 101
B. 3
C. 100
D. 7

13. Az alábbi algoritmus bemeneti paraméterei az ***n*** természetes számot tartalmazó ***a*** sorozat (***a***[1], ***a***[2], …, ***a***[***n***]) és az ***n*** természetes szám ($2 \leq n \leq 10000$).

```
Algoritmus h(a, n):
   Ha n ≤ 0 akkor
      visszatérít 0
   vége(ha)
   Ha (n MOD 2 = 0) ÉS (a[n] MOD 2 = 0) akkor
      visszatérít h(a, n - 1) + a[n]
   vége(ha)
   visszatérít h(a, n - 1) - a[n]
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az algoritmus visszatéríti két összeg különbségét: az ***a*** sorozat azon elemeinek összegéből, amelyeknek párossága (paritása) azonos a pozíciójuk párosságával az algoritmus kivonja azoknak az elemeknek az összegét, amelyeknek párossága különbözik a pozíciójuk párosságától.
B. Az algoritmus visszatéríti két összeg különbségét: az ***a*** sorozat azon páros szám értékű elemeinek összegéből, amelyek páros szám értékű pozíción találhatók az algoritmus kivonja azon páratlan szám értékű elemeinek az összegét, amelyek páratlan szám értékű pozíción találhatók.
C. Az algoritmus visszatéríti két összeg különbségét: az ***a*** sorozat páros szám értékű elemeinek összegéből kivonja a páratlan szám értékű elemeinek összegét.
D. Az algoritmus visszatéríti két összeg különbségét: az ***a*** sorozat páros szám értékű elemeinek összegéből, amelyek páros szám értékű pozíción találhatók kivonja a fennmaradó elemek összegét.

14. Legyen a `mitCsinál(n)` algoritmus, amelynek bemeneti paramétere az ***n*** nullától különböző természetes szám.

```
Algoritmus mitCsinál(n):
   i ← 1
   Amíg n > 0 végezd el
      Ha n MOD 2 ≠ 0 akkor
         Ki: i
      vége(ha)
      i ← i + 1
      n ← n DIV 2
   vége(amíg)
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Ha ***n*** = 31, az algoritmus a következő sort írja ki: 12345.
B. Ha ***n*** = 14, az algoritmus a következő sort írja ki: 234.
C. Ha ***n*** páratlan szám, az algoritmus 1-et ír ki a számjegysorozat elején.
D. Ha $n = 2^k$, (***k*** természetes szám), az algoritmus egyetlen számot ír ki.

15. Adott az ***S*** halmaz, amely ***n*** intervallumot tartalmaz. Egy intervallumot a b_i bal végpontjának és a j_i jobb végpontjának segítségével adunk meg ($b_i < j_i \ \forall\ i = 1, ..., n$). Azt az ***m*** elemű $S' \subseteq S$ részhalmazt szeretnénk meghatározni, amelyre érvényes, hogy ***S'***-ben nincs két olyan intervallum, amelyek metszik egymást (vagyis üres a keresztmetszetük) és ***m***-nek az értéke a lehető legnagyobb.
A következő stratégiák közül melyik oldja meg helyesen a feladatot?

A. Növekvően rendezzük az ***S*** halmazhoz tartozó intervallumokat a bal végpontjuk szerint. A rendezett sorozatból az első intervallumot betesszük ***S’***-be. Bejárjuk a rendezett sorozat többi elemét, és amikor találunk egy olyan intervallumot, amely nem metszi azt az intervallumot, amelyet legutoljára helyeztünk ***S’***-be, akkor ezt is hozzáadjuk ***S’***-hez.
B. Növekvően rendezzük az ***S*** halmazhoz tartozó intervallumokat a jobb végpontjuk szerint. A rendezett sorozatból az első intervallumot betesszük ***S’***-be. Bejárjuk a rendezett sorozat többi elemét, és amikor találunk egy olyan intervallumot, amely nem metszi azt az intervallumot, amelyet legutoljára helyeztünk ***S’***-be, akkor ezt is hozzáadjuk ***S’***-hez.
C. Növekvően rendezzük az ***S*** halmazhoz tartozó intervallumokat a hosszúságuk szerint. A rendezett sorozatból az első intervallumot betesszük ***S’***-be. Bejárjuk a rendezett sorozat többi elemét, és amikor találunk egy olyan intervallumot, amely nem metszi azt az intervallumot, amelyet legutoljára helyeztünk ***S’***-be, akkor ezt is hozzáadjuk ***S’***-hez.
D. Növekvően rendezzük az ***S*** halmazhoz tartozó intervallumokat aszerint, hogy hány intervallumot metszenek ***S***-ből. A rendezett sorozatból az első intervallumot betesszük ***S’***-be. Bejárjuk a rendezett sorozat többi elemét, és amikor találunk egy olyan intervallumot, amely nem metszi azt az intervallumot, amelyet legutoljára helyeztünk ***S’***-be, akkor ezt is hozzáadjuk ***S’***-hez.

16. Legyen az `f(a, b)` algoritmus, amelynek bemeneti paraméterei az ***a*** és ***b*** természetes számok ($1 \leq \boldsymbol{a} < \boldsymbol{b} \leq 1000$).

```
Algoritmus f(a, b):
   m ← 0
   Minden n ← a, b végezd el
      c ← 0
      Minden d ← 1, n végezd el
         Ha n MOD d = 0 akkor
            c ← c + 1
         vége(ha)
      vége(minden)
      Ha c > m akkor
         m ← c
      vége(ha)
   vége(minden)
   Minden n ← a, b végezd el
      c ← 0
      Minden d ← 1, n végezd el
         Ha n MOD d = 0 akkor
            c ← c + 1
         vége(ha)
      vége(minden)
      Ha c = m akkor
         Kiír n
      vége(ha)
   vége(minden)
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az algoritmus kiírja az ***a*** osztóinak darabszáma és a ***b*** osztóinak darabszáma közül a nagyobbat.
B. Az algoritmus kiírja azokat az [***a***, ***b***] intervallumhoz tartozó természetes számokat, amelyeknek az a tulajdonságuk, hogy osztóiknak darabszáma maximális.
C. Az algoritmus kiírja az [***a***, ***b***] intervallumhoz tartozó minden egyes természetes szám osztóinak darabszámát.
D. Az algoritmus kiírja azokat az [***a***, ***b***] intervallumhoz tartozó természetes számokat, amelyeknek az a tulajdonságuk, hogy valódi osztóiknak darabszáma maximális.

17. Legyen ***a*** és ***b*** két természetes szám, ahol $\boldsymbol{b} \neq 0$. A következő változatok közül melyik számolja ki:

- `a DIV b`, ha `a MOD b = 0`
- `(a / b)` felfele kerekítve a legközelebbi egész számra, ha `a MOD b ≠ 0`

A. `(a - 1) DIV b`
B. `(a + b + 1) DIV b`
C. `(a + b - 1) DIV b`
D. `((a + 2 * b - 1) DIV b) - 1`

18. Jancsinak implementálnia kell a bináris keresés algoritmusát, amellyel meg kell keresnie egy $\boldsymbol{a}$ elemet az $\boldsymbol{n}$ elemű, egész számokat tároló, növekvően rendezett $\boldsymbol{V}$ sorozatban ($1 \le \boldsymbol{n} \le 1000$) ($\boldsymbol{V}[1]$, $\boldsymbol{V}[2]$, …, $\boldsymbol{V}[\boldsymbol{n}]$). Jancsi a következő algoritmust írta:

```
Algoritmus binárisKeresés(a, n, V):
   bal ← 1
   jobb ← n
   Amíg jobb – bal > 1 végezd el
      közép ← (bal + jobb) DIV 2
      Ha a ≤ V[közép] akkor
         jobb ← közép
      különben
         bal ← közép
      vége(ha)
   vége(amíg)
   visszatérít jobb
Vége(algoritmus)
```

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. Ha $\boldsymbol{n} = 1$, akkor az algoritmus mindig 1-et térít vissza.
B. Bármely $\boldsymbol{n} \ge 1$-re, ha $\boldsymbol{a}$ kisebb a sorozat minden eleménél, a Jancsi algoritmusa 1-et térít vissza.
C. Ha az $\boldsymbol{a}$ érték megtalálható a sorozatban, a Jancsi algoritmusa NEM mindig téríti vissza az $\boldsymbol{a}$ pozícióját (indexét a $\boldsymbol{V}$ vektorban).
D. Ha $\boldsymbol{n} > 1$ és $\boldsymbol{a}$ nagyobb a sorozat minden eleménél, a Jancsi algoritmusa az $\boldsymbol{n}$ értékét téríti vissza.

19. Legyen a `számol(x, n)` algoritmus, amelynek bemeneti paraméterei az $\boldsymbol{n}$ és $\boldsymbol{x}$ természetes számok, ahol $1 \le \boldsymbol{x} \le \boldsymbol{n} < 10$.

```
Algoritmus számol(x, n):
   b ← 1
   Minden i ← 1, n - x végezd el
      b ← b * i
   vége(minden)
   a ← b
   Minden i ← n – x + 1, n végezd el
      a ← a * i
   vége(minden)
   visszatérít a DIV b
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Ha $\boldsymbol{n} = 5$ és $\boldsymbol{x} = 2$, akkor az algoritmus 20-at térít vissza.
B. Ha $\boldsymbol{n} = 3$ és $\boldsymbol{x} = 2$, akkor az algoritmus 6-ot térít vissza.
C. Az algoritmus a $\{\overline{c_1 c_2 \ldots c_x}\ : \boldsymbol{c_i} \neq \boldsymbol{c_j}\ \forall\ 1 \le \boldsymbol{i}, \boldsymbol{j} \le \boldsymbol{x}, \boldsymbol{i} \neq \boldsymbol{j},\ 1 \le \boldsymbol{c_i} \le \boldsymbol{n}\}$ halmaz számosságát téríti vissza.
D. Az algoritmus $\boldsymbol{n}$ szorzási műveletet hajt végre.

20. Legyen a `mitCsinál(n, k)` algoritmus, amelynek bemeneti paraméterei az $\boldsymbol{n}$ és $\boldsymbol{k}$ ($1 \le \boldsymbol{n}, \boldsymbol{k} \le 1000000$) nullától különböző természetes számok:

```
Algoritmus mitCsinál(n, k):
```

```
   Amíg n ≥ 1 végezd el
      Ha k ≤ n akkor
         i ← k
      különben
         i ← n
      vége(ha)
      n ← n - i
      x ← 1
      Amíg i ≥ 1 végezd el
         Ki: x, ' '
         x ← x + 1
         i ← i - 1
      vége(amíg)
   vége(amíg)
Vége(algoritmus)
```

Döntsétek el, hogy a következő állítások közül melyek igazak:

A. Ha n = 8 és k = 3, az algoritmus a következő sorozatot írja ki: 1 2 3 1 2 3 1 2
B. Ha k = 2, n legkisebb értéke, amelyre az algoritmus az 1-est háromszor írja ki: n = 3.
C. Ha k = 5, n legkisebb értéke, amelyre az algoritmus a 2-est 37-szer írja ki: n = 182.
D. Ha n = 7 és k = 3, az algoritmus a következő sorozatot írja ki: 1 2 3 1 2 3

21. Legyen a `számol(a, b, c)` algoritmus, amelynek bemeneti paraméterei nullától különböző természetes számok. Az algoritmus kiszámolja a három szám legnagyobb közös osztóját.
Melyik helyes a következő implementálások közül:

A.

```
Algoritmus számol(a, b, c):
   Amíg (a ≠ b) VAGY (a ≠ c) VAGY (b ≠ c) végezd el
      x ← a
      Ha a ≠ x akkor
         a ← a - x
      vége(ha)
      Ha b ≠ x akkor
         b ← b - x
      vége(ha)
      Ha c ≠ x akkor
         c ← c - x
      vége(ha)
   vége(amíg)
   visszatérít x
Vége(algoritmus)
```

B.

```
Algoritmus számol(a, b, c):
   x ← a
   y ← b
   Amíg x ≠ y végezd el
      Ha x > y akkor
         x ← x - y
      különben
         y ← y - x
      vége(ha)
   vége(amíg)
   z ← c
   Amíg x ≠ z végezd el
      Ha x > z akkor
         x ← x - z
      különben
         z ← z - x
      vége(ha)
   vége(amíg)
   visszatérít x
Vége(algoritmus)
```

C.

```
Algoritmus számol(a, b, c):
   Amíg (a ≠ b) VAGY (a ≠ c) VAGY (b ≠ c) végezd el
      x ← a
      Ha b < x akkor
         x ← b
      vége(ha)
      Ha c < x akkor
         x ← c
      vége(ha)
      Ha a ≠ x akkor
         a ← a - x
      vége(ha)
      Ha b ≠ x akkor
         b ← b - x
      vége(ha)
      Ha c ≠ x akkor
         c ← c - x
      vége(ha)
   vége(amíg)
   visszatérít x
Vége(algoritmus)
```

D.

```
Algoritmus számol(a, b, c):
   x ← a
   y ← b
   r ← x MOD y
   Amíg r ≠ 0 végezd el
      x ← y
      y ← r
      r ← x MOD y
   vége(amíg)
   z ← c
   r ← y MOD z
   Amíg r ≠ 0 végezd el
      y ← z
      z ← r
      r ← y MOD z
   vége(amíg)
   visszatérít z
Vége(algoritmus)
```

22. A `mitCsinál(n)` algoritmus bemeneti paramétere az ***n*** természetes szám ($1 \leq n \leq 100$).

```
Algoritmus mitCsinál(n):
   s ← 0
   Ha n MOD 2 = 0 akkor
      a ← 1
      Amíg a < n végezd el
         s ← s + a
         a ← a + 2
      vége(amíg)
   különben
      b ← 2
      Amíg b < n végezd el
         s ← s + b
         b ← b + 2
      vége(amíg)
   vége(ha)
   visszatérít s
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Ha ***n*** páros szám, az algoritmus visszatéríti az ***n***-nél szigorúan kisebb természetes számok összegét; ha ***n*** páratlan szám, az algoritmus visszatéríti az ***n***-nél kisebb természetes páros számok összegét.
B. Ha ***n*** páros szám, az algoritmus visszatéríti az ***n***-nél szigorúan kisebb természetes páros számok összegét; ha ***n*** páratlan szám, az algoritmus visszatéríti az ***n***-nél kisebb természetes páratlan számok összegét.
C. Ha ***n*** páros szám, az algoritmus visszatéríti az ***n***-nél kisebb természetes páratlan számok összegét; ha ***n*** páratlan szám, az algoritmus visszatéríti az ***n***-nél kisebb természetes páros számok összegét.
D. Ha ***n*** páros szám, az algoritmus visszatéríti az ***n***-nél szigorúan kisebb természetes páros számok összegét; ha ***n*** páratlan szám, az algoritmus visszatéríti az ***n***-nél szigorúan kisebb természetes számok összegét.

23. A `mitCsinál(n)` algoritmus bemeneti paramétere az ***a*** természetes szám ($1 \leq a \leq 100000$).

```
Algoritmus mitCsinál(a):
   b ← 0
   c ← 0
   d ← 0
   e ← 1
   Amíg a > 0 végezd el
      d ← a MOD 10
      Ha (d ≠ 4) ÉS (d < 7) akkor
         b ← b + e * (d DIV 2)
         c ← c + e * (d - d DIV 2)
      különben
         b ← b + e
         c ← c + e * (d - 1)
      vége(ha)
      a ← a DIV 10
      e ← e * 10
   vége(amíg)
   Kiír b
   Kiír c
Vége(algoritmus)
```

A következő értékpárok közül melyiket nem fogja kiírni az algoritmus egyetlen megengedett bemeneti paraméterérték esetében sem?

A. 1112 és 11233
B. 1111 és 88888
C. 21001 és 33011
D. 3141 és 3258

24. Legyenek az `f(n, c)` és a `g(n, c)` algoritmusok, amelyeknek bemeneti paraméterei az ***n*** és a ***c*** természetes számok.

```
Algoritmus f(n, c):
   Ha n ≤ 9 akkor
      Ha n = c akkor
         visszatérít 1
      különben
         visszatérít 0
      vége(ha)
   különben
      Ha n MOD 10 = c akkor
         visszatérít f(n DIV 10, c) + 1
      különben
         visszatérít f(n DIV 10, c)
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

```
Algoritmus g(n, c):
   Ha c = 0 akkor
      visszatérít 0
   különben
      Ha f(n, c) > 0 akkor
         visszatérít g(n, c - 1) + 1
      különben
         visszatérít g(n, c - 1)
      vége(ha)
   vége(ha)
Vége(algoritmus)
```

Mit térít vissza a `g(n, 9)` hívás, ahol ***n*** természetes szám?

A. Az algoritmus visszatéríti az ***n*** szám számjegyeinek darabszámát.
B. Az algoritmus visszatéríti az ***n*** szám különböző számjegyeinek darabszámát.
C. Az algoritmus visszatéríti az ***n*** szám 1-nél nagyobb számjegyeinek darabszámát.
D. A fenti válaszok közül egyik sem helyes.

25. Egy bizonyos honlapon, jelszó helyett, minden nyilvántartásba vett felhasználó egy ***n*** számjegyből álló titkos kóddal rendelkezik. Ahhoz, hogy a felhasználó beléphessen a honlapra, nem kell megadnia a teljes kódot, hanem a honlap véletlenszerűen generál 3 különböző pozíciót, ***p***1-et, ***p***2-t és ***p***3-at úgy, hogy 1 ≤ ***p***1 < ***p***2 < ***p***3 ≤ ***n*** és a felhasználónak csak azokat a számjegyeket kell megadnia, amelyek ezen a három pozíción találhatók. Például, ha a felhasználó kódja 987654321 és a honlap által véletlenszerűen generált pozíciók a 2, 5 és 7, a felhasználónak a 8, 5, 3 számjegyeket kell megadnia. Alább adva vannak azok az értékek, amelyeket egy felhasználó 9 bejelentkezés alkalmával adott meg.
1, 2, 3
2, 9, 0
6, 3, 2
2, 0, 2
1, 4, 7
9, 3, 2
4, 4, 3
4, 3, 1
5, 6, 0

Feltételezve, hogy mind a 9 bejelentkezés helyes és a felhasználó kódja nem változott időközben, állapítsátok meg, hogy a következő állítások közül melyek helyesek.

A. A felhasználó kódja biztosan nem tartalmazza a 8-as számjegyet.
B. A legrövidebb lehetséges kódnak 12 számjegye van.
C. A legrövidebb lehetséges kód a 2-es számjegyet legkevesebb 3-szor tartalmazza.
D. A legrövidebb lehetséges kód számjegyeinek összege lehet 44.

26. Legyen az `f(x, n)` algoritmus, ahol ***x*** és ***n*** természetes számok és ***x*** > 0.

```
Algoritmus f(x, n):
   Ha n = 0 akkor
      visszatérít 1
   vége(ha)
   m ← n DIV 2
   p ← f(x, m)
   Ha n MOD 2 = 0 akkor
      visszatérít p * p
   vége(ha)
   visszatérít x * p * p
Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az algoritmus visszatéríti az x^n értéket.
B. Ha „`n MOD 2`” helyett „`m MOD 2`” lenne, akkor az algoritmus az x^n értéket térítené vissza.
C. Mivel az algoritmus meghívja önmagát, az algoritmus önmeghívása utáni sorok soha nem kerülnek végrehajtásra.
D. Az algoritmus akkor és csakis akkor téríti vissza az x^n értéket, ha n páros szám.

27. Legyen az `f2(a, b)` algoritmus, amelynek bemeneti paraméterei az ***a*** és ***b*** természetes számok, és az `f(arr, i, n, p)` algoritmus, amelynek bemeneti paraméterei az ***n*** egész számot tartalmazó ***arr*** sorozat (***arr***[1], ***arr***[2], …, ***arr***[***n***]) és az ***i*** és ***p*** egész számok.

```
Algoritmus f2(a, b):
   Ha a > b akkor
      visszatérít a
   különben
      visszatérít b
   vége(ha)
Vége(algoritmus)

Algoritmus f(arr, i, n, p):
   Ha i = n akkor
      visszatérít 0
   vége(ha)
   n1 ← f(arr, i + 1, n, p)
   n2 ← 0
   Ha p + 1 ≠ i akkor
      n2 ← f(arr, i + 1, n, i) + arr[i]
   vége(ha)
   visszatérít f2(n1, n2)
Vége(algoritmus)
```

Adjátok meg az `f(arr, 1, 9, -10)` hívás eredményét, ha az ***arr*** sorozat a következő elemeket tartalmazza: (10, 1, 3, 4, 8, 12, 1, 11, 6).

A. 24
B. 37
C. 26
D. 56

28. Legyen az `ellenőríz(n)` algoritmus, amelynek bemeneti paramétere az ***n*** egész szám ($1 \leq n \leq 100000$) és amely *igaz* értéket térít vissza, ha az ***n*** számnak van egy olyan számjegye, amely egyenlő a többi számjegy összegével. Például, az `ellenőríz(1517)` hívás *igaz*-at térít vissza, mivel 7 = 1 + 5 + 1. A következő algoritmusok közül melyik implementálja helyesen az `ellenőríz(n)` algoritmust?

A.

```
Algoritmus ellenőríz(n):
   s ← 0
   c ← n
   r ← hamis
   Amíg c > 0 végezd el
      s ← s + c MOD 10
      c ← c DIV 10
   vége(amíg)
   c ← n
   Amíg c > 0 végezd el
      d ← c MOD 10
      Ha d = s - d akkor
         r ← igaz
      különben
         r ← hamis
      vége(ha)
      c ← c DIV 10
```

```
   vége(amíg)
   visszatérít r
Vége(algoritmus)
```

B.

```
Algoritmus ellenőríz(n):
   m ← -1
   c ← n
   r ← hamis
   Amíg c > 0 végezd el
      d ← c MOD 10
      c ← c DIV 10
      Ha d > m akkor
         m ← d
      vége(ha)
   vége(amíg)
   c ← n
   s ← 0
   Amíg c > 0 végezd el
      d ← c MOD 10
      Ha d ≠ m akkor
         s ← s + d
      vége(ha)
      c ← c DIV 10
   vége(amíg)
   Ha s = m akkor
      r ← igaz
   vége(ha)
   visszatérít r
Vége(algoritmus)
```

C.

```
Algoritmus ellenőríz(n):
   v ← [0, 0, 0, 0, 0, 0, 0, 0, 0]
   r ← hamis
   Amíg n > 0 végezd el
      d ← n MOD 10
      Ha d > 0 akkor
         v[d] ← v[d] + 1
      vége(ha)
      n ← n DIV 10
   vége(amíg)
   m ← 9
   Amíg v[m] = 0 végezd el
      m ← m - 1
   vége(amíg)
   Ha v[m] = 1 akkor
      d ← m
      s ← 0
      m ← m - 1
      Amíg m > 0 végezd el
         s ← s + v[m] * m
         m ← m - 1
      vége(amíg)
      Ha d = s akkor
         r ← igaz
      vége(ha)
   vége(ha)
   visszatérít r
Vége(algoritmus)
```

D. A fenti változatok közül egyik sem helyes.

29. Legyen az `f(x, n, e, y, m)` algoritmus, amelynek bemeneti paraméterei az ***n*** egész számot tartalmazó ***x*** sorozat (***x***[1], ***x***[2], …, ***x***[***n***]), az ***m*** egész számot tartalmazó ***y*** sorozat (***y***[1], ***y***[2], …, ***y***[***m***]) és az ***e*** egész szám, amely nem tartozik az ***y*** sorozathoz. Az algoritmus egy sorozatot és egy természetes számot térít vissza. Adottak a következő algoritmusok:

- `(c, p) ← ragasztás(a, n, b, m)`, amelynek bemeneti paraméterei az ***n*** egész számot tartalmazó ***a*** sorozat és az ***m*** egész számot tartalmazó ***b*** sorozat; az algoritmus a ***p*** egész számot tartalmazó ***c*** sorozatot téríti vissza, amely az ***a*** és ***b*** sorozatok összeragasztásának eredménye, vagyis: ***a***[1], ***a***[2], …, ***a***[***n***], ***b***[1], ***b***[2], …, ***b***[***m***].
- `(c, p) ← különbség(a, n, b, m)`, amelynek bemeneti paraméterei az ***n*** egész számot tartalmazó ***a*** sorozat és az ***m*** egész számot tartalmazó ***b*** sorozat; az algoritmus a ***p*** egész számot tartalmazó ***c*** sorozatot téríti vissza, amely az ***a*** sorozat azon elemeit tartalmazza, amelyek nem találhatók meg a ***b*** sorozatban (a megmaradó elemek megőrzik eredeti sorrendjüket).

```
1.  Algoritmus f(x, n, e, y, m):
2.      Ha n = 0 akkor
3.          visszatérít [], 0
4.      vége(ha)
5.      Ha x[1] ≠ e akkor
6.          s ← []
7.          s[1] ← x[1]
8.          (r1, l1) ← különbség(x, n, s, 1)
9.          (r2, l2) ← f(r1, l1, e, y, m)
10.         (r3, l3) ← ragasztás(s, 1, r2, l2)
11.         visszatérít r3, l3
12.     különben
13.         (r1, l1) ← f(y, m, e, x, n)
14.         s ← []
15.         s[1] ← x[1]
16.         (r2, l2) ← különbség(x, n, s, 1)
17.         (r3, l3) ← f(r2, l2, e, y, m)
18.         (r4, l4) ← ragasztás(r1, l1, r3, l3)
19.         visszatérít r4, l4
20.     vége(ha)
21. Vége(algoritmus)
```

Döntsétek el, hogy az alábbi állítások közül melyek igazak:

A. Az `f(x, n, e, y, m)` algoritmus létrehoz egy egydimenziós tömböt, amely az ***x*** sorozatból törli az ***e*** értékű elemeket és minden felszabaduló helyre beszúrja az ***y*** sorozat elemeit. Az algoritmus visszatéríti a létrehozott tömböt és ennek méretét.
B. Ha az ***x*** és ***y*** sorozatoknak nincs közös elemük, akkor az `f(x, n, e, y, m)` algoritmus által visszatérített sorozat csak különböző elemeket tartalmaz.
C. Ha az ***x*** és ***y*** sorozatok nem üresek, az `f(x, n, e, y, m)` algoritmus által visszatérített tömbméret lehet ***n***-nél kisebb.
D. Ha a 18. sorban ***r1*** és ***l1*** helyett ***y*** és ***m*** lenne, akkor az algoritmus egy olyan egydimenziós tömböt (és annak méretét) térítené vissza, amely az ***y*** sorozat elemeivel kezdődne, majd következnének az ***x*** sorozat elemei, ahol az ***e*** értékű elemek helyén az ***y*** sorozat elemei lennének.

30. Legyen az `s(a, b, c)` algoritmus, ahol ***a***, ***b***, ***c*** nullától különböző természetes számok és ***b*** ≥ ***a***.

```
Algoritmus s(a, b, c):
   Ha c = 0 akkor
      visszatérít 1
   különben
      Ha a > b akkor
         visszatérít (1 / a) * s(a - 1, b, c)
      különben
         Ha a < b akkor
            visszatérít (1 / b) * s(a, b - 1, c)
```

```
                különben
                   visszatérít c * s(a - 1, b - 1, c - 1)
                vége(ha)
             vége(ha)
          vége(ha)
       Vége(algoritmus)
```

Mely relációnak kell teljesülnie ***a***, ***b*** és ***c*** között ahhoz, hogy az algoritmus eredménye 1 / C_b^a legyen (C_b^a ***b*** elem ***a***-dik osztályú kombinációját jelöli).

A. ***a*** + ***b*** = ***c***
B. ***a*** + ***c*** = ***b***
C. ***b*** – ***c*** = ***a***
D. ***b*** + ***c*** = ***a*** – ***b***

BABEŞ-BOLYAI TUDOMÁNYEGYETEM

MATEMATIKA ÉS INFORMATIKA KAR

Felvételi verseny – 2021 július 19.

Informatika írásbeli

JAVÍTÁSI KULCS & MEGOLDÁSOK

HIVATALBÓL: 10 pont

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