Concurs de admitere, Varianta 1
Facultatea de Matematică și Informatică
Universitatea de Vest din Timișoara

Sesiunea Iulie, 2025

1. Timp total de lucru: 3 ore
2. Toate subiectele sunt obligatorii!
3. Citiți cu atenție informațiile legate de tipurile de întrebări, prezentate mai jos.

Tipuri de întrebări

Pe foaia de concurs veți întâlni trei tipuri de întrebări. Pentru a le recunoaște mai ușor, în partea dreaptă veți regăsi un șablon pentru răspunsuri, cu indicații legate de tipul de întrebare.

Tipul 1 Pentru întrebările de acest tip veți marca pe foaia de concurs un singur răspuns corect. Le veți recunoaște după modelul indicat în partea dreaptă a acestui rând.

MODEL, marcați exact un răspuns: 1 2 3 4

Punctaj pe întrebare: 6p (standard)

Tipul 2 Pentru întrebările de acest tip veți marca pe foaia de concurs fie un singur răspuns corect, fie două răspunsuri corecte. Le veți recunoaște după modelul indicat în partea dreaptă a acestui rând.

MODEL, marcați 1-2 răspunsuri: 1 2 3 4 5

Punctaj maxim pe întrebare: 6p (standard)+1p (bonus pentru răspuns complet)

Tipul 3 Pentru întrebările de acest tip veți marca pe foaia de concurs fie un singur răspuns corect, fie două răspunsuri corecte, fie trei răspunsuri corecte. Le veți recunoaște după modelul indicat în partea dreaptă a acestui rând.

MODEL, marcați 1-3 răspunsuri: 1 2 3 4 5

Punctaj maxim pe întrebare: 6p (standard)+1p (bonus pentru răspuns complet)

Punctaj

Fiecare întrebare valorează 6 puncte. Dacă o întrebare are mai multe variante corecte, atunci cele 6 puncte se vor împărți egal la numărul de variante corecte. Dacă răspunsul oferit identifică toate variantele corecte de răspuns, la punctajul întrebării se adaugă un bonus pentru răspuns complet de 1 punct. Orice răspuns greșit anulează întregul punctaj al întrebării respective, indiferent de numărul de răspunsuri corecte.
La punctajul obținut se adaugă 10 puncte, pentru start.
Pentru nota 5 sunt necesare 52 de puncte, inclusiv punctele de start, pentru nota 10 sunt necesare cel puțin 130 de puncte, inclusiv punctele de start.

Terminologie și convenții

Tipuri de date

int — număr întreg
real — număr real
boolean — valoare logică

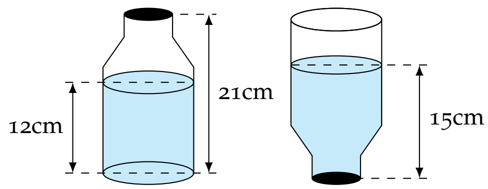
Operații și simboluri

a **DIV** b — calculează câtul împărțirii întregi (cu rest) a lui a la b
a **MOD** b — calculează restul împărțirii întregi a lui a la b
a **AND** b — conjuncția logică
a **OR** b — disjuncția logică
NOT a — negarea logică
$a = 2$ — operația de atribuire a valorii 2 pentru variabila a
$a == 2$ — verificarea egalității variabilei a cu valoarea 2
$a \neq 2$ — verificarea inegalității variabilei a cu valoarea 2

Partea I

Problema 1: Sticla cu apă

O sticlă cu baza cilindrică este umplută parțial cu apă, închisă etanş cu un capac, şi pusă pe o masă orizontală plată. Ce fracție din volumul sticlei este plină cu apă dacă ştim nivelul apei din sticlă când sticla este pusă normal pe masă, şi când sticla este pusă cu fundul în sus – vezi figurile de mai jos.



Ciornă, marcați exact un răspuns: (1) (2) (3) (4)

(1) 2/3 ✓ (2) 4/7 (3) 5/7 (4) 3/4

Explicații

Spaţiul gol din sticlă este un cilindru înalt de $21 - 15 = 6$cm iar spaţiul cu apă este un cilindru înalt de 12cm. Deci fracţia cu apă din volumul sticlei este $12/(12+6) = 2/3$.

Problema 2: Arbori

Se consideră graful neorientat G care are mulţimea de noduri $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$ şi mulţimea de muchii $\{(i, j) \mid i^2 < j\}$. Câte muchii trebuie eliminate din G pentru ca acesta să devină arbore?

Ciornă, marcați exact un răspuns: (1) (2) (3) (4)

(1) 1 (2) 5 (3) 7 ✓ (4) 8

Explicații

NOTĂ: Enunțul problemei conține o eroare de tehnoredactare: mulțimea de noduri ar fi trebuit să fie $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$.
Toți candidații vor obține punctaj maxim la această problemă.
Justificare răspuns: Mulţimea de muchii a lui G este $\{(1, j) \mid 2 \leq j \leq 10\} \cup \{(2, j) \mid 5 \leq j \leq 10\} \cup \{(3, j) \mid 10 \leq j \leq 10\}$. G are $9 + 6 + 1 = 16$ muchii. Se știe că un arbore cu n noduri are $n - 1$ muchii. G are 10 noduri, deci un arbore cu nodurile lui G are 9 muchii. Rezultă că trebuie eliminate $16 - 9 = 7$ muchii.

Problema 3: O problemă de căutare binară

Fie $x[1..127]$ tabloul tuturor întregilor din mulţimea $\{1, 2, \ldots, 64\} \cup \{2n + 3 \mid 64 < n < 128\}$, ordonate strict crescător, şi M mulţimea elementelor care pot fi găsite în x de către o căutare binară după compararea elementului căutat cu 4 elemente din x.

Care din următoarele secvenţe de numere reprezintă elementele mulţimii M, ordonate crescător?

Ciornă, marcați exact un răspuns: (1) (2) (3) (4)

① 1 31 33 63 132 193 197 257

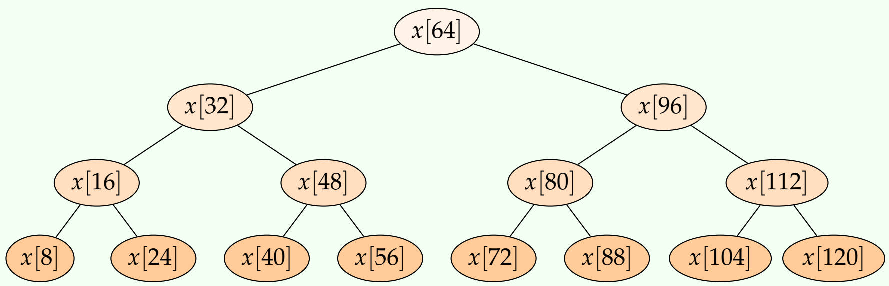
② 8 24 40 56 147 179 211 243 ✔

③ 16 48 163 227

④ 4 12 16 28 36 44 52 60 139 155 171 187 203 219 235 251

Explicații

Mulţimea M conţine elementele $\{x[8], x[24], x[40], x[56], x[72], x[88], x[104], x[120]\}$ din frunzele arborelui de căutare binară



În cazul nostru, $x[n] = n$ dacă $n \leq 64$ şi $x[n] = 2\,n + 3$ dacă $n > 64$.
Deci $M = \{8, 24, 40, 56, 147, 179, 211, 243\}$. Răspunsul corect este ②.

Problema 4: Suma vecinilor

Se consideră un tablou a de $n \geq 3$ numere întregi indexate de la 1 la n și subalgoritmul SumaVecinilor(a, n, k) care primește o poziție k (cu $2 \leq k \leq n-1$) și modifică elementul $a[k]$ în suma vecinilor săi.

```
procedura SumaVecinilor(int a[1..n], int n, int k)
    a[k] = a[k − 1] + a[k + 1]
```

Presupunem că înainte de apelul procedurii, tabloul b este o copie identică a lui a, adică $b[i] = a[i]$ pentru orice $i \in \{1, \ldots, n\}$.

Precizați care dintre următoarele afirmații este adevărată după execuția procedurii SumaVecinilor$(a, n, 2)$ urmată de execuția SumaVecinilor$(a, n, 3)$.

Ciornă, marcați exact un răspuns: 1 2 3 4

① $a[3] = b[2] + b[4]$

② $a[3] = b[1] + b[3] + b[4]$ ✔

③ $a[3] = 2 \cdot b[2] + b[3] + b[4]$

④ $a[3] = b[3]$

Explicații

Procedura modifică doar poziția k, înlocuind valoarea cu suma vecinilor. Așadar, $a[2]$ devine suma valorilor anterioare de pe pozițiile 1 și 3, iar $a[3]$ folosește această sumă la care se adaugă $b[4]$

Problema 5: Împăratul vesel

Vesel Împărat observă cum lumea din cetate se îmbulzește să cumpere mere de la o căruță. Curios, acesta dă 8 galbeni unui argat să cumpere mere de toți cei 8 galbeni. Argatul păstrează jumătate din sumă și trimite un alt argat să cumpere mere. Procesul se tot repetă până la un argat care păstrează galbenul ajuns la el și confiscă toate merele din căruță.

Câți argați sunt implicați în acest proces?

Ciornă, marcați exact un răspuns:
1 2 3 4

1. 3 argați
2. 6 argați
3. 4 argați ✔
4. 5 argați

Explicații

Primul argat primește 8 galbeni, al doilea argat primește 4 galbeni, al treilea argat primește 2 galbeni, iar al patrulea va primi 1 galben. Prin urmare sunt 4 argați. De asemenea se poate observa că problema presupune împărțirea succesivă la 2 până la obținerea valorii 0. Pentru un număr 2^x este nevoie de $x+1$ împărțiri.

Problema 6: Creștere rapidă

Se consideră un recipient în care se adaugă apă astfel încât cantitatea de apă se dublează în fiecare zi. Știind că în ziua $n = 10$ recipientul a fost umplut cu apă, să se marcheze variantele corecte.

Ciornă, marcați exact un răspuns:
1 2 3 4

1. în ziua 4, 25% (o pătrime) din volumul recipientului este umplut cu apă
2. în ziua 5, jumătate din volumul recipientului este umplut cu apă
3. în prima zi, cantitatea de apă este egală cu zero
4. în prima zi, cantitatea de apă este sub 1% ✔

Explicații

Dacă în ziua 10 recipientul este umplut 100%, în ziua 9 trebuie să fie umplut în proporție de 50%, în ziua 8 în proporție de 25% ș.a.m.d. Prin urmare, răspunsurile 1 și 2 nu pot fi corecte. Având în vedere că recipientul nu poate fi gol la început, nici răspunsul 3 nu este corect. Deci răspunsul corect este 4

Problema 7: Găsește intrusul

Se dau reprezentări ale aceluiași număr în baze diferite. Determinați pe cea care *nu* este egală cu 42 în baza 10.

Ciornă, marcați exact un răspuns:
1 2 3 4

1. 101010_2
2. 52_8
3. $2A_{16}$
4. 104_7 ✔

Explicații

① $101010_2 = 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 32 + 8 + 2 = 42.$

② $52_8 = 5 \cdot 8^1 + 2 \cdot 8^0 = 40 + 2 = 42.$

③ $2A_{16} = 2 \cdot 16^1 + 10 \cdot 16^0 = 32 + 10 = 42.$

④ $104_7 = 1 \cdot 7^2 + 0 \cdot 7^1 + 4 \cdot 7^0 = 49 + 0 + 4 = 53 \neq 42$. Prin urmare acesta este intrusul.

Partea a II-a

Problema 8: Ionică şi mărul fermecat

În mărul fermecat din Gradina Hesperidelor sunt 35 mere de aur şi 30 mere de argint. În fiecare seară, Ionică Fărăfrică ia două mere din pom, dar imediat în pom creşte alt măr. Dacă ia un măr de aur şi unul de argint, în locul lor creşte un măr de aur. Dacă merele culese sunt de acelaşi fel, în locul lor creşte un măr de argint.

Care din afirmaţiile următoare sunt corecte?

Ciornă, marcați 1-2 răspunsuri: 1 2 3 4 5

① Ultimul măr rămas în pom este de aur. ✔

② Ultimul măr rămas în pom este de argint.

③ Ultimele două mere culese de Ionică sunt de aur.

④ Ultimele două mere culese de Ionică sunt de argint.

⑤ Ultimele două mere culese de Ionică sunt diferite: unul de aur şi unul de argint. ✔

Explicații

Numărul de mere de aur fie rămâne constant (când se culeg fie două mere diferite, fie două mere de argint) fie descrește cu 2 (când se culeg două mere de aur). Intrucât numărul inițial de mere de aur este impar, el va rămâne tot timpul impar. Prin urmare răspunsurile ②, ③ şi ④ nu pot fi corecte.

Problema 9: Sortare perturbată

Presupunem că $x[1..n]$ este un tablou global de $n > 2$ numere întregi distincte, sortat în ordine strict crescătoare. Vlad perturbă tabloul x alegând trei indecşi $1 \le i < j < k \le n$ şi modificând valorile elementelor $x[i], x[j], x[k]$ cu operaţia PERTURBA(i, j, k) definită de procedura

```
procedura PERTURBA(int i, int j, int k)
    int tmp
    tmp = x[i], x[i] = x[j], x[j] = x[k], x[k] = tmp
```

Alin, care nu ştie indecşii i, j, k folosiţi de Vlad, trebuie să readucă tabloul x la starea neperturbată. Alin propune următorii algoritmi:

```
procedura REPARA1()
    int i = 1, j, k = n
    cât timp x[k − 1] < x[k] execută
        k = k − 1
    cât timp x[i] < x[k] execută
        i = i + 1
    j = i + 1
    cât timp x[j] < x[i] execută
        j = j + 1
    PERTURBA(k, j, i)
```

```
procedura REPARA2()
    int i = 1, j, k = n
    cât timp x[k − 1] < x[k] execută
        k = k − 1
    cât timp x[i] < x[k] execută
        i = i + 1
    j = i + 1
    cât timp x[j] < x[i] execută
        j = j + 1
    PERTURBA(i, j, k)
```

```
procedura REPARA3()
    int i = 1, j, k = n
    cât timp x[i] < x[i + 1] execută
        i = i + 1
    cât timp x[i] < x[k] execută
        i = i + 1
    j = i + 1
    cât timp x[j] < x[i] execută
        j = j + 1
    PERTURBA(k, j, i)
```

```
procedura REPARA4()
    int i, tmp
    pentru i = 1 to n − 1 execută
        dacă x[i] > x[i + 1] atunci
            tmp = x[i], x[i] = x[i + 1], x[i + 1] = tmp
    pentru i = n downto 2 execută
        dacă x[i − 1] > x[i] atunci
            tmp = x[i − 1], x[i − 1] = x[i], x[i] = tmp
```

Care din algoritmii propuşi de Alin îi rezolvă problema? Marcaţi toate răspunsurile corecte.

Ciornă, marcați 1-2 răspunsuri:
1 2 3 4 5

1. REPARA1✓
2. REPARA2
3. REPARA3
4. REPARA4✓
5. Nici unul din aceşti algoritmi.

Explicații

Operaţia PERTURBA(k, j, i) efectuată imediat după PERTURBA(i, j, k) reface tabloul x. Deci, dacă Alin reuşeste să deducă valorile i, j, k folosite de Vlad, atunci poate readuce tabloul x la starea neperturbată.
REPARA1() identifică valorile corecte ale lui i, j, k înainte să apeleze PERTURBA(k, j, i), deci rezolva problema lui Alin.
REPARA2() identifică valorile corecte ale lui i, j, k, dar PERTURBA(i, j, k), nu readuce tabloul x la starea neperturbată.
REPARA3() nu identifică valorile corecte ale lui i, j, k: dacă $x[1..4] = [1, 2, 3, 4]$ este perturbat de Vlad cu PERTURBA(1,2,3) atunci REPARA3() deduce eronat că $i = 2$, $j = 4$, $j = 3$ şi îl aduce pe x în starea $x[1..4] = [2, 1, 4, 3]$.
REPARA4() este inspirat de metoda bulelor (*Bubblesort*) şi readuce tabloul perturbat la starea iniţială în felul următor:

$$x[1], \ldots x[i-1], x[j], x[i+1], \ldots, x[j-1], x[k], x[j+1], \ldots, x[k-1], x[i], x[k+1], \ldots, x[n]$$

$$\Big\Downarrow \text{ prima buclă } \textbf{cât timp}$$

$$x[1], \ldots, x[i-1], x[i+1], \ldots, x[k-1], x[i], x[k], \ldots, x[n]$$

$$\Big\Downarrow \text{ a doua buclă } \textbf{cât timp}$$

$$x[1], \ldots, x[n] \text{ neperturbat}$$

Deci răspunsurile corecte sunt 2 şi 4.

Problema 10: Step

Roboţelul STEP poate merge doar în linie dreaptă, iar printr-un singur "pas" poate parcurge 10cm, 20cm sau 40cm. La oricare dintre paşi, STEP alege una dintre cele trei variante, şi am vrea să aflăm în câte moduri poate parcurge distanţa de 1 metru.

Se consideră următoarele funcţii:

```
funcția S1(int n): int
    dacă n < 0 atunci
        întoarce 0
    dacă n == 0 atunci
        întoarce 1
    întoarce S1(n − 10) + S1(n − 20) + S1(n − 40)

funcția S3(int n): int
    dacă n == 10 atunci
        întoarce 1
    dacă n == 20 atunci
        întoarce 2
    dacă n == 40 atunci
        întoarce 6
    întoarce S3(n − 10) + S3(n − 20) + S3(n − 40)

funcția S2(int n): int
    dacă n < 0 atunci
        întoarce 0
    dacă n == 0 atunci
        întoarce 1
    întoarce S2(n − 1) + S2(n − 2) + S2(n − 4)

funcția S4(int n): int
```

dacă n **DIV** $10 == 1$ **atunci**
  întoarce 1
dacă n **DIV** $20 == 1$ **atunci**
  întoarce 2

dacă n **DIV** $40 == 1$ **atunci**
  întoarce 6
întoarce $S4(n-1) + S4(n-2) + S4(n-4)$

Care dintre apelurile următoare (corespunzătoare funcţiilor definite mai sus) permit obţinerea răspunsului? Marcaţi toate variantele corecte.

Ciornă, marcați 1-2 răspunsuri: 1 2 3 4 5

1. S1(100). ✔
2. S2(10). ✔
3. S3(100).
4. S4(100).
5. Niciuna dintre variante.

Explicații

Notăm cu $S(n)$ numărul de variante de paşi prin care poate fi parcursă distanţa n. Ultimul pas efectuat poate fi de lungime 10, 20 sau 40, prin urmare numărul total de variante este $S(n-10) + S(n-20) + S(n-40)$. Dacă $n = 10$ atunci există o singură variantă (efectuarea unui pas de lungime 10). Dacă $n = 20$ atunci există două variante (efectuarea unui pas de lungime 20 sau a doi paşi de lungime 10). Dacă $n = 40$ atunci există 6 variante ($n = 40$, $n = 10+10+10+10$, $n = 20+10+10$, $n = 10+20+10$, $n = 10+10+20$, $n = 20+20$). Prin urmare, pentru $n = 10k$ (orice distanţă care este multiplu de 10), relaţia de recurenţă care permite calculul lui $S(n)$ este:

$$S(n) = \begin{cases} 0 & \text{dacă } n < 0 \\ 1 & \text{dacă } n = 0 \\ S(n-10) + S(n-20) + S(n-40) & \text{dacă } n \in \{10, 20, \ldots, 100\} \end{cases}$$

Algoritmul care implementează această relaţie de recurenţă este algoritmul S1. Intrucât numărul de variante nu depinde de dimensiunea pasului, relaţia de recurenţă poate fi rescrisă:

$$S(k) = \begin{cases} 0 & \text{dacă } k < 0 \\ 1 & \text{dacă } k = 0 \\ S(k-1) + S(k-2) + S(k-4) & \text{dacă } k \in \{1, 2, \ldots, 10\} \end{cases}$$

Prin urmare şi algoritmul S2 cu apelul S2(10) este corect.
Algoritmii S3 şi S4 sunt incorecţi pentru că nu includ cazurile particulare când n ajunge mai mic decât 10.

Problema 11: Antisimetrie

O matrice pătratică de numere întregi se numește antisimetrică dacă elementele simetrice față de diagonala sa principală sunt opuse una alteia. De exemplu, dacă

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ -2 & 1 & -5 & 6 \\ -3 & 5 & 9 & 7 \\ -4 & -6 & -7 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 7 & 6 & 5 \\ 2 & 1 & 4 & 3 \\ 3 & 5 & 1 & 2 \\ 4 & 6 & 7 & 1 \end{pmatrix}$$

atunci matricea A este antisimetrică, dar B nu este.

```
funcția ANTISIMETRICA1(int a[1..n, 1..n]): boolean
    int i, j
    pentru i = 1 to n − 1 execută
        pentru j = i + 1 to n execută
            dacă a[i][j] ≠ −a[j][i] atunci
                întoarce false
    întoarce true
```

```
funcția ANTISIMETRICA2(int a[1..n, 1..n]): boolean
    int i, j
    pentru i = n − 1 to 1 pas −1 execută
        pentru j = n to i + 1 pas −1 execută
            dacă −a[i][j] ≠ a[j][i] atunci
                întoarce false
    întoarce true
```

```
funcția ANTISIMETRICA3(int a[1..n, 1..n]): boolean
    int i, j
    pentru i = 1 to n execută
        pentru j = 1 to n execută
            dacă i ≠ j AND a[i][j] ≠ −a[j][i] atunci
                întoarce false
    întoarce true
```

Ciornă, marcați 1-2 răspunsuri:

1 2 3 4 5

Care dintre următoarele afirmații sunt adevărate?

1. Toate cele 3 funcții returnează true dacă matricea a este antisimetrică şi false în caz contrar ✔
2. Există cel puțin o funcție care să nu verifice corect că matricea a este antisimetrică
3. Numărul de evaluări ale condiției din instrucțiunea **dacă** este egal pentru funcțiile ANTISIMETRICA1() și ANTISIMETRICA2() dacă matricea **nu** este antisimetrică
4. Numărul de evaluări ale condiției din instrucțiunea **dacă** este egal pentru funcțiile ANTISIMETRICA1() și ANTISIMETRICA2() dacă matricea este antisimetrică ✔
5. Toate cele trei funcții evaluează de același număr de ori condiția instrucțiunii **dacă**

Explicații

Toate variantele verifică corect proprietatea de antisimetrie doar că în moduri diferite: ANTISIMETRICA1 parcurge elementele de sub diagonala principală; ANTISIMETRICA2 parcurge elementele deasupra diagonalei principale și ANTISIMETRICA3 parcurge toate elementele matricii.

Problema 12: Împarte și adună

Se consideră funcția ÎMPARTE2 definită mai jos care primește ca parametru un număr natural n.

```
funcția ÎMPARTE2(int n): int
    int v, s = 0
    cât timp n ≠ 0 execută
        v = n MOD 7
        dacă v MOD 2 == 0 atunci
            s = s + v
        n = n DIV 7
    întoarce s
```

Să se marcheze răspunsurile corecte.

Ciornă, marcați 1-2 răspunsuri: 1 2 3 4 5

1. apelul ÎMPARTE2(50) returnează 1.
2. funcția ÎMPARTE2(n) returnează întotdeauna o valoare pară. ✔
3. funcția ÎMPARTE2(n) returnează întotdeauna o valoare impară.
4. există valori pentru n astfel încât apelul ÎMPARTE2(n) returnează o valoare impară.
5. apelul ÎMPARTE2(63) returnează 2. ✔

Explicații

Funcția ÎMPARTE2 calculează suma cifrelor pare din reprezentare în baza 7 a numărului n; o sumă de numere pare este de asemenea un număr par.

Problema 13: O traversare în adâncime

Se consideră labirintul ilustrat mai jos, în care celulele negre reprezintă obstacole.

	1	2		3
4		G	5	6
S	7		8	
9	10	11	12	13
		14	15	

Se pune problema găsirii unui drum de la celula de start S la celula G cu ajutorul metodei de parcurgere în adâncime (Depth First). Pe durata parcurgerii în adâncime se aplică următoarele reguli:

1. Vizitarea celulelor se face în ordinea: sus, dreapta, jos, stânga. Nu este posibilă mutarea pe diagonală.
2. Nici o celulă nu se poate vizita de 2 ori.

De exemplu, căutarea în adâncime a unui drum pornind de la S începe cu S, 4, 7, 10, 11 în această ordine. Fie σ secvenţa de celule de pe drumul găsit de căutarea în adâncime, care începe cu S şi se termină cu G.

Care din afirmaţiile următoare sunt adevărate? Marcaţi toate variantele corecte.

Ciornă, marcați 1-2 răspunsuri: 1 2 3 4 5

① σ are 8 celule. ③ 13 este în σ. ⑤ 15 este în σ.

② σ are 11 celule.✔ ④ 3 este în σ.✔

Explicații

Întrucât $\sigma = [S, 4, 7, 10, 11, 12, 8, 5, 6, 3, G]$, răspunsurile corecte sunt ② și ④.

Partea a III-a

Problema 14: Judeţe pivot

În turul final al alegerilor prezidențiale din Infolanda, fiecare dintre cele n judeţe publică o *diferenţă netă de voturi* $V_i = (\text{nr. voturi pentru A}) - (\text{nr. voturi pentru B}), \quad i = 1, 2, \ldots, n$. Fie $T = \sum_{i=1}^{n} V_i$ marja națională. Procedura *flip* presupune inversarea tuturor buletinelor din județul i, obținând noua marjă $T' = T - 2V_i$. Definim județul i **pivot** dacă flip-ul schimbă rezultatul național, adică dacă transformă o victorie a lui A ($T > 0$) într-un $T' \leq 0$, sau o victorie a lui B ($T < 0$) într-un $T' \geq 0$, sau o egalitate ($T = 0$) în $T' \neq 0$.

Dat fiind algoritmul de mai jos, stabiliți care din variante sunt adevărate.

```
funcția RunMe(int V[1..n]) : int
    int T = 0, pivot = 0
    pentru i = 1 to n execută
        T = T + V[i]

    pentru i = 1 to n execută
        dacă V[i] ≠ 0  AND T · (T − 2 · V[i]) ≤ 0 atunci
            pivot = pivot + 1

    întoarce pivot
```

① Dacă marja națională totală T este 0, există cel puțin un județ care este pivot.

Ciornă, marcați 1-3 răspunsuri:
① ② ③ ④ ⑤

② Numărul de judeţe pivot nu poate fi mai mare de $n/2$.

③ Numărul maxim de judeţe pivot este n. ✔

④ Funcția RunMe returnează numărul de județe pivot dacă $T \neq 0$✔

⑤ Funcția RunMe returnează numărul de județe pivot dacă $T = 0$ ✔

Explicații

Algoritmul verifică schimbarea de semn atunci când se schimbă voturile. Dacă T este pozitiv și rămâne pozitiv după flip, atunci rezultatul înmulțirii este pozitiv. Dacă T este negativ și rămâne negativ după flip, rezultatul înmulțirii va fi pozitiv. Dacă după flip se schimbă semnul, rezultatul va fi negativ.
Atunci când $T = 0$, orice $V_i \neq 0$ poate fi pivot.

① Fals. Contra-exemplu: 2 județe în care cei doi candidați au fost la egalitate $V = [0, 0]$

② Fals. Există cazuri în care toate județele sunt pivot. De exemplu, pentru $n = 2$, e posibilă următoarea situație în care fiecare dintre cele două județe este pivot: fiecare candidat câștigă într-un județ și pierde în celălalt $V = [-1, 1]$.

③ Adevărat. Vezi exemplul precedent.

④ Adevărat.

⑤ Adevărat.

Problema 15: Ce face?

Se consideră funcţiile VERIFICA(n, x) şi ALEGE(n, x), unde $1 < n < 10^4$ este un număr natural iar x este un vector de n numere întregi indexate de la 1 la n. Aceste funcţii sunt definite astfel:

```
funcția VERIFICA(int n, int x[1..n]): boolean
    dacă n < 3 atunci
        întoarce false
    int i, p =ALEGE(n, x)
    dacă p == 1  OR p == n atunci
        întoarce false
    pentru i = 2 to p execută
        dacă x[i] ≥ x[i − 1] atunci
            întoarce false
    pentru i = p + 1 to n − 1 execută
        dacă x[i] ≥ x[i + 1] atunci
            întoarce false
    întoarce true
```

```
funcția ALEGE(int n, int x[1..n]): int
    int i
    int r = 1
    int v = x[1]
    pentru i = 2 to n execută
        dacă x[i] < v atunci
            r = i
            v = x[i]
    întoarce r
```

Care dintre afirmaţiile următoare sunt adevărate? Marcaţi toate răspunsurile corecte.

Ciornă, marcați 1-3 răspunsuri: ① ② ③ ④ ⑤

① Dacă vectorul x este ordonat descrescător și are cel puțin 3 elemente, atunci funcția VERIFICA(n, x) returnează true.

② Dacă $x = [12, 10, 8, 5, 9, 11, 15, 18]$ și $n = 8$ atunci funcția VERIFICA(n, x) returnează true.✓

③ Dacă $x = [20, 10, 5, 1, 2, 4, 6, 10, 8]$ și $n = 9$ atunci funcția VERIFICA(n, x) returnează false.✓

④ Dacă vectorul x este ordonat strict crescător și are cel puțin 3 elemente, atunci funcția VERIFICA(n, x) returnează true.

⑤ Dacă vectorul x este ordonat strict crescător atunci ALEGE(n, x) returnează indexul celui mai mic element din x. ✔

Explicații

Funcția ALEGE întoarce indicele celei mai mici valori din tablou. In cazul în care valoarea minimului apare de mai multe ori, atunci va întoarce cel mai mic indice.
Varianta ① este incorectă, deoarece este posibil ca funcția VERIFICA să returneze false (de exemplu când valoarea minimă este prezentă o singură dată în tabloul ordonat descrescător, ceea ce conduce la $p = n$).
Varianta ② este corectă, întrucât poziția minimul se află pe poziția 4 și elementele din fața sa sunt ordonate descrescător, iar cele aflate după minim sunt ordonate crescător.
Varianta ③ este corectă, deoarece $x[n-1] \geq x[n]$.
Varianta ④ este incorectă, deoarece dacă tabloul x este ordonat strict crescător atunci $p = 1$ și algoritmul va returna false.
Varianta ⑤ este corectă.

Problema 16: BAC

Se consideră funcţia BAC(a, n, p) unde n este un număr natural ($1 \leq n \leq 10^4$), p este un număr întreg ($-10^4 \leq p \leq 10^4$), iar $a[1..n]$ este un tablou de numere naturale nenule, nu neapărat distincte:

```
funcția BAC(int a[1..n], int n, int p): int
    dacă n < 1 atunci
        întoarce 0
    altfel
        dacă 1 ≤ p AND p ≤ n atunci
            întoarce a[p]
        altfel
            întoarce -1
```

Care dintre afirmaţiile următoare sunt adevărate? Marcaţi toate răspunsurile corecte.

Ciornă, marcați 1-3 răspunsuri: ① ② ③ ④ ⑤

① BAC(a, n, p) returnează -1 dacă și numai dacă $p \leq 0$ sau $p > n$. ✔

② BAC(a, n, p) returnează elementul de pe poziția p dacă $0 < p \leq n$.✔

③ BAC(a, n, p) nu returnează niciodată 0 pentru valori ale parametrilor care respectă precondițiile din enunț. ✔

④ BAC(a, n, p) returnează elementul de pe poziția p dacă $0 \leq p < n$.

⑤ BAC(a, n, p) returnează valoarea lui $a[1]$ dacă și numai dacă $p = 1$.

Explicații

Varianta (1) este corectă pentru că se returnează -1 doar dacă condiția $1 \leq p$ **AND** $p \leq n$ este falsă, adică $p \leq 0$ sau $p > n$.
Varianta (2) este corectă.
Varianta (3) este corectă pentru că $n \geq 1$ și tabloul nu conține elemente nule.
Varianta (4) este incorectă, deoarece indicii tabloului sunt între 1 și n.
Varianta (5) este incorectă, deoarece în vector valoarea $a[1]$ se poate regăsi pe mai multe poziții.

Problema 17: Suma parțială recursivă

Se consideră următorul algoritm recursiv Sum(n) care primește un număr natural $n \geq 1$ și returnează o valoare întreagă:

```
funcția Sum(int n): int
    dacă n = 1 atunci
        întoarce 1
    altfel
        întoarce n+ Sum(n DIV 2)
```

Marcați toate afirmațiile adevărate.

Ciornă, marcați 1-3 răspunsuri:
(1) (2) (3) (4) (5)

(1) Sum(20) presupune 4 apeluri ale funcției.

(2) Sum(20) presupune 5 apeluri ale funcției. ✔

(3) Sum(20) returnează valoarea 38. ✔

(4) Valoarea maximă ce poate fi returnată este $2n$

(5) Valoarea maximă ce poate fi returnată este $2n - 1$ ✔

Explicații

Se observă că apelurile funcției recursive Sum sunt pentru n egal cu 20, 10, 5, 2 și 1. Deci sunt 5 apeluri și funcția returnează suma: $20 + 10 + 5 + 2 + 1 = 38$.
Considerăm că n poate fi reprezentat pe k biți astfel $n = b_{k-1}2^{k-1} + b_{k-2}2^{k-2} + \ldots + b_1 2 + b_0$. Utilizând regula de calcul a sumei unei progresii geometrice de rație 2 se obține că suma calculată de către funcție este $S = b_{k-1}(2^k - 1) + b_{k-2}(2^{k-1} - 1) + \ldots + b_1(2^2 - 1) + b_0(2^1 - 1)$ care poate fi rescrisă ca $S = 2(b_{k-1}2^{k-1} + b_{k-2}2^{k-2} + \ldots + b_1 2 + b_0) - (b_{k-1} + b_{k-2} + \ldots + b_1 + b_0) = 2n - \sum_{i=0}^{k-1} b_i$ adică $2n$ minus numarul de biți egali cu 1 din reprezentarea în baza 2 a lui n. Valoarea maximă se obține când numărul de biți egali cu 1 din reprezentarea lui n este 1, adică $2n - 1$.

Problema 18: Permutarea elementelor pare

Se consideră un vector v de n numere întregi indexate de la 1 la n, unde $n > 2$ este par. Următorul algoritm modifică vectorul v în felul următor:

procedura EvenSwap(int $v[1..n]$, int n)
  pentru $i = 1$ to n step 2 **execută**
    dacă $v[i]$ **MOD** $2 == 0$ **AND** $v[i+1]$ **MOD** $2 == 0$ **atunci**
      swap($v[i]$, $v[i+1]$)

Se aplică procedura pe vectorul $v = [6, 4, 7, 8, 11, 10, 14, 16]$. Marcați toate afirmațiile adevărate despre vectorul rezultat.

Ciornă, marcați 1-3 răspunsuri: 1 2 3 4 5

① Elementele de pe pozițiile 1 și 2 sunt inversate. ✔

② Vectorul final conține aceleași elemente ca vectorul inițial. ✔

③ Elementele de pe pozițiile 3 și 4 sunt inversate.

④ Numărul de interschimbări efectuate este 2. ✔

⑤ Toate elementele pare din vector sunt ordonate crescător.

Explicații

Perechile de elemente din vectorul v parcurse se află pe pozițiile (1,2), (3,4), (5,6) și (7,8) deoarece pasul ciclului este 2. Procedura *swap* se apelează doar când ambele elemente sunt pare. Sunt efectuate 2 *swap*-uri pentru elementele de pe pozițiile: (1,2) și (7,8).

- Inițial: v = [6, 4, 7, 8, 11, 10, 14, 16]
- După aplicarea procedurii *swap* pentru perechea de indecși (1,2): v = [4, 6, 7, 8, 11, 10, 14, 16]
- După aplicarea procedurii *swap* pentru perechea de indecși (7,8): v = [4, 6, 7, 8, 11, 10, 16, 14]

Elementele rămân aceleași, dar reordonate. Doar răspunsurile ① , ②și ④ sunt corecte.

Problema 19: RunMe2

Se consideră procedura RunMe2 definită mai jos.

procedura RunMe2(int n)
  int $i = 1$
  cât timp $n > 0$ **execută**
    dacă n **MOD** $2 == 1$ **atunci**
      scrie i
    $i = i + 1$
    $n = n$ **DIV** 2

Să se marcheze răspunsurile corecte.

Ciornă, marcați 1-3 răspunsuri: 1 2 3 4 5

① pentru apelul RunMe2(126) se afișează secvența 234567 ✔

② pentru apelul RunMe2(11) se afișează secvența 123

③ pentru fiecare apel RUNME2(a), unde a este număr impar, prima valoare afișată este 1 ✓

④ pentru fiecare apel RUNME2(a), unde a este număr par, prima valoare afișată este 1

⑤ pentru fiecare apel RUNME2(a), unde a este număr par, prima valoare afișată este diferită de 1 ✓

Explicații

Procedura AFISARE2 afișează pozițiile elementelor egale cu 1 în reprezentarea în baza 2 a numărului n. Numărătoarea pozițiilor începe de la 1.

Problema 20: Injecție

Spunem despre o funcție că este *injectivă* dacă pentru valori diferite de intrare (input), funcția întoarce rezultate (output) diferite. Se consideră funcția SUM, definită mai jos, care primește ca dată de intrare un număr natural n ($1 \leq n \leq 10^6$) și întoarce un număr natural s calculat în felul următor:

```
funcția SUM(int n): int
    int c, s = 0
    dacă n < 10 atunci întoarce n
    cât timp n ≠ 0 execută
        c = n MOD 10
        s = s + c
        n = n DIV 10
    întoarce s
```

Să se marcheze variantele corecte.

Ciornă, marcați 1-3 răspunsuri:
① ② ③ ④ ⑤

① funcția SUM este injectivă

② SUM$(400) = 4$ ✓

③ există exact 5 date de intrare astfel încât valoarea returnată de funcție este egală cu 10

④ există cel puțin 2 date de intrare distincte a, b astfel încât SUM(a) = SUM(b) ✓

⑤ există exact 7 valori pentru care funcția SUM va returna 1 ✓

Explicații

Funcția dată primește un număr natural n și returnează suma cifrelor lui n. Funcția nu este injectivă conform definiției deoarece există multiple date de intrare care returnează același rezultat, exemplu SUM(400) = SUM(4000) = SUM(22) = 4; restul răspunsurilor corecte se deduc din observația anterioară. În intervalul $[1, 10^6]$ sunt 7 numere naturale care au suma cifrelor egală cu 1.