



**SUBIECTE DE EXAMEN. BACALAUREAT 2022-2025.
INFORMATICĂ C++**

**EDITURA CASA CORPULUI DIDACTIC PRAHOVA
PLOIEȘTI, 2026**

Autor:
Prof. Lungescu Violeta-Cristina

SUBIECTE DE EXAMEN
BACALAUREAT 2022-2025
INFORMATICĂ C++

- AUXILIAR CURRICULAR -

Toate drepturile asupra prezentei ediții aparțin
Editurii Casa Corpului Didactic Prahova

Editura Casa Corpului Didactic Prahova
Adresa: Ploiești, Str. Democrației, Nr. 35
Tel./Fax: 0244577338
Email: edituraccdprahova@ccdph.ro

ISBN 978-606-047-373-2

Ploiești, 2026

PREFAȚĂ

În contextul transformărilor accelerate generate de digitalizare și al cerințelor tot mai complexe ale societății bazate pe cunoaștere, informatica ocupă un rol esențial în formarea competențelor necesare elevilor de liceu pentru integrarea cu succes în mediul academic și profesional. Studiată în învățământul gimnazial și liceal la profilurile matematică-informatică, precum și la profilul științe ale naturii, disciplina Informatică reprezintă una dintre opțiunile pentru susținerea examenului național de bacalaureat, în cadrul probei la alegere a profilului și specializării – proba E.d).

Totodată, rezultatele obținute la această disciplină constituie un criteriu important de admitere în numeroase programe universitare din domeniul informaticii, automaticii, ingineriei, matematicii sau al altor specializări conexe. Dincolo de rolul său în evaluările naționale, informatica contribuie la dezvoltarea gândirii algoritmice, a raționamentului logic, a creativității și a capacității de analiză și rezolvare a problemelor, competențe considerate indispensabile într-o societate profund influențată de evoluția tehnologiilor digitale și a inteligenței artificiale, oferindu-le elevilor posibilitatea de a deveni creatori de soluții digitale, capabili să răspundă provocărilor viitorului.

Auxiliarul curricular **„Subiecte de examen – Bacalaureat 2022–2025 – Informatică C++”** a fost elaborat cu scopul de a sprijini elevii în pregătirea sistematică pentru examenul național de bacalaureat. Lucrarea reunește toate subiectele oficiale elaborate de Ministerul Educației și Cercetării în perioada 2022–2025, adresate profilurilor matematică-informatică intensiv informatică și matematică-informatică, incluzând, pentru fiecare an școlar, subiectele de la simularea națională, sesiunea specială, sesiunea iunie–iulie și sesiunea august.

Toate variantele de examen sunt preluate din sursele oficiale publicate pe platforma <https://subiecte.edu.ro> și sunt organizate într-o structură accesibilă și ușor de parcurs. Lucrarea este alcătuită din două părți: prima cuprinde enunțurile integrale ale subiectelor de examen, iar cea de-a doua oferă indicații de rezolvare și propune soluții complete, redactate în limbajul de programare C++, în conformitate cu programa de examen și cu cerințele actuale de evaluare.

Prin diversitatea problemelor incluse și prin abordarea lor metodică, acest auxiliar urmărește consolidarea cunoștințelor, dezvoltarea strategiilor eficiente de rezolvare și familiarizarea elevilor cu structura și nivelul de dificultate al subiectelor de bacalaureat. Lucrarea se adresează atât elevilor care doresc să obțină rezultate foarte bune la examen, cât și profesorilor care caută un suport didactic util în activitatea de pregătire și evaluare.

Partea I – ENUNȚURI

Varianta - Simulare 2022

SUBIECTUL I

(20 de puncte)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Variabila x este de tip întreg. Indicați o expresie care are valoarea 1 dacă și numai dacă expresia C/C++ alăturată are valoarea 1.

a. $!(x < 3) \ \&\& \ x < 30$

c. $!(x < 3 \ || \ x <= 30)$

b. $!(x >= 3 \ \&\& \ x <= 30)$

d. $!(x > 3) \ || \ !(x <= 30)$
2. Subprogramul f este definit alăturat.

```
int f(int st,int dr,int v[])
{ if(st==dr) if(v[st]==st) return 1;
  else return 0;
  else return f(st,(st+dr)/2,v) + f(1+(st+dr)/2,dr,v);
}
```

Știind că variabila x memorează un tablou unidimensional cu elementele $(7, 1, 2, 5, 4)$, în această ordine, numerotate de la 0 la 4, indicați valoarea $f(0, 4, x)$.

a. 2

b. 3

c. 4

d. 5
3. Un sport se desfășoară pe o pistă înghețată, cu o echipă alcătuită din 4 jucători, care sunt, în această ordine, lider, secund, vice-skip și skip. Lotul este format din jucătorii din mulțimea {Alin, Bogdan, Cristian, Răzvan, Ștefan, Valentin}; Cristian poate fi doar lider, Alin și Valentin pot fi doar lider sau secund, Bogdan poate fi doar secund, iar ceilalți doi jucători pot fi doar vice-skip sau skip. Două echipe sunt distincte dacă există cel puțin un rol care, în cele două echipe, este atribuit unor jucători diferiți. Pentru antrenamente, s-au format echipe în toate formulele posibile, primele șase soluții generate fiind, în această ordine: {Alin, Bogdan, Răzvan, Ștefan}, {Alin, Bogdan, Ștefan, Răzvan}, {Alin, Valentin, Răzvan, Ștefan}, {Alin, Valentin, Ștefan, Răzvan}, {Cristian, Alin, Răzvan, Ștefan}, {Cristian, Alin, Ștefan, Răzvan}. Indicați soluția generată imediat după {Cristian, Valentin, Ștefan, Răzvan}.

a. {Cristian, Bogdan, Răzvan, Ștefan}

b. {Cristian, Bogdan, Ștefan, Răzvan}

c. {Valentin, Alin, Răzvan, Ștefan}

d. {Valentin, Bogdan, Ștefan, Răzvan}
4. Variabila $oras$, declarată alăturat, memorează unele date obținute în urma recensământului populației în 100 dintre orașele țării. Știind că densitatea unui oraș este egală cu raportul dintre numărul de locuitori și suprafața acestuia (exprimată în km^2), indicați o expresie a cărei valoare este egală cu densitatea primului oraș.

```
struct recensamant
{ char nume[21];
  int nrLocuitori;
  float suprafata;
}oras[100];
```

a. $oras[0].nrLocuitori/oras[0].suprafata$

c. $nrLocuitori[0].oras/suprafata[0].oras$

b. $oras.nrLocuitori[0]/oras.suprafata[0]$

d. $nrLocuitori.oras[0]/suprafata.oras[0]$
5. Un graf neorientat complet are 4 noduri. Indicați o valoare care NU poate fi numărul de muchii ale unui subgraf al acestui graf.

a. 0

b. 1

c. 2

d. 3

SUBIECTUL al II-lea

(40 de puncte)

1. Algoritmul alăturat este reprezentat în pseudocod. S-a notat cu $a \div b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $[a]$ partea întreagă a numărului real a .
- a. Scrieți ce se afișează în urma executării algoritmului dacă se citesc, în această ordine, numerele 6, 16, 4273, 1679, 3165, 16, 200, 167. (6p.)
- b. Dacă primele două numere citite sunt, în această ordine, 2 și 22, scrieți o secvență de numere distincte din intervalul $[10^3, 10^4]$ care pot fi citite în continuare astfel încât, în urma executării algoritmului, să se afișeze valoarea 2. (6p.)
- c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
- d. Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind adecvat prima structură `cât timp...execută` cu o structură repetitivă de tip `pentru...execută`. (6p.)
2. Într-un arbore cu rădăcină un nod se află pe nivelul x dacă lanțul elementar care are o extremitate în nodul respectiv și cealaltă extremitate în rădăcina arborelui are lungimea x . Pe nivelul 0 se află un singur nod (rădăcina). Un arbore cu rădăcină are 10 noduri și este reprezentat prin vectorul de „tați” $(10, 5, 5, 10, 0, 10, 9, 2, 5, 2)$. Enumerați trei dintre frunzele situate în arbore pe niveluri impare. (6p.)
3. Într-o seră se folosesc vase pătrate, care conțin câte o singură floare (lalea, frezie sau narcisă). Vasele sunt așezate pe trei rânduri și sunt aliniate ca în exemplu. Patru flori de același tip formează un romb dacă vasele în care se află sunt dispuse în jurul unui alt vas, cu care fiecare are în comun câte o latură. Variabilele j și ok sunt întregi, iar variabila a este un tablou bidimensional cu 3 linii și 15 coloane, cu elemente de tip `char`, în care fiecare linie, în ordine, memorează succesiunea de flori aflate pe un rând al serei, notându-se cu **L** lalelele, cu **F** freziile și cu **N** narcisele. Fiecare linie a tabloului corespunde, în ordine, unui rând de vase din seră. Liniiile și coloanele sunt numerotate începând de la 0. Scrieți secvența C/C++ de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila ok să aibă valoarea 1 dacă florile formează cel puțin un romb, sau valoarea 0 în caz contrar. Exemplu: în urma executării secvenței pentru tabloul de mai jos, $ok=1$ (sunt evidențiate 4 romburi).

```
ok=.....;
for(j=.....;j<.....;j++)
    .....
```

L	L	L	N	L	F	F	F	N	F	N	F	N	F	F
N	L	L	L	N	L	L	F	F	F	F	N	F	N	F
F	N	L	F	L	N	F	L	F	F	N	L	F	F	N

(6p.)

SUBIECTUL al III-lea

(30 de puncte)

1. Subprogramul `rest` are patru parametri:
- x , y și n , prin care primește câte un număr natural din intervalul $[1, 10^6]$, $x < y < n$;
 - k , prin care furnizează cea mai mare valoare naturală din intervalul $[1, n]$ pentru care atât restul împărțirii la x , cât și restul împărțirii la y , sunt egale cu 2, sau 0 dacă nu există o astfel de valoare.
- Scrieți definiția completă a subprogramului.
- Exemplu: pentru $x=10$, $y=101$ și $n=3000$, subprogramul returnează numărul 2022 (pentru numerele 2, 1012 și 2022 atât restul împărțirii la 10, cât și restul împărțirii la 101, este 2). (10p.)
2. Un text, de cel mult 250 de caractere, reprezintă o listă cu date de identificare ale invitaților la o petrecere; fiecare invitat are un prenume și un nume, care apar în listă în această ordine, urmate de simbolul ; (punct și virgulă), ca în exemplu. Numele și prenumele sunt formate din câte un singur cuvânt, compus din litere mari ale alfabetului englez, și sunt separate printr-un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat mai sus apoi, de pe rândul următor, un cuvânt, x , și afișează pe ecran, separate prin câte un spațiu, numele tuturor invitaților care au prenumele x , ca în exemplu, sau mesajul **NU** dacă nu există astfel de invitați. Exemplu: dacă lista este **DAN MARIS**; **DANIILA PREPELEAC**; **DAN POPA**; **EDANA DAN**; și cuvântul x este **DAN** se afișează pe ecran **MARIS POPA** (10p.)
3. Se citește de la tastatură un număr natural, n ($n \in [1, 10^9]$), și se cere să se scrie în fișierul text `bac.txt` cel mai mare număr natural p cu proprietatea că numărul $45p$ este divizor al numărului obținut prin evaluarea produsului $1 \cdot 2 \cdot 3 \cdot \dots \cdot n$. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al memoriei utilizate. Exemplu: dacă $n=14$, fișierul conține numărul 2 ($45 \cdot 2 = 90$ este divizor al lui $1 \cdot 2 \cdot 3 \cdot \dots \cdot 14 = 87178291200$)
- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)
- b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 4 – Sesiunea specială 2022

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Indicați expresia C/C++ cu valoarea 2022.

- a. $4044/4/2$ b. $4044/(4*2)$ c. $1011*1+1$ d. $1011*(1+1)$

2. Subprogramul `f` este definit alăturat. Indicați valoarea parametrului `x` pentru care `f(x,3)` are valoarea 2022.

```
int f(int n,int p)
{ if(n==0) return 0;
  else if(p==0) return n*10+2;
    else return f(n/10,p-1);
}
```

- a. 120234 b. 122023 c. 202345 d. 220223

3. Utilizând metoda backtracking, se generează toate posibilitățile de a expune câte trei mașini produse în România, din gama Dacia, din mulțimea {Dokker (negru), Duster (negru), Lodgy (gri), Logan (alb), Sandero (roșu)}, astfel încât să nu fie expuse simultan două mașini de aceeași culoare, Sandero să NU apară pe prima poziție, iar Logan să NU fie înainte de Dokker sau Duster. Două soluții sunt distincte dacă au cel puțin o mașină diferită sau dacă ordinea mașinilor este diferită. Primele cinci soluții generate sunt, în această ordine (Dokker, Lodgy, Logan), (Dokker, Lodgy, Sandero), (Dokker, Logan, Lodgy), (Dokker, Logan, Sandero), (Dokker, Sandero, Lodgy). Indicați penultima soluție, în ordinea generării acestora.

- a. Logan, Lodgy, Sandero b. Logan, Sandero, Lodgy
c. Lodgy, Dokker, Logan d. Lodgy, Logan, Dokker

4. Variabilele `s` și `d` permit memorarea câte unui șir cu maximum 50 de caractere. Indicați șirul memorat prin intermediul variabilei `d` în urma executării secvenței alăturate.

```
strcpy(s,"info@rocnee.eu");
s[strlen(s)-3]='\0';
strcpy(d,strchr(s,'@')+1);
```

- a. info@ b. info@ c. rocnee d. rocnu

5. Într-un arbore cu 7 noduri, numerotate de la 1 la 7, nodul 1 este rădăcină și pentru orice nod numerotat cu i ($i \in [1,7]$), dacă se notează cu $F(i)$ numărul de descendenți direcți („fii”) ai săi, atunci $F(i)=0$, dacă i este „frunză”, altfel $F(i)=i+1$. Indicați numărul maxim de „frați” ai nodului 7.

- a. 4 b. 3 c. 2 d. 1

SUBIECTUL al II-lea**(40 de puncte)**

1. Algoritmul alăturat este reprezentat în pseudocod. S-a notat cu $a \div b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $[c]$ partea întreagă a numărului real c .
- a. Scrieți ce se afișează în urma executării algoritmului dacă se citește numărul 1237518. (6p.)
- b. Scrieți două numere din intervalul $[100, 999]$ care pot fi citite, astfel încât, pentru fiecare dintre acestea, în urma executării algoritmului, să se afișeze două numere. (6p.)
- c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
- d. Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind structura **repetă... până când** cu o structură repetitivă cu test inițial. (6p.)
2. Un graf neorientat cu 6 noduri, numerotate de la 1 la 6, are muchiile $[1, 2]$, $[1, 3]$, $[1, 4]$, $[1, 5]$, $[2, 3]$, $[2, 4]$, $[2, 5]$, $[3, 5]$, $[4, 6]$, $[5, 6]$. Scrieți mulțimea muchiilor unui subgraf al acestui graf, care să fie complet și să aibă un număr maxim de noduri. (6p.)
3. Pentru o sală de spectacole se memorează următoarele informații specifice: numărul de locuri disponibile și prețul biletului pentru un loc. Variabilele i și s sunt întregi, iar variabila t memorează informațiile specifice pentru fiecare dintre cele 15 săli de spectacole dintr-un oraș.
- Scrieți o secvență de instrucțiuni în urma executării căreia variabila s să memoreze suma totală obținută în urma vânzării biletelor pentru un festival de teatru care s-ar desfășura simultan în toate sălile, iar acestea ar fi ocupate complet. (6p.)

```

citește n (număr natural)
repetă
    x ← n % 100
    dacă x > 1 atunci
        p ← (x % 2) * (x % 3) * (x % 5) * (x % 7)
        dacă p ≠ 0 atunci
            scrie x, ' '
        n ← [n / 10]
    până când n < 10

```

```

struct sala
{
    int nrLocuri;
    int pret;
} t[15];

```

SUBIECTUL al III-lea**(30 de puncte)**

1. Subprogramul **patrate** are trei parametri:
- n , prin care primește un număr natural ($n \in [2, 10^9]$);
 - x și y , prin care furnizează câte un număr natural cu proprietatea că $x^2 \cdot y^2 = n$ și $2 \leq x < y$ sau valoarea 0, prin fiecare dintre acestea, dacă nu există două astfel de numere. Dacă sunt mai multe astfel de valori, se furnizează cele corespunzătoare unei valori minime a lui x .
- Scrieți definiția completă a subprogramului.
- Exemplu: pentru $n=400$, după apel, $x=2$ și $y=10$, iar pentru $n=16$ sau $n=24$, după apel, $x=0$ și $y=0$. (10p.)
2. Suprafața unei scene a fost împărțită în zone dispuse ca elementele unui tablou bidimensional cu număr egal de linii și coloane. Fiecare instrument are alocată o zonă și este identificat printr-un număr natural. Dirijorul hotărăște eliminarea tuturor instrumentelor plasate pe diagonala principală a tabloului și rearanjarea celor păstrate, prin deplasarea unora dintre acestea cu câte o poziție spre stânga, astfel încât să nu existe zone intermediare libere, ca în exemplu.
- Scrieți un program C/C++ care citește de la tastatură numere naturale din intervalul $[2, 10^2]$, și anume n și elementele unui tablou bidimensional cu n linii și n coloane, reprezentând numerele de identificare ale instrumentelor, în ordinea plasării lor inițiale pe scenă. Programul modifică apoi tabloul în memorie corespunzător hotărârii dirijorului și afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.
- Exemplu: pentru $n=4$ și tabloul
- | | | | |
|----|----|----|----|
| 20 | 30 | 80 | 52 |
| 71 | 93 | 64 | 51 |
| 62 | 81 | 55 | 49 |
| 90 | 65 | 32 | 22 |
- se obține tabloul
- | | | | |
|----|----|----|--|
| 30 | 80 | 52 | |
| 71 | 64 | 51 | |
| 62 | 81 | 49 | |
| 90 | 65 | 32 | |
- (10p.)
3. Numim **secvență progresivă** a unui șir crescător de numere naturale un subșir al acestuia, format din termeni aflați pe poziții consecutive în șirul dat, cu proprietatea că fiecare termen apare în subșir de un număr de ori egal cu valoarea sa. Lungimea secvenței este egală cu numărul de termeni ai acesteia.
- Fișierul **bac.txt** conține un șir crescător de cel mult 10^6 numere naturale din intervalul $[1, 10^6]$, astfel încât orice termen al șirului apare de un număr de ori cel mult egal cu valoarea sa. Numerele sunt separate prin câte un spațiu.
- Se cere să se afișeze pe ecran lungimea maximă a unei secvențe progresive din șirul aflat în fișier. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al memoriei utilizate.
- Exemplu: dacă fișierul conține numerele 1 2 2 3 4 4 4 4 6 6 6 6 6 6 7 7 7 8 8 8 8 8 8 8 atunci pe ecran se afișează valoarea 10.
- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)
- b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 1 – Sesiunea iunie-iulie 2022

SUBIECTUL I

(20 de puncte)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Indicați valoarea expresiei C/C++ alăturate. 22/20*20/2
 - a. 0
 - b. 0.55
 - c. 10
 - d. 55
2. Variabilele `x` și `valoare` sunt declarate alăturat. Indicați o instrucțiune de atribuire corectă din punctul de vedere sintactic.

- a. `x= ("apa minerala", 10, 2.5);`
 - b. `x.produs.pret=2.5;`
 - c. `x.denumire[5]= "Borsec";`
 - d. `valoare=2*x.cantitate*x.pret;`

```
struct produs
{ char denumire[50];
  float cantitate, pret;
}x;
float valoare;
```
3. La o sală de sport se organizează antrenamente pentru sporturi din mulțimea {tenis, fotbal, volei, handbal, baschet}, astfel încât o persoană poate opta pentru un pachet de antrenamente pentru două sau trei sporturi, dar nu poate alege fotbal și baschet în același pachet. Utilizând metoda backtracking se generează toate posibilitățile unei persoane de a opta pentru un pachet de antrenamente în cadrul sălii de sport. Două pachete sunt distincte dacă diferă prin cel puțin un sport ales. Primele cinci soluții generate sunt, în această ordine: {tenis, fotbal}, {tenis, fotbal, volei}, {tenis, fotbal, handbal}, {tenis, volei}, {tenis, volei, handbal}. Indicați soluția generată imediat după {fotbal, handbal}.
 - a. {volei, handbal}
 - b. {fotbal, handbal, baschet}
 - c. {handbal, baschet}
 - d. {volei, handbal, baschet}
4. Un graf neorientat cu 6 noduri, numerotate de la 1 la 6, are muchiile [1, 2], [1, 3], [2, 3], [4, 6]. Indicați numărul de componente conexe ale acestuia.
 - a. 1
 - b. 3
 - c. 5
 - d. 6
5. Într-un arbore cu rădăcină un nod se află pe nivelul `n` dacă lanțul elementar care are o extremitate în nodul respectiv și cealaltă extremitate în rădăcina arborelui are lungimea `n`. Pe nivelul 0 se află un singur nod (rădăcina). Un arbore cu rădăcină are 12 noduri, numerotate de la 1 la 12, și este reprezentat prin vectorul de „tați” (2, 4, 2, 0, 4, 4, x, 6, x, x, y, y). Indicați un set de valori pe care le pot avea `x` și `y`, știind că pe ultimul nivel există cinci noduri frunză și că un singur nod este numerotat cu o valoare egală cu numărul său de descendenți direcți („fii”) de tip „frunză”.
 - a. `x=11 y=12`
 - b. `x=8 y=3`
 - c. `x=3 y=8`
 - d. `x=3 y=3`

SUBIECTUL al II-lea**(40 de puncte)**

1. Algoritmul alăturat este reprezentat în pseudocod.
S-a notat cu $a \div b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $\lfloor c \rfloor$ partea întreagă a numărului real c .
- a. Scrieți ce se afișează în urma executării algoritmului dacă se citește numărul 56. (6p.)
- b. Scrieți două numere din intervalul $[10, 99]$ care pot fi citite, astfel încât, pentru fiecare dintre acestea, în urma executării algoritmului, să se afișeze valoarea 1. (6p.)
- c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
- d. Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind adecvat a doua structură **cât timp...execută** cu o structură repetitivă cu test final. (6p.)
2. Subprogramul f este definit alăturat. Scrieți două valori din intervalul $[2000, 2025]$ pentru variabila întreagă x , astfel încât, pentru fiecare dintre acestea, $f(2022, x)$ să aibă valoarea 2022. (6p.)
3. Variabilele s și id permit accesul la câte un șir de maximum 50 de caractere, șirul accesat prin id fiind inițial vid, iar cel accesat prin s memorând, în această ordine, separate printr-un spațiu, prenumele și numele unei persoane, fiecare fiind format numai din litere ale alfabetului englez. Scrieți o secvență de instrucțiuni C/C++ astfel încât, în urma executării acesteia, șirul accesat prin id să memoreze numele persoanei menționate, urmat de 2022. Declarați corespunzător eventualele alte variabile utilizate.
Exemplu: dacă șirul accesat prin variabila s este **Ana Popescu** atunci șirul accesat prin variabila id este **Popescu2022** (6p.)

```

citește n
    (număr natural)
i ← 2; k ← 0
cât timp n ≥ i execută
    cât timp n % i = 0 execută
        k ← k + 1
    n ← n / i
    dacă i = 2 atunci i ← i + 1
    altfel i ← i + 2
scrie k

int f(int a, int b)
{ if (a <= b) return a;
  return f(a-1, b+1);
}

```

SUBIECTUL al III-lea**(30 de puncte)**

1. Subprogramul **secventa** are un singur parametru, n , prin care primește un număr natural ($n \in [10, 10^9]$) în care nu există secvențe de mai mult de două cifre identice aflate pe poziții consecutive. Subprogramul înlocuiește în n fiecare secvență 22 cu câte o secvență 20 și furnizează, prin același parametru, numărul obținut. Dacă nu se înlocuiește nicio secvență, subprogramul furnizează numărul nemodificat. Scrieți definiția completă a subprogramului.
Exemplu: dacă $n = 202233228$, după apel $n = 202033208$. (10p.)
2. Pentru prevenirea eroziunii la Marea Neagră, autoritățile au hotărât înnisiparea unei plaje. Plaja a fost împărțită în parcele de formă pătrată cu latura de 1 metru, dispuse ca elementele unui tablou bidimensional, reținându-se nivelul fiecărei parcele, măsurat în metri, de la nivelul mării. Innisiparea se realizează doar pe acele parcele care au nivelul strict mai mic decât al celei mai joase dintre parcelele vecine cu ea, adăugând nisip, astfel încât cele două să aibă același nivel. Două parcele sunt vecine dacă au o latură comună. Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, m și n , apoi elementele unui tablou bidimensional cu m linii și n coloane, numere naturale din intervalul $[1, 10]$, reprezentând nivelurile parcelelor, în ordinea dispunerii acestora pe plajă.
Programul afișează pe ecran cantitatea totală de nisip adăugată, măsurată în metri cubi.
Exemplu: pentru $m=5$, $n=4$ și tabloul alăturat, se vor innisipa parcelele marcate, pentru a fi aduse la nivelul parcelelor vecine corespunzătoare, și se va afișa valoarea 7 ($1+3+1+2$). (10p.)
- | | | | |
|---|---|---|---|
| 5 | 3 | 4 | 6 |
| 7 | 5 | 9 | 3 |
| 6 | 8 | 3 | 9 |
| 4 | 5 | 2 | 3 |
| 3 | 1 | 4 | 4 |
3. Fișierul **bac.txt** conține numere naturale din intervalul $[1, 10^9]$, astfel: pe prima linie două numere, x și y ($x < y$), iar pe a doua linie un șir de cel mult 10^6 numere, ordonate crescător. Numerele aflate pe aceeași linie a fișierului sunt separate prin câte un spațiu. Se cere să se afișeze pe ecran numărul de valori distincte din șirul aflat pe a doua linie a fișierului care aparțin intervalului $[x, y]$. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al memoriei utilizate.
Exemplu: dacă fișierul are conținutul alăturat, pe ecran se afișează 6
- | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|
| 2 | 9 | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 1 | 2 | 2 | 3 | 5 | 5 | 5 | 5 | 6 | 6 | 7 | 8 | 10 | 10 | 12 | 15 | 21 | 21 |
- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)
- b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 5 – Sesiunea august 2022

SUBIECTUL I

(20 de puncte)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

- Variabila x este de tip întreg. Indicați o expresie care are valoarea 1 dacă și numai dacă expresia C/C++ alăturată are valoarea 1.

a. $!(x < 3 \ \&\& \ x >= 10)$	b. $!(x < 3 \ \ x >= 10)$
c. $x < 3 \ \&\& \ !(x >= 10)$	d. $!(x < 3) \ \ x >= 10$
- Subprogramul f este definit incomplet, alăturat. Indicați o expresie cu care pot fi înlocuite punctele de suspensie pentru ca $f(n, 3)$ să aibă valoarea 1 pentru orice număr prim n ($n \in [2, 10^4]$) și valoarea 0 în caz contrar.

```

int f(int x, int y)
{ if(x!=2 && x%2==0) return 0;
  if(y*y>x) return 1;
  if(x%y==0) return 0;
  return f(x,.....);
}

```

a. $y+2$	b. $y-2$	c. $y*2$	d. $y/2$
----------	----------	----------	----------
- Utilizând metoda backtracking, se generează toate posibilitățile de a forma platouri de câte trei fructe distincte din mulțimea {măr, gutuie, prună, caisă, piersică}, astfel încât pe același platou nu se află o gutuie și o piersică simultan. Două platouri sunt distincte dacă ele conțin cel puțin un fruct diferit. Primele patru soluții generate sunt, în această ordine, (măr, gutuie, prună), (măr, gutuie, caisă), (măr, prună, caisă), (măr, prună, piersică). Indicați soluția generată imediat după (gutuiie, prună, caisă).

a. (caisă, piersică, măr)	b. (gutuiie, prună, piersică)
c. (mar, caisă, piersică)	d. (prună, caisă, piersică)
- Variabila j este de tip întreg, iar variabila A memorează un tablou bidimensional cu 100 de linii și 100 de coloane, numerotate de la 0 la 99. Indicați valoarea variabilei j , dacă elementul $A[20][j]$ se află pe diagonala secundară a tabloului.

a. 20	b. 49	c. 79	d. 80
-------	-------	-------	-------
- Un graf neorientat are 7 noduri, numerotate de la 1 la 7, și 8 muchii, dintre care șase sunt: [1, 2], [2, 4], [2, 7], [3, 4], [4, 5], [4, 6]. Știind că unul dintre lanțurile elementare care au lungimea maximă este 1, 2, 7, 5, 4, 6, indicați care ar putea fi celelalte două muchii ale grafului.

a. [1, 4] și [4, 7]	b. [1, 4] și [5, 7]	c. [2, 5] și [4, 7]	d. [3, 6] și [5, 7]
---------------------	---------------------	---------------------	---------------------

SUBIECTUL al II-lea**(40 de puncte)****1. Algoritmul alăturat este reprezentat în pseudocod.**

a. Scrieți ce se afișează în urma executării algoritmului dacă se citește, în această ordine, valorile 21 și 47. (6p.)

b. Dacă pentru variabila **m** se citește numărul 5, scrieți două valori care pot fi citite pentru variabila **n** astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, valoarea afișată să fie 10. (6p.)

c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)

d. Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind adecvat prima structură **cât timp...execută** cu o structură repetitivă cu test final. (6p.)

```

citește m, n
  (numere naturale nenule, m < n)
x ← 1
cât timp x = 1 execută
  x ← m; y ← n; n ← n - 1
  cât timp x ≠ y execută
    dacă x > y atunci x ← x - y
    altfel y ← y - x
  ■
  ■
  ■
scrie n + 1

```

2. Un arbore cu 9 noduri, numerotate de la 1 la 9, are muchiile [1, 8], [2, 3], [2, 5], [2, 7], [4, 5], [5, 6], [5, 8], [8, 9]. Scrieți două noduri care pot fi alese drept rădăcină, astfel încât nodul 5 să aibă doi "frați". (6p.)

3. Variabila **x**, declarată alăturat, memorează simultan un an calendaristic și temperatura medie a aerului, măsurată în grade Celsius, înregistrată în București în anul respectiv.

```

struct meteo
{
  int an, temperatura;
} x;

```

Scrieți o secvență de instrucțiuni în urma executării căreia să se afișeze pe ecran mesajul **CALDUROS** dacă temperatura menționată a fost strict mai mare decât 11°C, mesajul **RACOROS** dacă temperatura menționată a fost strict mai mică decât 10°C, sau mesajul **NORMAL** în orice alt caz. (6p.)

SUBIECTUL al III-lea**(30 de puncte)****1. Subprogramul *schimb* are trei parametri:**

- **n** și **x**, prin care primește câte un număr natural ($n \in [0, 10^8]$, $x \in [1, 9]$);
- **p**, prin care primește un număr natural reprezentând poziția unei cifre a numărului **n** ($0 \leq p$). Pozițiile cifrelor sunt numerotate de la dreapta la stânga, astfel: cifra unităților este pe poziția 0, cifra zecilor este pe poziția 1 ș.a.m.d.

Subprogramul transformă numărul **n**, înlocuind cifra de pe poziția **p** cu cifra **x**, și furnizează numărul obținut tot prin parametrul **n**. Scrieți definiția completă a subprogramului.

Exemplu: dacă **n=12587**, **x=6** și **p=3**, după apel, **n=16587**.

(10p.)

2. Se consideră o vocală oarecare a alfabetului englez, notată cu **v**, și o consoană oarecare a alfabetului englez, notată cu **c**. Litera **v** se numește **vocală prietenă** a lui **c** dacă în șirul literelor alfabetului englez, ordonat lexicografic, **v** îl precede pe **c**, iar între **v** și **c** nu există nicio vocală. Se consideră vocale literele **a, e, i, o, u**.

Exemplu: **e** este vocală prietenă pentru consoanele **f, g** și **h**, dar nu este vocală prietenă pentru consoanele **d** și **j**.

Un elev vrea să transmită unui prieten o parolă, codificată. Parola este formată dintr-un singur cuvânt de cel mult 50 de caractere, litere mici ale alfabetului englez, cel puțin una fiind consoană. Codificarea se face prin înlocuirea fiecărei consoane cu vocala sa prietenă, ca în exemplu.

Scrieți un program C/C++ care citește de la tastatură un cuvânt, reprezentând o parolă de tipul precizat și determină, în memorie, forma codificată a acesteia. Programul afișează pe ecran parola codificată obținută.

Exemplu: pentru parola rame se afișează oaie, iar pentru parola sport se afișează ooooo (10p.)

3. Numim **secvență paritară** a unui șir de numere naturale un subșir al acestuia, format din termeni cu aceeași paritate, aflați pe poziții consecutive în șirul dat. Lungimea secvenței este egală cu numărul de termeni ai acesteia.

Fișierul **bac.txt** conține un șir de cel puțin două și cel mult 10^6 numere naturale din intervalul $[0, 10^9]$. Numerele sunt separate prin câte un spațiu, iar în șir există cel puțin doi termeni cu aceeași paritate pe poziții consecutive.

Se cere să se afișeze pe ecran numărul secvențelor paritare de lungime maximă din șirul aflat în fișier, precum și această lungime maximă. Numerele afișate sunt separate printr-un spațiu. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al memoriei utilizate.

Exemplu: dacă fișierul conține numerele

2 3 5 1 7 9 8 4 4 11 15 17 21 11 6 11 15 17 21 11 6 5 2 6 4 0 16

atunci pe ecran se afișează valorile 4 5

a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)

b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta - Simulare 2023

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

- Indicați numerele pe care le pot memora variabilele întregi x și y , astfel încât valoarea expresiei $C/C++$ alăturate să fie 23. $x/2+y*3$
 - $x=8$ și $y=23$
 - $x=20$ și $y=39$
 - $x=23$ și $y=66$
 - $x=43$ și $y=20$
- Subprogramul f este definit alăturat. Indicați valoarea lui $f(2023)$.


```
int f(int n)
{ if(n==0) return 0;
  if(n%10==2) return f(n/10)*10+3;
  return f(n/10)*10+2;
}
```

- 2303
 - 3232
 - 3332
 - 5355
- Expresiile alăturate au ca valori trei numere naturale reprezentând, în această ordine, următoarele informații memorate pentru o motocicletă: anul fabricației și dimensiuni specifice (garda la sol și lungimea). Indicați o declarație corespunzătoare a variabilei m .


```
m.an
m.dm.garda
m.dm.lungime
```

- ```
struct
{ int an;
 struct{int garda, lungime;}dm;
}m;
```
  - ```
struct
{ int m.an;
  struct{int garda, lungime;}m.dm;
};
```
 - ```
struct
{ int an, dm.garda, dm.lungime;
}m;
```
  - ```
struct m
{ int an, dm (garda,lungime);
};
```
- Utilizând metoda backtracking, se generează toate pachetele formate din câte 3 tipuri distincte de ceai din mulțimea {matcha, mate, moringa, oolong, tulsi}. Două amestecuri sunt distincte dacă diferă prin cel puțin un tip de ceai. Primele patru soluții obținute sunt, în această ordine: (matcha, mate, moringa), (matcha, mate, oolong), (matcha, mate, tulsi) și (matcha, moringa, oolong). Indicați succesiunea care NU se obține, prin această metodă, în ordinea dată.
 - (matcha, moringa, tulsi)
 - (moringa, oolong, tulsi)
 - (mate, moringa, oolong)
 - (oolong, tulsi, mate)
- Într-un magazin sunt 8 raioane, distribuite în trei zone importante, în fiecare zonă fiind un număr par, nenul, de raioane. În scopul fluidificării deplasării clienților se marchează unele culoare, astfel încât fiecare culoar să conecteze două raioane, iar deplasarea pe el să se facă într-un singur sens. Se realizează o hartă, sub forma unui graf orientat, în care vârfurile reprezintă raioanele din magazin, iar arcele reprezintă culoarele marcate. Indicați numărul maxim de culoare care se pot marca, astfel încât graful să aibă trei componente tare conexe, fiecare componentă fiind reprezentarea pe hartă a câte uneia dintre cele trei zone importante din magazin.
 - 20
 - 28
 - 36
 - 56

SUBIECTUL al II-lea

(40 de puncte)

1. Algoritmul alăturat este reprezentat în pseudocod.

S-a notat cu $a \div b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $[c]$ partea întreagă a numărului real c .

a. Scrieți valoarea afișată dacă se citește numărul 5174. (6p.)

b. Scrieți trei numere impare, cu cifre distincte, din intervalul $[10^2, 10^4)$ care pot fi citite astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, să se afișeze valoarea 34. (6p.)

c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)

d. Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind prima structură **repetă...până când** cu o structură de tip **pentru...execută**. (6p.)

```

citește n
  (număr natural)
x ← 5
repetă
  cn ← n; n ← 0; p ← 1
  repetă
    c ← cn ÷ 10
    dacă c = x atunci c ← 5 - c
    ■
  n ← c * p + n; cn ← [cn / 10]; p ← p * 10
până când cn = 0
x ← x - 1
până când x = 0
scrie n

```

2. Un arbore cu 7 noduri, numerotate de la 1 la 7, este reprezentat prin vectorul de „tați” (4, 1, 1, 0, 7, 4, 4). Scrieți trei muchii care i se pot adăuga, astfel încât graful obținut să fie eulerian. (6p.)

3. Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 5 linii și 5 coloane, numerotate de la 1 la 5, având inițial toate elementele nule.

Fără a utiliza alte variabile decât cele menționate, scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila a să memoreze tabloul alăturat.

```

for (i=1; i<=5; i++)
  for (j=1; j<=5; j++)
    .....

```

```

4 3 2 1 0
4 3 2 1 1
4 3 2 2 2
4 3 3 3 3
4 4 4 4 4

```

(6p.)

SUBIECTUL al III-lea

(30 de puncte)

1. Subprogramul **NrImp** are trei parametri:

- x și y , prin care primește câte un număr natural ($2 \leq x < y \leq 10^9$)
- nr , prin care furnizează numărul valorilor naturale din intervalul $[x, y]$ cu trei divizori pozitivi impari.

Scrieți definiția completă a subprogramului.

Exemplu: dacă $x=4$ și $y=50$, după apel $nr=6$ (pentru valorile 9, 18, 25, 36, 49, 50).

(10p.)

2. Într-un text de cel mult 100 de caractere cuvintele sunt separate prin câte un spațiu și sunt formate din litere mari ale alfabetului englez, iar dacă sunt scrise prescurtat sunt urmate de caracterul . (punct). Textul reprezintă denumirea științifică a unei păsări și doar cuvintele din mulțimea {FAMILIA, GENUL, SPECIA}, specifice sistemului de clasificare a organismelor, sunt mereu prescurtate, prin eliminarea ultimelor lor litere.

Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat și construiește în memorie, apoi afișează pe ecran denumirea științifică, în care pentru cuvintele specifice sistemului de clasificare a organismelor se păstrează doar primele trei litere, scrise cu litere mici, și urmate de punct, ca în exemplu.

Exemplu: pentru textul FAMIL. PHASIANIDAE GEN. MELEAGRIS SP. GALLOPAVO

sau pentru textul FAM. PHASIANIDAE G. MELEAGRIS SPECI. GALLOPAVO

se obține fam. PHASIANIDAE gen. MELEAGRIS spe. GALLOPAVO

(10p.)

3. Pentru a studia un metal, s-a urmărit comportamentul său într-o succesiune de pași, la fiecare pas metalul fiind supus unei anumite temperaturi. Pașii sunt numerotați cu valori naturale consecutive, începând de la 1. Un pas se numește reprezentativ dacă la niciunul dintre pașii anteriori nu este utilizată o temperatură strict mai mare decât la acest pas. Dacă există o secvență de pași consecutivi la care se utilizează aceeași temperatură, se consideră reprezentativ doar primul pas din secvență.

Fișierul `bac.txt` conține cel mult 10^6 numere naturale din intervalul $[0, 10^4]$, separate prin câte un spațiu, reprezentând temperaturile la care este supus metalul, în ordinea pașilor corespunzători. Se cere să se afișeze pe ecran, separați prin câte un spațiu, pașii reprezentativi pentru datele din fișier. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al spațiului de memorie utilizat.

Exemplu: dacă fișierul conține numerele 7 4 9 10 10 10 3 9 2 10 10 8 2 30

se afișează pe ecran 1 3 4 10 14

a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)

b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 6 – Sesiunea specială 2023

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Variabilele întregi m și n memorează câte un număr natural. Indicați expresia C/C++ care are valoarea 1 dacă și numai dacă fiecare dintre numerele memorate în variabilele m și n este par.

a. $(m+n)\%2==0$ b. $(m-n)\%2==0$ c. $m\%2+n\%2==0$ d. $m\%2-n\%2==0$

2. Subprogramul f este definit alăturat. Indicați valoarea $f(35, 3)$.

```
int f(int n, int k)
{ if(n==0 || k==0) return 0;
  return n*k+10*f(n/k, k);
}
```

a. 1022 b. 355 c. 102 d. 33

3. Variabila p , declarată alăturat, memorează dimensiunile unui paralelipiped dreptunghic: lungimea, lățimea, respectiv înălțimea.

```
struct paralelipiped
{ float lg, lt, h;
}p;
```

Volumul paralelipipedului este egal cu produsul celor trei dimensiuni ale sale. Indicați o instrucțiune prin care i se atribuie variabilei reale v valoarea volumului paralelipipedului corespunzător variabilei p .

a. $v=lg*lt*h;$ b. $v=lg.p*lt.p*h.p;$ c. $v=p(lg*lt*h);$ d. $v=p.lg*p.lt*p.h;$

4. Utilizând metoda backtracking se generează toate șirurile de 5 litere distincte din mulțimea $\{A, M, U, R, G\}$, astfel încât în fiecare șir litera R precede literele A și M . Primele cinci soluții generate sunt, în această ordine: URAMG, URAGM, URMAG, URMGA, URGAM. Indicați soluția generată imediat după GURMA.

a. GARUM b. GRAMU c. GRAUM d. GRUAM

5. Într-o zonă sunt 23 de obiective turistice, numerotate de la 1 la 23; zona este reprezentată ca un graf orientat în care vârfurile corespund acestor obiective, iar arcele sunt stabilite în funcție de modalitatea de deplasare între două obiective, astfel: de la fiecare obiectiv numerotat cu i ($i \in [1, 23]$) se poate merge direct doar la obiectivul numerotat cu $2*i$, dacă acesta există, sau doar la obiectivul numerotat cu $i-1$, în caz contrar. Indicați numărul maxim de vârfuri ale unui drum elementar în acest graf, cu extremitatea inițială 1.

a. 5 b. 11 c. 16 d. 21

SUBIECTUL al II-lea**(40 de puncte)**

1. Algoritmul alăturat este reprezentat în pseudocod.

- a. Scrieți ce se afișează în urma executării algoritmului, dacă se citește numărul 3. (6p.)

- b. Scrieți două numere care pot fi citite, astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, în total, numărul de cifre 2 afișate să fie în intervalul $[20, 50]$. (6p.)

- c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)

- d. Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind una dintre structurile **pentru...execută** cu o structură de tip **cât timp...execută**. (6p.)

```
citește n
(număr natural nenul)
pentru i ← 1, n execută
  pentru j ← 1, n execută
    dacă j ≤ i atunci
      scrie 2
    altfel
      scrie 3
```

2. Un arbore cu 8 noduri, numerotate de la 1 la 8, este reprezentat prin vectorul de „tați” (0, 1, 1, 2, 2, 4, 5, 1). Enumerați toți descendenții nodului numerotat cu 2. (6p.)
3. Un șir de caractere **a** este numit **prefix** al unui șir de caractere **b** dacă este identic cu **b** sau dacă **b** se obține din **a** prin adăugarea la dreapta a unor alte caractere. Variabila **k** este de tip întreg, iar variabila **s** permite accesul la un șir de cel mult 20 de caractere. Scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, să se afișeze pe ecran, în ordinea descrescătoare a lungimii, separate prin câte un spațiu, toate prefixele șirului accesat prin variabila **s**, fiecare încheindu-se cu prima literă a șirului **s**, ca în exemplu. Declarați eventualele alte variabile utilizate. Exemplu: pentru șirul **elemente** se afișează: **elemente eleme ele e**

```
for(k=strlen(s)-1; k>=0; k--)
{ .....
```

 (6p.)

SUBIECTUL al III-lea**(30 de puncte)**

1. Subprogramul **Putere** are trei parametri:
- **n**, prin care primește un număr natural ($n \in [2, 10^9]$);
 - **x** și **p**, prin care furnizează două numere naturale cu proprietatea că $n = x^p$, iar **x** este cel mai mic număr cu această proprietate.
- Scrieți definiția completă a subprogramului.
Exemplu: dacă **n=16** atunci, după apel, **x=2** și **p=4**, dacă **n=216** atunci, după apel, **x=6** și **p=3**, iar dacă **n=12** atunci, după apel, **x=12** și **p=1**. (10p.)
2. La o expoziție sunt prezentate mai multe tipuri de flori, fiecare având un anumit cod. Zona de expunere este organizată pe **n** rânduri, fiecare având câte **n** parcele de dimensiune egală. Pe primul rând sunt expuse **n** tipuri de flori distincte, fiecare parcelă conținând flori de un anumit tip. Celelalte parcele sunt ocupate astfel:
- toate parcelele aflate în dreapta diagonalei secundare a zonei de expunere sunt ocupate de gazon;
 - fiecare parcelă, aflată pe diagonala secundară a zonei de expunere sau în stânga acestei diagonale, conține flori cu un cod de cel mult două cifre, obținut din ultimele două cifre ale sumei codurilor florilor aflate în două parcele de pe rândul anterior, și anume cea de pe aceeași coloană, respectiv cea de pe coloana din dreapta sa; dacă prima dintre cifrele codului obținut este 0, ea este ignorată, ca în exemplu.
- Scrieți un program C/C++ care citește de la tastatură un număr natural, **n** ($n \in [2, 20]$), apoi **n** numere naturale distincte din intervalul $[0, 10^2]$, reprezentând, în această ordine, de la stânga la dreapta, codurile florilor din parcelele aflate pe primul rând al zonei de expunere; gazonul are codul -1. Programul construiește în memorie un tablou bidimensional cu **n** linii și **n** coloane, în care fiecare linie corespunde unui rând de parcele și fiecare element al liniei corespunde unei parcele, în ordinea acestora, și memorează codul plantelor (flori sau gazon) plasate în acea parcelă.
- Programul afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.
- | | | | | |
|----|----|----|----|----|
| 90 | 10 | 4 | 27 | 30 |
| 0 | 14 | 31 | 57 | -1 |
| 14 | 45 | 88 | -1 | -1 |
| 59 | 33 | -1 | -1 | -1 |
| 92 | -1 | -1 | -1 | -1 |
- Exemplu:** dacă se citesc numerele **n=5**, apoi 90, 10, 4, 27, 30 se obține tabloul alăturat. (10p.)
3. Intervalul **[x, y]** se numește **p-interval** pentru un șir de valori întregi, dacă oricare dintre primii **p** termeni ai șirului aparține intervalului, iar numărul de valori întregi distincte din interval este minim.
- Exemplu:** pentru șirul 2, 7, -1, 8, 3, 10 există [2, 2] ca 1-interval, [-1, 8] ca 4-interval și 5-interval etc. Fișierul **bac.in** conține un șir de cel mult 10^6 numere întregi din intervalul $[-10^9, 10^9]$, separate prin câte un spațiu.
- Se cere să se afișeze pe ecran cea mai mică și cea mai mare valoare a lui **p** ($p \geq 2$) cu proprietatea că **(p-1)-interval** este identic cu **p-interval** pentru șirul aflat în fișier. Valorile afișate pot fi egale, iar dacă nu există nicio astfel de valoare, pe ecran se afișează mesajul **nu exista**. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al memoriei utilizate.
- Exemplu:** dacă fișierul conține numerele 2 7 1 8 3 10 6 -3 -2 13, se afișează pe ecran 5 9 (intervale conform cerinței se obțin pentru valorile 5, 7 și 9 ale lui **p**), iar dacă fișierul conține numerele 2 7 1 0 8 10 -3 13, se afișează pe ecran **nu exista**
- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)
- b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 5 – Sesiunea iunie-iulie 2023

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

- Indicați expresia C/C++ care are cea mai mare valoare, comparativ cu celelalte trei expresii.
 - $20 \cdot 23 / (2 \cdot 2)$
 - $20 / 2 \cdot 23 / 2$
 - $(20 \cdot 23) / 2$
 - $(20 \cdot 23) / 2 \cdot 2$
- Subprogramul `f` este definit alăturat. Indicați ce se afișează în urma apelului de mai jos.
`f(23);`

```
void f (int n)
{ if (n!=0) f(n/2);
  cout<<n%2; | printf("%d", n%2);
}
```

 - 100111
 - 111010
 - 010111
 - 01251123
- Variabila `k` este de tip întreg, iar variabila `s` permite memorarea unui șir de maximum 50 de caractere. Indicați valoarea variabilei `k` în urma executării secvenței alăturate.


```
strcpy (s, "bac2023");
s[s[2]-'a'] = '\0';
k=strlen(s);
```

 - 7
 - 6
 - 2
 - 1
- Indicați un vector de „tați” corespunzător unui arbore cu 7 noduri, în care cel puțin unul dintre noduri are trei ascendenți.
 - 0, 1, 2, 1, 1, 1, 2
 - 3, 0, 2, 1, 3, 2, 1
 - 4, 3, 0, 3, 4, 4, 3
 - 5, 4, 3, 0, 2, 3, 4
- O companie de colectare a fructelor are 6 depozite, numerotate de la 1 la 6: depozitele 1, 3 și 5 conțin mere, depozitele 2 și 4 conțin pere, iar depozitul 6 conține piersici. Compania a construit 4 benzi de transport unidirecțional: de la depozitul 1 la depozitul 5, de la depozitul 5 la depozitul 2, de la depozitul 5 la depozitul 4 și de la depozitul 6 la depozitul 1. Dacă depozitele reprezintă vârfurile unui graf orientat, iar benzile de transport reprezintă arcele acestuia, indicați numărul minim de benzi de transport care pot fi adăugate, astfel încât graful obținut să aibă trei componente tare conexe, fiecare dintre acestea având vârfuri care corespund depozitelor cu același tip de fructe.
 - 4
 - 3
 - 2
 - 1

SUBIECTUL al II-lea**(40 de puncte)****1. Algoritmul alăturat este reprezentat în pseudocod.**

S-a notat cu $a \% b$ restul împărțirii numărului natural a la numărul natural nenul b , și cu $[c]$ partea întreagă a numărului real c .

- Scrieți valoarea afișată dacă se citește numărul 6907512. (6p.)
- Scrieți două valori distincte din intervalul $[100, 999]$ care pot fi citite astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, să se afișeze o valoare identică cu cea citită. (6p.)
- Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
- Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind structura `cât timp...execută` cu o structură repetitivă cu test final. (6p.)

```

citește x (număr natural)
p ← 1; m ← -1
cât timp p ≤ x execută
  c ← [x/p] % 10
  dacă c > m atunci
    m ← c; p ← p * 10
  altfel
    x ← [x / (p * 10)] * p + x % p
■
dacă m ≥ 0 atunci scrie x
altfel scrie "nul"
■

```

2. Utilizând metoda backtracking, se generează toate amestecurile de apă provenită din surse distincte din mulțimea {lac, mare, ocean, ploaie, râu}, astfel încât o sursă să fie de apă sărată și una sau două surse să fie de apă dulce. Marea și oceanul sunt surse de apă sărată, iar lacul, ploaia și râul sunt surse de apă dulce. Două amestecuri sunt distincte dacă diferă prin cel puțin o sursă a apei. Primele patru soluții obținute sunt, în această ordine: (lac, mare), (lac, mare, ploaie), (lac, mare, râu) și (lac, ocean). Scrieți soluția generată imediat înainte și soluția generată imediat după (ocean, ploaie). (6p.)
3. Variabila **f** memorează, pentru fiecare dintre cele 10 soiuri de lalele care se vând într-o florărie, caracteristicile acestora: denumirea (șir de maximum 20 de caractere) și stocul, exprimat prin numărul de fire și prețul unui fir, în lei (numere naturale). Știind că expresiile de mai jos au ca valori denumirea primului soi de lalele, respectiv suma, în lei, necesară pentru a cumpăra toate lalelele din acest soi, scrieți în limbajul C/C++ definiția unei structuri cu eticheta **lalea**, care să permită memorarea informațiilor menționate pentru un soi de lalele, și declarați corespunzător variabila **f**.
f[0].denumire f[0].stoc.nrFire*f[0].stoc.pretFir (6p.)

SUBIECTUL al III-lea**(30 de puncte)**

1. Un număr natural nenul, **n**, se numește număr **abundent** dacă $s(n)/n > s(k)/k$, pentru orice număr natural nenul **k** ($k \leq n-1$), unde s-a notat cu **s(i)** suma divizorilor pozitivi ai numărului natural nenul **i**. Subprogramul **abundent** are un singur parametru, **n**, prin care primește un număr natural ($n \in [2, 10^6]$). Subprogramul returnează valoarea 1, dacă **n** este un număr abundent, sau valoarea 0, în caz contrar. Scrieți definiția completă a subprogramului.
Exemplu: pentru **n=6**, subprogramul returnează valoarea 1 ($s(6)/6=2$, iar cel mai mare raport obținut pentru valori strict mai mici decât 6 este $s(4)/4=1.75$), iar pentru **n=7** sau **n=8**, subprogramul returnează valoarea 0 ($s(7)/7=1.14$, $s(8)/8=1.87$). (10p.)
2. Pentru a identifica punctele în care se concentrează apa în albia unui râu în cazul secetei, se determină talvegul acesteia – linia care unește punctele cele mai adânci ale albiei. În acest scop s-au stabilit **ns** secțiuni transversale pe cursul apei, numerotate începând de la 1, și în cadrul fiecărei secțiuni s-a măsurat adâncimea apei în **np** puncte, numerotate începând de la 1. Din fiecare secțiune, în ordine, se include în talveg cel mai adânc punct al acesteia, iar dacă în secțiune sunt mai multe puncte aflate la aceeași adâncime, maximă, se va lua în considerare doar primul dintre ele, ca în exemplu. Scrieți un program C/C++ care citește de la tastatură două numere naturale, **ns** și **np** ($ns \in [1, 10^3]$, $np \in [1, 50]$), și cele **ns · np** elemente ale unui tablou bidimensional, valori naturale din intervalul $[0, 10^2]$. Fiecare linie a tabloului corespunde câte unei secțiuni, în ordinea numerotării acestora, iar valorile memorate pe linie reprezintă adâncimile celor **np** puncte stabilite pentru acea secțiune, în ordinea numerotării lor. Programul afișează pe ecran, pentru fiecare secțiune, o pereche formată din numărul de ordine al secțiunii și numărul de ordine al punctului său care s-a inclus în talveg. Numerele din fiecare pereche sunt afișate separate prin câte un caracter : (două puncte), iar fiecare pereche este urmată de un spațiu.
Exemplu: pentru **ns=6**, **np=4** și tabloul alăturat, se afișează pe ecran valorile:
 1:3 2:2 3:2 4:2 5:4 6:3 (10p.)
- | | | | |
|---|---|---|---|
| 2 | 4 | 5 | 3 |
| 2 | 6 | 6 | 3 |
| 1 | 5 | 2 | 5 |
| 1 | 3 | 3 | 3 |
| 3 | 4 | 3 | 5 |
| 0 | 1 | 2 | 1 |
3. Un număr natural **x** este numit prefix al unui număr natural **y** dacă se obține din acesta prin eliminarea a cel puțin unei cifre de la dreapta sa, și este numit sufix al lui **y** dacă se obține din acesta prin eliminarea a cel puțin unei cifre de la stânga sa.
Exemplu: 15 este prefix pentru 154 sau 1521, este sufix pentru 3415 sau 5115, dar nu este nici prefix, nici sufix pentru 15.
 Fișierul **bac.txt** conține maximum 10^6 numere naturale din intervalul $[10, 10^4]$, separate prin câte un spațiu. Se cere să se afișeze pe ecran numărul valorilor de două cifre care apar de același număr de ori ca sufix, respectiv ca prefix al numerelor din șirul aflat în fișier. Proiectați un algoritm eficient din punctul de vedere al timpului de executare.
Exemplu: dacă fișierul are conținutul
342 1684 2134 5434 111 98 98 3405 3412 7016 8634 1010, 102 310,
 se afișează pe ecran: 4 (pentru valorile 10, 11, 16, 34).
 a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)
 b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 7 – Sesiunea august 2023

SUBIECTUL I

(20 de puncte)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

- Indicați numerele pe care le pot memora variabilele întregi x și y , astfel încât valoarea expresiei C/C++ alăturată să fie 1.

$y \% x - (x/y) * 3 != 0$

 - $x=25$ și $y=75$
 - $x=25$ și $y=15$
 - $x=15$ și $y=0$
 - $x=10$ și $y=30$
- Subprogramul f este definit alăturat. Indicați valoarea $f(3, 2)$.


```
int f(int x, int y)
{
    int z;
    if (y--0) return 1;
    z=f(x, y/2);
    if (y%2!=0) return z*z*x;
    return z*z;
}
```

 - 1
 - 2
 - 9
 - 18
- Variabila s poate accesa un șir cu cel mult 20 de caractere. Indicați șirul accesat prin s în urma executării secvenței alăturate.


```
strcpy(s, "2019");
strcpy(s+3, "23");
```

 - 20123
 - 201923
 - 202223
 - 2023
- Utilizând metoda backtracking, se generează toate parfumurile formate prin amestecarea a câte 3 esențe distincte din mulțimea {bergamotă, cireș, iris, lămâie, salcâm}. Două parfumuri sunt distincte dacă diferă prin cel puțin o esență. Primele patru soluții obținute sunt, în această ordine: (bergamotă, cireș, iris), (bergamotă, cireș, lămâie), (bergamotă, cireș, salcâm) și (bergamotă, iris, lămâie). Indicați penultima soluție generată.
 - (cireș, iris, salcâm)
 - (cireș, lămâie, salcâm)
 - (lămâie, iris, salcâm)
 - (iris, lămâie, salcâm)
- Un graf neorientat are 5 noduri, numerotate de la 1 la 5, cu gradele figurate în tabelul alăturat. Indicați o pereche de valori posibile pentru x și y .

Nod	1	2	3	4	5
Grad	2	x	3	3	y

 - 0 și 4
 - 1 și 5
 - 2 și 3
 - 3 și 3

SUBIECTUL al II-lea**(40 de puncte)****1. Algoritmul alăturat este reprezentat în pseudocod.**S-a notat cu **[c]** partea întreagă a numărului real **c**.

- Scrieți valoarea afișată dacă se citește, în această ordine, numerele 3, 746, 82, 3067, 67, 78, 178. (6p.)
- Dacă pentru **n** se citește numărul 2, scrieți un șir de numere naturale din intervalul **[0, 9]** care pot fi citite în continuare, în acea ordine, astfel încât, în urma executării algoritmului, să se afișeze valoarea 0. (6p.)
- Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
- Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind prima structură **cât timp...execută** cu o structură repetitivă de tip **pentru...execută**. (6p.)

```

citește n (număr natural nenul)
x ← 0
cât timp n > 0 execută
    citește a, b (numere naturale)
    c ← a; p ← 1
    cât timp a > 9 execută
        a ← [a/10]; p ← p*10
    ■
    a ← a*p+b
    dacă a ≠ c atunci
        x ← x+1
    ■
    n ← n-1
    ■
scrie x

```

- Într-un arbore cu rădăcină un nod se află pe nivelul **x** dacă lanțul elementar care are o extremitate în nodul respectiv și cealaltă extremitate în rădăcina arborelui are lungimea **x**. Pe nivelul 0 se află un singur nod (rădăcina).
Un arbore cu 6 noduri, numerotate de la 1 la 6, are muchiile [1, 2], [2, 3], [2, 6], [3, 4], [3, 5].
Scrieți nodurile care pot fi desemnate ca rădăcină, astfel încât fiecare dintre arborii obținuți să aibă un număr minim de niveluri. (6p.)
- Variabila **p**, declarată alăturat, memorează caracteristicile unui produs: denumire și preț. Scrieți o secvență de instrucțiuni C/C++ prin care variabila **a** memorează valoarea primei litere a denumirii produsului respectiv, dacă acesta are prețul strict mai mic decât 100, sau caracterul * în caz contrar. (6p.)

```

struct produs
{
    char denumire[20];
    int pret;
}p;
char a;

```

SUBIECTUL al III-lea**(30 de puncte)**

- Subprogramul **DNP1** are un singur parametru, **n**, prin care primește un număr natural ($n \in [1, 10^9]$), și afișează pe ecran, separați prin câte un spațiu, toți divizorii pozitivi impari ai lui **n** care **NU** sunt primi. Scrieți definiția completă a subprogramului.

Exemplu: dacă **n=90**, se afișează pe ecran, nu neapărat în această ordine, numerele

1 9 15 45

(10p.)

- Un joc folosește o tablă dreptunghiulară, pe care sunt reprezentate celule de dimensiune egală, dispuse pe **m** linii și pe **n** coloane. În fiecare celulă este înscris un număr natural.
Numim **pătrat de valoare p** patru celule ale tablei, situate pe două linii consecutive și pe două coloane consecutive, cu proprietatea că suma valorilor înscrise în aceste celule este egală cu **p**. Scopul jocului este determinarea unui pătrat de valoare maximă.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul **[2, 20]**, **m** și **n**, apoi elementele unui tablou bidimensional cu **m** linii și **n** coloane, numere naturale din intervalul **[0, 10⁴]**, reprezentând numerele înscrise pe tabla de joc, în ordinea dispunerii celulelor corespunzătoare. Programul determină un pătrat de valoare maximă al tablei și afișează pe ecran această valoare.

1	1	2	30
3	10	1	2
1	13	12	1
1	2	3	1
7	1	20	1

Exemplu: pentru **m=5**, **n=4** și tabloul alăturat, se afișează pe ecran valoarea 36, corespunzătoare pătratului evidențiat în figură. **(10p.)**

- Fișierul **date.in** conține pe prima linie două numere naturale din intervalul **[1, 10⁶]**, **m** și **n**, iar pe următoarele două linii numere naturale din intervalul **[0, 10²]**: pe a doua linie un șir **A**, de **m** numere, iar pe a treia linie un șir **B**, de **n** numere. Numerele aflate pe aceeași linie sunt separate prin câte un spațiu. Se cere să se afișeze pe ecran numărul maxim de perechi de forma **(pa, pb)** ($pa \in [1, m]$, $pb \in [1, n]$), cu proprietatea că termenul de pe poziția **pa** din șirul **A** are aceeași valoare cu termenul de pe poziția **pb** din șirul **B** și că fiecare poziție, corespunzătoare șirului **A**, respectiv șirului **B**, apare în cel mult o pereche, ca în exemplu. Proiectați un algoritm eficient din punctul de vedere al timpului de executare.

Exemplu: dacă fișierul conține numerele alăturate, se afișează pe ecran 6 (de exemplu, pentru perechile (1, 1), (2, 9), (4, 2), (5, 5), (6, 6), (7, 7) sau pentru perechile (1, 2), (2, 9), (4, 1), (5, 7), (6, 8), (8, 5)).

8	9
1	0
4	1
5	3
5	5
5	5
3	0

- Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)

- Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta – Simulare 2024

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Indicați valoarea expresiei C/C++ alăturate.

6.3/20+24

- a. 0 b. 24 c. 24.315 d. 24.9

2. Subprogramul `f` este definit alăturat. Indicați ce se afișează în urma apelului de mai jos.
`f(2024);`

```
void f(int x)
{ if(x>=24) if(x%2==0)
    { cout<<2; | printf("2");
      f(x/2);
    }
    else
    { f(x/2);
      cout<<4; | printf("4");
    }
    else cout<<0; | printf("0");
}
```

- a. 04424222 b. 22220444 c. 22242440 d. 44402222

3. Utilizând metoda backtracking se generează toate variantele prin care patru persoane pot plăti o consumație totală de 24 de lei în următoarele condiții:

- fiecare plătește o sumă nenulă, strict mai mică decât 10;
- primul plătește mai puțin decât al doilea, al doilea mai puțin decât al treilea, iar acesta mai puțin decât al patrulea.

O soluție constă în patru valori, reprezentând, în ordine, sumele plătite de fiecare dintre cele patru persoane. Primele patru soluții generate sunt, în această ordine: (1, 6, 8, 9), (2, 5, 8, 9), (2, 6, 7, 9), (3, 4, 8, 9). Indicați soluția generată imediat după (3, 6, 7, 8).

- a. (3, 5, 7, 9) b. (4, 5, 6, 9) c. (4, 5, 7, 8) d. (5, 5, 6, 8)

4. În declararea de mai jos, câmpurile structurii memorează date referitoare la un joc video: categoria, numărul de niveluri de dificultate (maximum cinci) și numărul maxim de puncte care pot fi obținute la fiecare nivel.

Indicați expresia C/C++ care are valoarea 1 dacă și numai dacă numărul maxim de puncte care se pot obține pe nivelul notat cu 2 este strict mai mare decât 4.

```
struct joc
{ int categorie, nrNiveluri;
  int puncte[5];
}g;
```

- a. `joc(puncte[2])>4` b. `puncte.joc[2]>4` c. `puncte[2].g>4` d. `g.puncte[2]>4`

5. Într-un oraș sunt 5 parcuri, notate cu A, B, C, D și E, care sunt conectate prin piste pentru biciclete, pe care se circulă într-un singur sens, astfel: o pistă de la parcul A către parcul B, o pistă de la parcul B către parcul E, o pistă de la parcul C către parcul D, o pistă de la parcul D către parcul C, o pistă de la parcul E către parcul A, o pistă de la parcul E către parcul C și o pistă de la parcul E către parcul D. Indicați numărul minim de piste pentru care trebuie schimbat sensul de circulație, astfel încât graful orientat obținut, în care parcurile reprezintă vârfurile, iar pisteles reprezintă arcele, să fie tare conex.

- a. 1 b. 2 c. 4 d. 6

SUBIECTUL al II-lea

(40 de puncte)

1. Algoritmul alăturat este reprezentat în pseudocod.

S-a notat cu $a \div b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $[c]$ partea întreagă a numărului real c .

a. Scrieți valoarea afișată în urma executării algoritmului dacă se citește numărul 32625. (6p.)

b. Scrieți două numere din intervalul $[10, 100]$ care pot fi citite pentru variabila n , astfel încât, pentru fiecare dintre acestea, în urma executării algoritmului, să se afișeze valoarea 2. (6p.)

c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)

d. Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind adecvat a doua structură **cât timp...execută** cu o structură repetitivă cu test final. (6p.)

```

citește n
  (număr natural nenul)
s ← 0; d ← 2
cât timp d * d ≤ n execută
  p ← 0
  cât timp n % d ≠ 0 execută
    n ← [n/d]; p ← 1
  s ← s + d * p; d ← d + 1
dacă n ≠ 1 atunci
  s ← s + n
scrie s

```

2. Un arbore cu 10 noduri, numerotate de la 1 la 10, este reprezentat prin vectorul de „tați” (8, 1, 4, 5, 0, 9, 4, 5, 4, 3). Enumerați toate nodurile arborelui care au câte un singur descendent direct („fiu”). (6p.)

3. Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 5 linii și 5 coloane, numerotate începând de la 0, având inițial toate elementele egale cu 1.

Scrieți secvența de instrucțiuni C/C++ de mai jos, înlocuind punctele de suspensie cu instrucțiuni adecvate, dintre care cel mult patru de atribuire, astfel încât, în urma executării secvenței obținute, variabila a să memoreze tabloul alăturat.

```

for (i=0; i<5; i++)
  for (j=0; j<5; j++)

```

.....

```

0 0 0 0 0
0 2 2 2 2
0 2 4 4 4
0 2 4 6 6
0 2 4 6 8

```

(6p.)

SUBIECTUL al III-lea

(30 de puncte)

1. Subprogramul **Impare** are un parametru, n , prin care primește un număr natural ($n \in [1, 10^4]$). Subprogramul returnează numărul obținut prin duplicarea fiecărei cifre impare a lui n sau -1 dacă acesta nu are nicio cifră impară. Scrieți definiția completă a subprogramului.

Exemplu: dacă $n=3361$, subprogramul returnează 3333611.

(10p.)

2. Un **șablon** este un text în care cuvintele sunt separate prin câte un spațiu și sunt formate fie numai din litere mici și mari ale alfabetului englez, fie numai din caractere *, în ultimul caz numindu-se **cuvinte generice**. Lungimea unui cuvânt este egală cu numărul de caractere care îl compun.

Un computer generează o frază pe baza unui astfel de șablon, prin înlocuirea fiecărui cuvânt generic cu unul dintre cuvintele de aceeași lungime, preluat dintr-o listă dată.

Scrieți un program C/C++ care citește de la tastatură un număr natural, n ($n \in [1, 100]$), și o listă de n cuvinte, urmată de un șablon de tipul precizat. Fiecare cuvânt din listă este format din maximum 10 litere mici și mari ale alfabetului englez și la citire este introdus singur pe linie. Șablonul conține maximum 100 de caractere. Programul obține în memorie și apoi afișează pe ecran una dintre frazele care pot fi generate pe baza șablonului și a listei citite sau mesajul **imposibil** dacă nu se poate genera o astfel de frază.

Exemplu: dacă $n=6$, iar lista de cuvinte este cea alăturată,

```

pentru șablonul      Era o vreme ***** si ***** din belsug *****
se generează fraza   Era o vreme placuta si soare din belsug soare
sau fraza            Era o vreme calduta si soare din belsug acasa
etc., iar pentru șablonul *** o vreme ***** si *****
se afișează mesajul imposibil

```

```

rece
placuta
acasa
calduta
innorata
soare

```

(10p.)

3. Un șir se numește de tip **api** dacă numărul de apariții ale fiecărui termen este mai mic sau egal cu acel termen și are o paritate egală cu a acestuia.

Fișierul **bac.in** conține un șir de cel mult 10^6 numere naturale din intervalul $[1, 10^3]$, separate prin câte un spațiu. Se cere să se afișeze pe ecran mesajul **DA**, dacă șirul este de tip **api**, sau mesajul **NU** în caz contrar. Proiectați un algoritm eficient din punctul de vedere al timpului de executare.

Exemplu: dacă fișierul conține numerele 6 27 2 6 27 6 6 14 14 2 27 se afișează pe ecran **DA**

(termenul par 6 apare de 4 ori, 4 fiind tot număr par și $4 \leq 6$, termenii pari 2 și 14 apar de câte 2 ori, 2 fiind tot număr par și $2 \leq 2$, respectiv $2 \leq 14$, iar termenul impar 27 apare de 3 ori, 3 fiind tot număr impar și $3 \leq 27$).

a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)

b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 4 – Sesiunea specială 2024

SUBIECTUL I

(20 de puncte)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Variabilele x și y sunt de tip real. Indicați o expresie C/C++ echivalentă cu cea alăturată.

a. $x <= -25 \text{ \&\& } y >= 6$	b. $x < -25 \text{ } y >= 6$
c. $x < -25 \text{ \&\& } y < 6$	d. $x < -25 \text{ } y < 6$

 2. Subprogramul f este definit alăturat. Indicați ce se afișează în urma apelului de mai jos.

```
void f(int x)
{ if(x%5!=0)
  { x=x*5/2; f(x); }
  else x=x-1;
  cout<<x; | printf("%d",x);
}
```

a. 225554	b. 545522	c. 55109	d. 92255
-----------	-----------	----------	----------

 3. Utilizând metoda backtracking, se generează, respectând ordinea enumerării elementelor din mulțimile precizate mai jos, toate numerele de mașină care cuprind câte trei elemente constitutive, separate prin cratimă:
 - indicativul județului, din mulțimea {B, BR, HD, MM, SV, TL};
 - un număr, format din două cifre din mulțimea {2, 4, 6, 8}, în ordine strict crescătoare;
 - trei litere mari distincte din mulțimea {A, B, C}, cea din mijloc fiind A.

Primele șapte numere generate sunt, în această ordine: B-24-BAC, B-24-CAB, B-26-BAC, B-26-CAB, B-28-BAC, B-28-CAB, B-46-BAC.

Indicați două soluții, prima generată imediat înainte de soluția SV-68-CAB, iar a doua generată imediat după soluția SV-68-CAB.

a. MM-68-CAB, SV-86-BAC	b. SV-46-CAB, TL-24-BAC
c. SV-48-BAC, SV-68-BAC	d. SV-68-BAC, TL-24-BAC

 4. Variabila t , declarată alăturat, memorează, pentru un telefon, următoarele date: tipul sistemului de operare (litera A pentru Android și litera W pentru Windows), prețul produsului, precum și data achiziționării (ziua, luna și anul). Indicați o expresie C/C++ prin care se accesează anul achiziționării telefonului.


```
struct data
{ int zi, luna, an;
};
struct telefon
{ char sistem;
  float pret;
  struct data achizitionare;
} t;
```

a. telefon.an	b. t.data.an
c. telefon.achizitionare.data.an	d. t.achizitionare.an

 5. Oricare dintre cele 5 vârfuri ale unui graf orientat are proprietatea că suma dintre gradul său extern și gradul său intern este 4. Indicați lungimea maximă a unui drum elementar în acest graf.

a. 10	b. 8	c. 4	d. 3
-------	------	------	------

SUBIECTUL al II-lea**(40 de puncte)****1. Algoritmul alăturat este reprezentat în pseudocod.**

S-a notat cu $a \div b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $[c]$ partea întreagă a numărului real c .

- Scrieți ce se afișează dacă se citește numărul 10. (6p.)
- Scrieți două valori distincte care pot fi citite astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, numărul de valori egale cu 1 afișate să fie 4. (6p.)
- Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
- Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind adecvat prima structură repetitivă **cât timp...execută** cu o structură de tip **pentru...execută**. (6p.)

```

citește n
(număr natural,  $n \geq 2$ )
i ← 1
cât timp i ≤ n execută
    m ← i
    cât timp  $m \div 2 = 0$  execută
        m ←  $[m/2]$ 
    dacă  $m \neq i$  atunci
        scrie m, ' '
    i ← i + 1

```

- Un graf neorientat cu 6 noduri, numerotate de la 1 la 6, are muchiile [1,2], [2,3], [2,4], [2,5], [4,5], [4,6], [5,6]. Scrieți listele de adiacență ale unui graf parțial al său care să fie conex și fără cicluri. (6p.)
- Variabilele c și i sunt de tip întreg, iar variabila s permite memorarea unui șir de cel mult 20 de caractere. Se citesc de la tastatură 10 cuvinte, formate din litere mici ale alfabetului englez și separate prin Enter. Scrieți secvența de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila c să memoreze valoarea 1 dacă există printre cuvintele citite cel puțin unul format din două litere și care să conțină o vocală și o consoană, sau valoarea 0 altfel. Se consideră vocale literele a, e, i, o, u.

Exemplu: dacă se citesc cuvintele alăturate, variabila c are valoarea 1.

```

c = .....;
for(i=1; i<=10; i++) {   cin>>s;   |   scanf("%s", s);
                        .....
}

```

```

daca
au
plecat
el
nu
primeste
si
inghetata
de
fragi

```

(6p.)

SUBIECTUL al III-lea**(30 de puncte)**

- La un laborator sunt studiate aglomerările de fulgi de nea formate din câte **nouă** cristale de **patru** tipuri diferite date (notate cu 1, 2, 3 sau 4), astfel încât din fiecare tip să existe cel puțin câte un cristal. O astfel de aglomerare de fulgi a fost reprezentată printr-un număr natural, în care fiecare cifră reprezintă tipul unui cristal. Subprogramul **fulg** are un parametru, n , prin care primește un număr natural ($n \in [0, 10^9]$). Subprogramul returnează valoarea 1, dacă prin n este reprezentată o aglomerare de fulgi de nea dintre cele studiate, sau 0 în caz contrar. Scrieți în C/C++ definiția completă a subprogramului.

Exemplu: dacă $n=112243413$ subprogramul returnează 1, iar dacă $n=12314$ sau $n=112253513$ sau $n=112243457$ sau $n=111122223$, subprogramul returnează 0. (10p.)

- Fiind date două numere naturale a și b , numim valoare **generată** de a și b un număr obținut din a prin alipirea la stânga sau la dreapta sa a cifrelor lui b , în ordinea în care apar în acesta.

Exemplu: dacă $a=123$ și $b=45$, se pot obține două numere generate de acestea: 12345 și 45123.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 50]$, m și n , și construiește în memorie un tablou bidimensional cu m linii, numerotate de la 1 la m , și n coloane, numerotate de la 1 la n , în care fiecare element este egal cu cea mai mică valoare generată de numerele de ordine ale liniei, respectiv coloanei pe care se află.

Programul afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele fiecărei linii fiind separate prin câte un spațiu.

Exemplu: pentru $m=5$ și $n=4$ se obține tabloul alăturat.

(10p.)

11	12	13	14
12	22	23	24
13	23	33	34
14	24	34	44
15	25	35	45

- Fișierul **numere.in** conține un șir de cel mult 10^6 numere naturale din intervalul $[0, 99]$. Numerele din fișier sunt separate prin câte un spațiu.

Se cere să se determine primul și ultimul număr din șir care conțin cea mai mare cifră ce apare în scrierea numerelor din fișier. Numerele determinate se afișează pe ecran, în ordinea apariției lor în șir, separate printr-un spațiu. Dacă nu există două astfel de numere pe poziții distincte, se afișează pe ecran mesajul **nu exista**. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al spațiului de memorie utilizat.

Exemplu: dacă fișierul conține numerele 34 5 38 30 87 70 11 8 82 25 se afișează pe ecran 38 82, dacă fișierul conține numerele 34 5 38 30 87 70 11 8 38 25 se afișează pe ecran 38 38, iar dacă fișierul conține numerele 34 5 38 30 se afișează pe ecran **nu exista**.

- Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)

- Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 3 – Sesiunea iunie-iulie 2024

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Indicați expresia C/C++ care are valoarea 1 dacă și numai dacă numerele memorate în variabilele întregi x și y sunt pare.

a. $x\%2==0 \ \&\& \ (y+1)\%2!=0$ b. $(x-y)\%2==0$ c. $(x+y)\%2==0$ d. $x\%2==y\%2$

2. Subprogramul f este definit alăturat. Indicați ce se afișează în urma apelului de mai jos.
 $f(2020, 0);$

```
void f(int x, int y)
{ if (x<10) cout<<x; | printf("%d",x);
  else
  { f(x/10,y+1);
    cout<<x%10; | printf("%d",x%10);
  }
  cout<<y; | printf("%d",y);
}
```

a. 23020

b. 2022100

c. 02023210

d. 23022100

3. Utilizând metoda backtracking se generează toate permutările elementelor mulțimii ordonate astfel: $\{1, 2, 3, 4, 5, 6\}$; pentru fiecare permutare, pe primele trei poziții sunt doar valori pare, iar pe ultimele trei poziții sunt doar valori impare. Primele șase permutări generate sunt, în această ordine: $(2, 4, 6, 1, 3, 5)$, $(2, 4, 6, 1, 5, 3)$, $(2, 4, 6, 3, 1, 5)$, $(2, 4, 6, 3, 5, 1)$, $(2, 4, 6, 5, 1, 3)$, $(2, 4, 6, 5, 3, 1)$. Indicați a șaptea permutare generată.

a. $(4, 2, 6, 1, 5, 3)$ b. $(4, 2, 6, 1, 3, 5)$ c. $(2, 6, 4, 1, 3, 5)$ d. $(2, 4, 6, 5, 3, 2)$

4. Variabila x memorează, pentru fiecare dintre cele 20 de sortimente de ciocolată, următoarele date: tipul (litera N pentru ciocolată neagră și litera L pentru ciocolată cu lapte) și prețul produsului. Indicați o expresie a cărei valoare este egală cu tipul celui de al 11-lea sortiment de ciocolată.

```
struct ciocolata
{ char tip;
  float pret;
}x[20];
```

a. $x.ciocolata[10].tip$ b. $x.tip[10]$ c. $x[10].ciocolata.tip$ d. $x[10].tip$

5. Într-un graf neorientat, cu 10 muchii, două noduri au gradul 0, șase noduri au grade impare, iar celelalte noduri au grade pare, nenule. Indicați numărul maxim de noduri ale grafului.

a. 17

b. 15

c. 12

d. 10

SUBIECTUL al II-lea**(40 de puncte)****1. Algoritmul alăturat este reprezentat în pseudocod.**S-a notat cu **[c]** partea întreagă a numărului real **c**.

- a. Scrieți valoarea afișată dacă se citește, în această ordine, numerele 5, 15, 27, 10, 1, 17. (6p.)
- b. Dacă pentru **n** se citește valoarea 2, scrieți un set de numere distincte din intervalul $[0, 10^3]$ care pot fi citite în continuare, astfel încât, în urma executării algoritmului, să se afișeze valoarea 4. (6p.)
- c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
- d. Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind adecvat structura **pentru...execută** cu o structură repetitivă de tip **cât timp...execută**. (6p.)

```

citește n
    (număr natural nenul)
p ← 1
pentru i ← 1, n execută
    citește x
    (număr natural)
    repetă
        x ← [x/3]
    până când x ≤ 3
    dacă x ≠ 0 atunci
        p ← p * x
scrie p

```

2. Un arbore cu 8 noduri, numerotate de la 1 la 8, este reprezentat prin vectorul de „tați”: (3,0,2,5,2,5,1,5). Enumerați, în ordinea parcurgerii lor, nodurile celui mai lung lanț elementar care are extremitatea inițială în rădăcină. (6p.)

3. Variabilele **i** și **j** sunt de tip întreg, iar variabila **a** memorează un tablou bidimensional cu 9 linii și 9 coloane, numerotate începând de la 0, având inițial toate elementele nule.

Scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie cu instrucțiuni adecvate, dintre care cel mult patru de atribuire, astfel încât, în urma executării secvenței obținute, variabila **a** să memoreze tabloul alăturat.

```

for (i=0; i<9; i++)
    for (j=0; j<9; j++)

```

.....

(6p.)

4	4	4	4	2	2	2	2	2
4	4	4	2	2	2	2	2	2
4	4	2	2	2	2	2	2	2
4	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	4
2	2	2	2	2	2	2	4	4
2	2	2	2	2	2	4	4	4
2	2	2	2	2	4	4	4	4

SUBIECTUL al III-lea**(30 de puncte)**

1. Un număr natural se numește **major impar** dacă suma divizorilor săi proprii impari este strict mai mare decât suma divizorilor săi proprii pari. Divizorii proprii ai unui număr sunt divizorii săi naturali diferiți de 1 și de el însuși.

Exemplu: 18 este număr major impar (divizorii săi proprii pari sunt 2, 6, cei impari 3, 9, iar $3+9 > 2+6$).

Subprogramul **majImp** are doi parametri, **a** și **b**, prin care primește câte un număr natural ($2 \leq a \leq b \leq 10^4$).

Subprogramul returnează cel mai mic număr major impar din intervalul **[a, b]**, sau valoarea 0, dacă în interval nu există un astfel de număr. Scrieți în C/C++ definiția completă a subprogramului.

Exemplu: dacă **a=16**, **b=30**, atunci subprogramul returnează 18. (10p.)

2. Într-un text, de cel mult 100 de caractere, cuvintele sunt formate din litere ale alfabetului englez și sunt separate prin câte un spațiu. Textul are cel puțin două cuvinte.

Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat mai sus și afișează pe ecran mesajul **DA** și un număr natural **n**, separate printr-un spațiu, dacă toate cuvintele din text au câte **n** litere, sau mesajul **NU** în cazul în care nu toate cuvintele au același număr de litere.

Exemplu: dacă textul citit este **Ana are cel mai bun mar** se afișează pe ecran **DA 3**

iar dacă textul citit este **Ana are cel mai dulce mar** se afișează pe ecran **NU** (10p.)

3. De-a lungul unui traseu montan este utilizată o succesiune de marcaje turistice, care trebuie urmate în acea ordine. Pentru fiecare marcaj se cunoaște cota (înălțimea, măsurată în metri) la care este plasat. Numim **scară** într-un traseu o secvență de marcaje aflate pe poziții consecutive în cadrul traseului, care au drept cote numere consecutive, ordonate strict crescător. O scară este formată din cel puțin două marcaje, iar lungimea acesteia este egală cu numărul de marcaje care o compun.

Fișierul **bac.txt** conține un șir de cel mult 10^6 numere naturale din intervalul $[10, 10^4]$, separate prin câte un spațiu, reprezentând cotele marcajelor turistice din cadrul unui traseu, în ordinea în care se succed în acesta. Se cere să se afișeze pe ecran, separate prin câte un spațiu, în ordine strict crescătoare, cotele corespunzătoare marcajelor unei scări de lungime maximă pe acest traseu. Dacă în cadrul traseului există mai multe astfel de scări, se afișează doar cotele corespunzătoare marcajelor uneia dintre ele, iar dacă nu există nicio scară, pe ecran se afișează mesajul **nu exista**. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al spațiului de memorie utilizat.

Exemplu: dacă fișierul conține numerele 500 600 601 405 569 570 700 701 625 626 627 520 atunci pe ecran se afișează 625 626 627

- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)

- b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 8 – Sesiunea august 2024

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Variabilele x și y memorează câte un număr natural ($x \leq y$). Indicați expresia C/C++ cu valoarea 1 dacă și numai dacă intervalul $[x, y]$ **NU** conține niciun număr de două cifre.
 - a. $!(x >= 10 \mid \mid y > 99)$
 - b. $!(x < -99 \ \&\& \ y >= -10)$
 - c. $x >= 10 \mid \mid y <= 99$
 - d. $x > 99 \ \&\& \ y < 10$
2. Subprogramul f este definit alăturat. Indicați un apel în urma căruia se afișează pe ecran secvența:
13579


```
void f(int x, int y, int z)
{ if(z < x)
  { cout << y; | printf("%d", y);
    f(x, y+2, z+1);
  }
}
```

- a. $f(5, 0, 0);$
 - b. $f(5, 1, 1);$
 - c. $f(6, 1, 6);$
 - d. $f(6, 1, 1);$
3. Variabilele s , x și y permit memorarea câte unui șir de cel mult 15 caractere. Indicați șirul accesat prin variabila s în urma executării secvenței alăturate.


```
strcpy(x, "soare");
strcpy(y, "ploaie");
if (strcmp(x, y) > 0) strcpy(s, x+1);
else strcpy(s, y+2);
```

- a. oare
 - b. oaie
 - c. ploaie
 - d. soare
4. Un arbore cu 14 noduri, numerotate de la 1 la 14, este reprezentat prin vectorul de "tați" (13, 3, 0, 6, 13, 3, 3, 7, 6, 2, 13, 2, 6, 13). Indicați nodul care este rădăcina arborelui.
 - a. 1
 - b. 3
 - c. 7
 - d. 13
5. Indicați numărul maxim de vârfuri izolate pe care le poate avea un graf orientat cu 24 de vârfuri și 24 de arce poziționate adecvat.
 - a. 5
 - b. 6
 - c. 18
 - d. 21

SUBIECTUL al II-lea**(40 de puncte)**

1. Algoritmul alăturat este reprezentat în pseudocod.
S-a notat cu $a \% b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $[c]$ partea întreagă a numărului real c .
 - a. Scrieți valoarea afișată în urma executării algoritmului dacă se citește numărul 412531. (6p.)
 - b. Scrieți două numere din intervalul $[1, 11110]$ care pot fi citite pentru variabila n , astfel încât, pentru fiecare dintre acestea, în urma executării algoritmului, să se afișeze valoarea 11111. (6p.)
 - c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
 - d. Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind adecvat structura **cât timp...execută** cu o structură repetitivă cu test final. (6p.)
- ```

citește n
 (număr natural nenul)
x ← -1; y ← -1
cât timp n > 9 execută
 dacă x = -1 atunci x ← n % 100
 altfel y ← n % 100
 n ← [n/10]
dacă x < y atunci n ← (n*100+x)*100+y
altfel n ← (n*100+y)*100+x
scrie n

```

2. Utilizând metoda backtracking, se generează toate buchetele formate din câte trei flori distincte din mulțimea ordonată astfel: {trandafir, crin, gerbera, iris, eustoma, orhidee, zambilă}; în același buchet crinul nu poate fi împreună cu eustoma sau cu zambila. Două buchete diferă prin cel puțin o floare. Primele cinci soluții sunt, în ordinea generării: (trandafir, crin, gerbera), (trandafir, crin, iris), (trandafir, crin, orhidee), (trandafir, gerbera, iris), (trandafir, gerbera, eustoma). Scrieți două soluții, una generată imediat înainte, iar cealaltă generată imediat după (crin, gerbera, orhidee). (6p.)
3. Gama muzicală conține, în această ordine, notele **do, re, mi, fa, sol, la, si**. Două note formează un interval muzical, iar acesta este numit **terță** dacă, în ordinea din gamă, între cele două note există o singură altă notă muzicală. În notația englezească se folosesc unele litere mari ale alfabetului pentru notele muzicale, astfel: litera **A** pentru nota **la**, litera **B** pentru nota **si**, apoi, în ordine, litera **C** pentru nota **do**, litera **D** pentru nota **re**, litera **E** pentru nota **mi**, litera **F** pentru nota **fa** și litera **G** pentru nota **sol**. Variabila **m**, declarată alături, memorează literele corespunzătoare notației englezești pentru două note care formează un interval muzical, date în ordinea din gamă. Știind că în câmpul **nota1** se memorează una dintre literele **C, D, E, F** sau **G**, scrieți o secvență de instrucțiuni C/C++ în urma executării căreia să se afișeze pe ecran mesajul **TERTA**, dacă intervalul respectiv formează o terță, sau mesajul **NU**, în caz contrar.
- ```
struct interval
{
    char nota1;
    char nota2;
}m;
```
- Exemplu:** dacă în câmpul **nota1** se memorează litera **G** (pentru nota **sol**), iar în câmpul **nota2** se memorează litera **B** (pentru nota **si**), se afișează mesajul **TERTA**. (6p.)

SUBIECTUL al III-lea**(30 de puncte)**

1. Un număr natural nenul, **n**, se numește **moderat** dacă este egal cu produsul a două numere prime, iar acestea sunt consecutive în șirul numerelor prime (2, 3, 5, 7, 11, 13, 17...). Subprogramul **moderat** are un singur parametru, **n**, prin care primește un număr natural ($n \in [1, 10^5]$). Subprogramul returnează valoarea 1, dacă **n** este un număr moderat, sau valoarea 0, în caz contrar. Scrieți definiția completă a subprogramului. **Exemplu:** dacă **n=35**, subprogramul returnează 1 ($35=5 \cdot 7$), iar dacă **n=18** sau **n=55** sau **n=4**, subprogramul returnează 0. (10p.)
2. O grădină este împărțită în parcele pătrate identice, dispuse pe rânduri succesive. Toate parcelele de pe un rând conțin același tip de plante (doar flori sau doar gazon) și oricare două rânduri alăturate conțin tipuri diferite de plante, **pe primul rând fiind gazon**. Pentru fiecare parcelă se memorează înălțimea acesteia, egală cu înălțimea medie a plantelor conținute, exprimată în decimetri. Pentru a pune în valoare florile, fiecare parcelă cu flori trebuie să fie mai înaltă cu cel puțin un decimetru decât oricare dintre parcelele cu gazon de pe rândul vecin anterior. S-a hotărât tunderea parcelelor cu gazon prea înalte, astfel încât ele să fie aduse la înălțimea maximă permisă. Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, **m** și **n**, apoi elementele unui tablou bidimensional cu **m** linii și **n** coloane, numere naturale din intervalul $[2, 10]$, reprezentând înălțimile parcelelor, în ordinea dispunerii acestora în grădină, rând după rând, și pe fiecare rând de la stânga la dreapta. Programul modifică apoi tabloul în memorie corespunzător situației terenului după tunderea gazonului și afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.
- Exemplu:** pentru **m=4**, **n=6** și tabloul
- | | | | | | |
|---|---|---|---|---|---|
| 6 | 2 | 3 | 7 | 9 | 4 |
| 7 | 8 | 5 | 9 | 8 | 5 |
| 8 | 7 | 2 | 6 | 2 | 9 |
| 7 | 2 | 5 | 9 | 3 | 8 |
- se obține tabloul
- | | | | | | |
|---|---|---|---|---|---|
| 4 | 2 | 3 | 4 | 4 | 4 |
| 7 | 8 | 5 | 9 | 8 | 5 |
| 1 | 1 | 1 | 1 | 1 | 1 |
| 7 | 2 | 5 | 9 | 3 | 8 |
- (10p.)
3. Șirul 0, 0, 1, 4, 13, 38, 105, 280, 729 ... este definit astfel: $f_1=f_2=0$, $f_3=1$, $f_n=4 \cdot f_{n-1}-3 \cdot f_{n-2}-2 \cdot f_{n-3}$ (unde **n** este un număr natural $n \geq 4$). Se citesc de la tastatură trei numere naturale **x**, **y** și **z** ($x \leq y < z \leq 10^9$), valorile a trei termeni aflați pe poziții consecutive în șirul dat, și se cere să se scrie în fișierul **bac.txt**, în ordine descrescătoare, separați prin câte un spațiu, toți termenii șirului care sunt mai mici sau egali cu **z**. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al spațiului de memorie utilizat. **Exemplu:** dacă se citesc numerele 38 105 280 fișierul conține numerele 280 105 38 13 4 1 0 0
- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)
- b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta – Simulare 2025

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

- Indicați valoarea expresiei C/C++ alăturate. 20.25/2.5*100
 a. 0 b. 0.081 c. 810 d. 1000
- Subprogramul `f` este definit alăturat. Indicați apelul în urma căruia se obține valoarea 2025.

```
int f(int x)
{
    if(x>500) return 5;
    else return 5*f(x*10+5)+x;
}
```

 a. `f(5)` b. `f(2)` c. `f(1)` d. `f(0)`
- Variabila `m` memorează elementele unui tablou bidimensional cu 2025 de linii și 2025 de coloane, numerotate de la 0 la 2024. Indicați expresia C/C++ prin care poate fi accesat un element aflat pe diagonala secundară a tabloului.
 a. `m[1999][25]` b. `m[52:1999]` c. `m[25][52]` d. `m[25:1999]`
- Un arbore cu 11 noduri, numerotate de la 1 la 11, este reprezentat prin vectorul de „tați” (4,3,7,6,7,8,6,0,7,7,7). Indicați numărul maxim de descendenți direcți („fii”) ai unui nod.
 a. 6 b. 5 c. 4 d. 3
- Un graf neorientat are 25 de noduri și 5 componente conexe, fiecare dintre acestea fiind fără cicluri. Indicați numărul de muchii ale grafului.
 a. 20 b. 21 c. 24 d. 30

SUBIECTUL al II-lea**(40 de puncte)**

- Algoritmul alăturat este reprezentat în pseudocod.
 S-a notat cu $a \% b$ restul împărțirii numărului natural a la numărul natural nenul b .
 a. Scrieți ce se afișează în urma executării algoritmului dacă se citesc, în această ordine, numerele 20 și 25. (6p.)
 b. Dacă pentru `m` se citește numărul 5, scrieți două numere din intervalul [1,50] care pot fi citite pentru `n` astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, ultima valoare numerică afișată să fie 2. (6p.)
 c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
 d. Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind structura `pentru...execută` cu o structură repetitivă de tip `cât timp...execută`. (6p.)

```

citește m,n
    (numere naturale)
dacă m<n atunci
    n←n+m
    m←n-m
    n←n-m
    ■
    k←m
    pentru i←m,n,-1 execută
        scrie k, ' '
        dacă i%2=0 atunci
            k←k-1
            scrie '*'
        ■
        k←k-1
    ■

```

2. Utilizând metoda backtracking se generează toate grupele de accesorii pentru înot din mulțimea ordonată astfel: {cască, clipsuri pentru nas, costum de înot, dopuri pentru urechi, înotătoare, ochelari, pantofi acvatici, placă}. Accesoriile au prețurile următoare, exprimate în lei: cască - 20, clipsuri pentru nas - 10, costum de înot - 50, dopuri pentru urechi - 20, înotătoare - 150, ochelari - 50, pantofi acvatici - 100, placă - 150. Într-o grupă accesoriiile sunt distincte și costă, în total, 200 de lei. Două soluții diferă prin cel puțin un accesoriu. Primele trei soluții generate sunt, în această ordine: (cască, clipsuri pentru nas, costum de înot, dopuri pentru urechi, pantofi acvatici), (cască, clipsuri pentru nas, dopuri pentru urechi, înotătoare), (cască, clipsuri pentru nas, dopuri pentru urechi, ochelari, pantofi acvatici). Scrieți a patra și a cincea soluție obținute, în ordinea generării acestora. (6p.)
3. Variabila `s` memorează date ale fiecăruia dintre cei 30 de specialiști IT ai unei companii: date personale (codul numeric personal – CNP și anul nașterii) și anul angajării. Știind că expresiile C/C++ de mai jos reprezintă codul numeric personal (un șir de 13 caractere/cifre), anul nașterii, respectiv anul angajării (numere naturale din intervalul [1970,2025]) celui de al șaselea angajat, scrieți definiția unei structuri, cu eticheta `specialist`, înregistrare care să permită memorarea datelor unui specialist IT, și declarați corespunzător variabila `s`.
- `s[5].personal.CNP` `s[5].personal.anNastere` `s[5].anAngajare` (6p.)

SUBIECTUL al III-lea**(30 de puncte)**

1. Un număr „care aduce bucurie” - harsad (sau număr Niven), este un număr întreg divizibil cu suma cifrelor sale.
Subprogramul `harsad` are doi parametri:
- `k`, prin care primește un număr natural ($k \in [1, 10^6]$);
 - `n`, prin care furnizează cel mai mare număr natural harsad mai mic sau egal cu `k`.
- Scrieți definiția completă a subprogramului.
Exemplu: pentru `k=2027`, după apel, `n=2025` ($2+0+2+5=9$, iar 2025 este divizibil cu 9). (10p.)
2. Două cuvinte se numesc asemenea dacă sunt distincte și au același număr de vocale. Se consideră vocale literele `a`, `e`, `i`, `o`, `u`.
Scrieți un program C/C++ care citește de la tastatură un număr natural `n` ($n \in [1, 10^2]$), apoi `n` cuvinte, separate prin Enter. Fiecare cuvânt este format din cel mult 20 de caractere, numai litere mici ale alfabetului englez. Programul afișează pe ecran, separate prin câte un spațiu, toate cuvintele asemenea cu ultimul cuvânt citit, sau mesajul `nu exista` dacă nu există astfel de cuvinte.
Exemplu: dacă se citesc datele alăturate, se afișează pe ecran, nu neapărat în această ordine, cuvintele:
`lalelele brandusele` (10p.)
3. Pentru o paradă a modei sunt pregătite seturi de bijuterii, un set fiind format din cercei și pandantiv, cu câte cel puțin două pietre prețioase și semiprețioase. Sunt utilizate nouă tipuri de pietre, numerotate de la 1 la 9, iar orice bijuterie are o etichetă, număr natural în care fiecare cifră corespunde unei pietre din montură, în ordinea descrescătoare a importanței în cadrul modelului. Un set este potrivit dacă cele mai importante două pietre ale fiecărei bijuterii din set sunt de același tip, chiar dacă nu în aceeași ordine a importanței. Fișierul `bijuterii.in` conține numere naturale din intervalul [10, 999]: pe prima linie două numere `nc` și `np`, reprezentând numărul de cercei, respectiv de pandantive disponibile, pe a doua linie un șir de `nc` numere, reprezentând etichetele cerceilor, iar pe a treia linie un șir de `np` numere, reprezentând etichetele pandantivelor. Numerele aflate pe aceeași linie a fișierului sunt separate prin câte un spațiu. Se cere să se afișeze pe ecran mesajul `DA`, dacă se poate forma cel puțin un set potrivit de bijuterii, sau mesajul `NU`, în caz contrar. Proiectați un algoritm eficient din punctul de vedere al timpului de executare.
Exemplu: dacă fișierul conține numerele alăturate, se afișează pe ecran mesajul `DA`
- | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|
| 10 | 11 | | | | | | | | |
| 497 | 125 | 521 | 497 | 513 | 258 | 491 | 55 | 551 | 16 |
| 21 | 259 | 943 | 77 | 945 | 57 | 52 | 552 | 16 | 17 |
| 71 | | | | | | | | | |
- (două dintre cele 14 seturi potrivite se pot forma din cerceii cu eticheta 258 și fiecare dintre pandantivele cu etichetele 259, respectiv 52, pentru toate aceste bijuterii pietrele de tipurile 2 și 5 fiind cele mai importante).
- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)
- b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 6 – Sesiunea specială 2025

SUBIECTUL I**(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Variabilele m , n , p , q și x sunt de tip întreg, iar $m < p < q < n$. Indicați o expresie C/C++ care are valoarea 1 dacă și numai dacă numărul memorat în variabila x aparține intervalului $[m, n]$, dar **NU** și intervalului (p, q) .

- a. $x \leq m \ || \ x \geq p \ \&\& \ x \leq q \ || \ x \geq n$ b. $x \leq m \ \&\& \ x \geq p \ || \ x \leq q \ \&\& \ x \geq n$
 c. $x \geq m \ || \ x \leq p \ \&\& \ x \geq q \ || \ x \leq n$ d. $x \geq m \ \&\& \ x \leq p \ || \ x \geq q \ \&\& \ x \leq n$

2. Subprogramul f este definit alăturat. Indicați ce se afișează în urma apelului de mai jos.
 $f(2050);$

```
void f(int n)
{ cout<<n<<' '; | printf("%d ", n);
  if(n%10!=0) { cout<<n<<' '; | printf("%d ", n);
               f(n/10);
             }
  else if(n!=0) { f(n/10);
                cout<<n<<' '; | printf("%d ", n);
              }
}
```

- a. 2050 205 20 2 0 b. 2050 205 20 2 0 20 2050
 c. 2050 205 205 20 2 2 0 20 2050 d. 2050 2050 205 205 20 20 2 2 0

3. Utilizând metoda backtracking se generează toate tipurile de propoziții formate din patru unități sintactice (părți ale propoziției): un **subiect**, un **predicat**, un **atribut** și un **complement**, astfel încât atributul să fie plasat imediat înainte ori imediat după subiect. Două propoziții diferă prin ordinea unităților sale sintactice. Primele patru soluții generate sunt, în această ordine: (subiect, atribut, predicat, complement), (subiect, atribut, complement, predicat), (predicat, subiect, atribut, complement), (predicat, atribut, subiect, complement). Indicați ultima soluție generată.

- a. (complement, atribut, subiect, predicat) b. (complement, atribut, predicat, subiect)
 c. (complement, subiect, atribut, predicat) d. (complement, predicat, subiect, atribut)

4. Variabila m memorează elementele unui tablou bidimensional cu 100 de linii și 100 de coloane, numerotate de la 0 la 99. Indicați o expresie C/C++ prin care poate fi accesat un element aflat pe diagonala principală a tabloului.

- a. $m[1, 16]$ b. $m[16][16]$ c. $m(16, 16)$ d. $m(16)(1)$

5. Un graf neorientat are 7 noduri și 21 de muchii. Indicați numărul minim de muchii care pot fi eliminate din acest graf, astfel încât graful parțial obținut să aibă două componente conexe, cu cel puțin două noduri fiecare.

- a. 6 b. 8 c. 10 d. 12

SUBIECTUL al II-lea**(40 de puncte)****1. Algoritmul alăturat este reprezentat în pseudocod.**

S-a notat cu $a \div b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $[c]$ partea întreagă a numărului real c .

a. Scrieți numerele afișate dacă se citește, în această ordine, numerele 75 și 90. (6p.)

b. Dacă pentru variabila m se citește valoarea 2025, scrieți două numere distincte care pot fi citite pentru variabila n , astfel încât, pentru fiecare dintre acestea, în urma executării algoritmului, să se afișeze o singură valoare. (6p.)

c. Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)

d. Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind structura **pentru...execută** cu o structură repetitivă de tip **cât timp...execută**. (6p.)

2. Un arbore cu 8 noduri, numerotate de la 1 la 8, este reprezentat prin vectorul de „tați” (3, 0, 2, 5, 2, 5, 1, 5). Enumerați două noduri ale arborelui care pot fi alese drept rădăcină (cu excepția rădăcinii curente), astfel încât nodul 4 să aibă aceiași „frați” ca în reprezentarea dată. (6p.)

3. În declarațiile alăturate variabila d memorează în câmpurile A și B coordonatele, în sistemul de coordonate xOy , ale vârfurilor din stânga – sus, respectiv din dreapta – jos ale unui dreptunghi cu laturile paralele cu axele sistemului de coordonate. Scrieți o secvență de instrucțiuni C/C++ în urma executării căreia se afișează pe ecran mesajul **DA**, dacă figura corespunzătoare variabilei d este pătrat sau mesajul **NU** în caz contrar. (6p.)

```
struct punct
{ int x,y;
};
struct figura
{ punct A, B;
} d;
```

SUBIECTUL al III-lea**(30 de puncte)**

1. Subprogramul **diviz** are un singur parametru, n , prin care primește un număr natural ($n \in [1, 10^3]$). Subprogramul returnează cel mai mare divizor al lui n care este pătrat perfect.

Scrieți definiția completă a subprogramului C/C++.

Exemplu: pentru $n=72$ subprogramul returnează 36, pentru $n=16$ subprogramul returnează 16, iar pentru $n=15$ subprogramul returnează 1. (10p.)

2. Într-un text de cel mult 100 de caractere cuvintele sunt formate doar din litere mici ale alfabetului englez și sunt separate prin unul sau mai multe spații. Textul are cel puțin două cuvinte.

Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat mai sus și îl transformă în memorie prin eliminarea sau inserarea unor spații și a unor cratime (simbolul –), astfel încât între oricare două cuvinte consecutive în text să fie câte o cratimă, încadrată la stânga și la dreapta de câte un spațiu, ca în exemplu. Programul afișează pe ecran textul obținut.

Exemplu: pentru textul

anul acesta devin student

se obține

anul - acesta - devin - student

(10p.)

3. Fișierul text **bac.txt** conține un șir de cel mult 10^6 triplete de numere naturale din intervalul $[1, 10^2]$, numerele din fiecare triplet reprezentând lungimile laturilor câte unui triunghi. Fiecare triplet se află pe câte o linie a fișierului, iar numerele care îl compun sunt separate prin câte un spațiu.

Într-un triunghi dreptunghic pătratul lungimii ipotenuzei este egal cu suma pătratelor lungimilor celor două catete.

Se cere să se afișeze pe ecran numărul maxim de triunghiuri dreptunghice din fișier care au aceeași lungime a ipotenuzei. Proiectați un algoritm eficient din punctul de vedere al timpului de executare.

Exemplu: dacă fișierul conține numerele alăturate, pe ecran se afișează 3

(sunt trei triunghiuri de tipul cerut cu ipotenuza 25: două au catetele 7, respectiv 24, și unul are catetele 15, respectiv 20).

```
6 10 6
7 24 25
7 25 24
25 14 13
8 6 10
15 20 25
4 5 3
```

a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)

b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 1 – Sesiunea iunie-iulie 2025

SUBIECTUL I

(20 de puncte)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

- Indicați expresia C/C++ cu aceeași valoare ca a expresiei alăturate. 2025%2019+6
 - 2025/2020+5
 - 2025/2021+8
 - 2025%2020+5
 - 2025%2021+8
- Subprogramul `f` este definit alăturat. Indicați ce se afișează în urma apelului de mai jos. `f(3);`

```
void f(int n)
{
    int i;
    for(i=1; i<=n; i++) if(i%2==0) { cout<<i; | printf("%d", i);
                                f(i-1);
                                }
                                else { f(i-1);
                                cout<<i; | printf("%d", i);
                                }
}
```

 - 1211213
 - 123121
 - 123
 - 01201012013
- Indicați o declarație a unui tablou bidimensional `m`, care poate memora maximum 100 de numere reale.
 - `float m[2, 50];`
 - `float m[4][25];`
 - `float m[10] x float m[10];`
 - `int m[100];`
- Utilizând metoda backtracking, s-au generat toate codurile posibile pentru deblocarea unor telefoane, coduri de câte 6 cifre distincte, din mulțimea cifrelor, ordonată crescător. Fiecare cod are primele trei cifre impare și ultimele trei cifre pare. Primele patru coduri sunt 135024, 135026, 135028, 135042. Indicați penultimul cod generat.
 - 957862
 - 957846
 - 975862
 - 975846
- Un graf orientat fără circuite are 10 vârfuri. Indicați numărul maxim de arce ale grafului.
 - 10
 - 45
 - 50
 - 90

SUBIECTUL al II-lea

(40 de puncte)

- Algoritmul alăturat este reprezentat în pseudocod.
 - Scrieți ce se afișează în urma executării algoritmului dacă se citește, în această ordine, numerele 7 și 17. (6p.)
 - Dacă pentru variabila `n` se citește valoarea 25, scrieți două numere distincte care pot fi citite pentru variabila `m`, astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, să se afișeze valoarea 25. (6p.)
 - Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
 - Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind structura `cât timp...execută` cu o structură repetitivă cu test final. (6p.)

```

citește m, n
    (numere naturale nenule, m ≤ n)
nr ← 0; i ← m
repetă
    x ← 1
    cât timp x * x < i execută
        x ← x + 1
    ■
    dacă x * x = i atunci nr ← i
    altfel i ← i + 1
    ■
până când i > n sau nr ≠ 0
scrie nr

```

2. Un graf neorientat cu 6 noduri, numerotate de la 1 la 6, este reprezentat prin listele de adiacență alăturate. Scrieți mulțimea nodurilor și mulțimea muchiilor unui subgraf al acestuia, fără noduri izolate, care să fie graf eulerian. (6p.)
- | | |
|----|------------|
| 1: | 2, 3, 4, 6 |
| 2: | 1, 3, 5 |
| 3: | 1, 2, 5 |
| 4: | 1, 5, 6 |
| 5: | 2, 3, 4, 6 |
| 6: | 1, 4, 5 |
3. Variabila **p** memorează simultan, pentru un tip de prăjitură, codul (un număr natural de două cifre), prețul (număr real) și un set de trei numere naturale din intervalul $[1, 10^2]$, reprezentând informații specifice, în această ordine: tipul glazurii, tipul cremei principale și numărul de blaturi. Știind că expresiile C/C++ de mai jos au ca valori codul, prețul, respectiv tipul glazurii pentru o prăjitură, scrieți definiția unei structuri cu eticheta **prajitura**, care permite memorarea datelor despre o prăjitură, și declarați corespunzător variabila **p**.
- p.cod p.pret p.informatii[0]** (6p.)

SUBIECTUL al III-lea**(30 de puncte)**

1. Numărul natural **an** este **ascendent** al numărului natural **n**, dacă oricare dintre cifrele lui **an** este mai mare sau egală cu cifra unităților lui **n**.
Exemplu: oricare dintre numerele 7, 9, 98 sau 7998 este ascendent al lui 827, dar numărul 857 nu este ascendent al lui 827.
 Subprogramul **ascendent** are trei parametri:
 • **n**, prin care primește un număr natural ($n \in [0, 10^3]$);
 • **x** și **y**, prin care primește câte un număr natural din intervalul $[0, 10^3]$ ($x < y$).
 Subprogramul returnează suma ascendenților lui **n** din intervalul $[x, y]$, sau valoarea 0, dacă nu există niciun astfel de ascendent. Scrieți definiția completă a subprogramului C/C++.
Exemplu: dacă $n=827$, $x=9$, $y=800$, subprogramul returnează 7893 ($9+77+78+79+87+88+89+97+98+99+777+778+779+787+788+789+797+798+799=7893$). (10p.)
2. Un cuvânt **semioglindit** se obține dintr-un cuvânt cu $2 \cdot k$ ($k \in [1, 10^2]$) litere, prin interschimbarea în acesta a secvenței formate din primele **k** litere cu secvența formată din ultimele **k** litere.
Exemplu: din cuvântul platim se obține cuvântul semioglindit timpla.
 Într-un text de cel mult 200 de caractere, cuvintele sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat, pe care îl transformă în memorie, prin înlocuirea fiecărui cuvânt cu număr par de litere, cu cel semioglindit obținut din acesta, ca în exemplu. Programul afișează pe ecran textul obținut, sau mesajul **nu exista**, dacă toate cuvintele au număr impar de litere.
Exemplu: pentru textul am facut fotografii unei flori mari se afișează pe ecran textul ma facut rafiifotog eiun flori rima (10p.)
3. Un tânăr pasionat de călătorii are o listă cu muzee virtuale și, pentru fiecare, câte un singur interval orar, în care acesta poate fi vizitat online, gratuit. Tânărul dispune zilnic de același interval orar pentru vizite; un muzeu este **convenabil** dacă poate fi vizitat online gratuit în timpul disponibil și dacă pentru vizită îi poate alocă cel puțin o oră. Muzeele din listă sunt numerotate cu valori naturale consecutive, începând cu 1, și cel puțin unul este convenabil.
 Fișierul text **bac.in** conține cel mult 10^5 linii, iar pe fiecare linie câte o pereche de numere, reprezentând limitele câte unui interval orar: pe prima linie intervalul orar de care tânărul dispune zilnic, iar pe fiecare dintre următoarele linii, intervalul orar de vizitare gratuită pentru câte un muzeu, în ordinea din listă. Limitele intervalelor sunt ore fixe, numere naturale din intervalul $[8, 22]$, iar cele aflate pe aceeași linie a fișierului sunt în ordine strict crescătoare și sunt separate printr-un spațiu.
 Se cere să se afișeze pe ecran, separate printr-un spațiu, două valori, reprezentând numărul de muzee convenabile, respectiv numărul de ordine al ultimului astfel de muzeu din lista tânărului. Utilizați un algoritm eficient din punctul de vedere al timpului de executare și al memoriei utilizate.
Exemplu: dacă fișierul conține valorile alăturate, atunci pe ecran se afișează numerele 3 4.
 (pot fi vizitate trei muzee cu numerele de ordine 1, 2 și 4, în intervalele 16–18, 17–19, respectiv 18–19).
- | | |
|----|----|
| 16 | 19 |
| 15 | 18 |
| 17 | 21 |
| 19 | 21 |
| 18 | 20 |
| 12 | 13 |
- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)
 b. Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)

Varianta 4 – Sesiunea august 2025

SUBIECTUL I

(20 de puncte)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

- Indicați intervalul căruia îi aparține valoarea memorată în variabila `x`, dacă și numai dacă expresia C/C++ alăturată are valoarea 1. `!(x<=2020) && !(x>2025)`
 - (2020, 2025]
 - (2020, 2025)
 - [2020, 2025]
 - (2021, 2025)
- Variabila `k` este de tip întreg, iar variabila `s` permite memorarea unui șir de maximum 20 de caractere. Indicați valoarea variabilei `k` în urma executării secvenței alăturate. `strcpy(s, "calculator");
k=strchr(s, s[1]) - strchr(s, s[3]);`
 - 9
 - 3
 - 2
 - 1
- La o probă de dans a unui concurs s-au calificat 6 perechi, notate cu litere distincte din mulțimea ordonată {A, B, C, D, E, F}, fiecare pereche prezentând câte un dans. Utilizând metoda backtracking, se generează toate posibilitățile de a stabili ordinea susținerii probei, astfel încât perechea A să prezinte prima, iar perechea F să prezinte ultima. Primele trei soluții generate sunt (A, B, C, D, E, F), (A, B, C, E, D, F), (A, B, D, C, E, F). Indicați ultima soluție generată.
 - (A, D, B, C, E, F)
 - (A, E, D, B, C, F)
 - (A, E, D, C, B, F)
 - (A, F, E, D, C, B)
- Un graf orientat cu 6 vârfuri, numerotate de la 1 la 6, are arcele (3, 1), (4, 1), (4, 2), (4, 3), (5, 6), (6, 4). Indicați numărul vârfurilor pentru care gradul exterior este egal cu gradul interior.
 - 1
 - 2
 - 3
 - 4
- Într-un arbore cu 50 de noduri, numerotate de la 1 la 50, rădăcina este nodul numerotat cu 1, iar tatăl oricărui alt nod `i` al său este nodul numerotat cu `[i/2]`. Indicați lungimea lanțului elementar cu o extremitate în nodul 32 și cealaltă extremitate în nodul 23.
 - 5
 - 7
 - 9
 - 11

SUBIECTUL al II-lea

(40 de puncte)

- Algoritmul alăturat este reprezentat în pseudocod.
S-a notat cu `a%b` restul împărțirii numărului natural `a` la numărul natural nenul `b` și cu `[c]` partea întreagă a numărului real `c`.
 - Scrieți ce se afișează în urma executării algoritmului dacă se citesc, în această ordine, numerele 9767 și 8204. (6p.)
 - Dacă primul număr citit este 2025, scrieți două valori distincte din intervalul [10, 99] care pot fi citite pentru `y` astfel încât, pentru fiecare dintre acestea, în urma executării algoritmului, mesajul afișat să fie DA, urmat de o valoare numerică. (6p.)
 - Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)
 - Scrieți în pseudocod un algoritm echivalent cu cel dat, înlocuind prima structură `cât timp...execută` cu o structură repetitivă cu test final. (6p.)

```

citește x, y
  (numere naturale distincte)
cx ← 0; cy ← 0
cât timp x ≥ 10 sau y ≥ 10 execută
  cât timp x + y ≠ 0 execută
    cx ← cx + x % 10; x ← [x / 10]
    cy ← cy + y % 10; y ← [y / 10]
  x ← cx; cx ← 0; y ← cy; cy ← 0
dacă x = y atunci scrie "DA ", x
altfel scrie "NU ", x, " ", y

```

2. Subprogramul **f** este definit alăturat. Scrieți valorile **f(3, 9)** și **f(1, 1000)**.

(6p.)

```
int f(int x, int y)
{ if(x*5>y/5) return x;
  return f(x*5, y/5);
}
```

3. Variabila **b** memorează simultan date despre o bijuterie: greutatea acesteia, exprimată în grame, și metalul prețios din care este confecționată (denumirea și prețul unui gram din acel metal). Greutatea bijuteriei și prețul unui gram de metal sunt numere naturale din intervalul $[1, 10^4]$, iar denumirea este un șir de maximum 30 de caractere. Știind că expresiile C/C++ de mai jos au ca valori numele metalului din care este confecționată bijuteria, respectiv prețul acesteia, scrieți definiția unei structuri cu eticheta **bijuterie**, care permite memorarea datelor precizate mai sus, și declarați corespunzător variabila **b**.

b.metal.denumire**b.greutate*b.metal.preț**

(6p.)

SUBIECTUL al III-lea**(30 de puncte)**

1. Un fermier are un teren de formă dreptunghiulară, pe care vrea să îl împartă în parcele de arie egală, astfel încât numărul de parcele și aria să fie valori naturale, iar numărul de parcele să fie par și strict mai mic decât aria unei parcele. Subprogramul **teren** are doi parametri, **x** și **y**, prin care primește două numere naturale din intervalul $[2, 100]$, reprezentând lungimea și lățimea terenului, măsurate în metri. Subprogramul afișează toate soluțiile posibile de împărțire a terenului, fiecare soluție pe câte o linie a ecranului, sub forma a două numere, separate prin mesajul **parcele de arie** și spații adecvate, unde primul număr reprezintă numărul de parcele obținute, iar al doilea aria unei parcele corespunzătoare, ca în exemplu. Dacă nu există nicio astfel de soluție, subprogramul afișează pe ecran mesajul **nu exista**. Scrieți definiția completă a subprogramului C/C++.

Exemplu: pentru **x=6** și **y=8**, subprogramul afișează pe ecran valorile alăturate, nu neapărat în această ordine (terenul poate fi împărțit și în 3 parcele de arie 16 sau 8 parcele de arie 6, dar nu corespund cerinței). (10p.)

```
2 parcele de arie 24
4 parcele de arie 12
6 parcele de arie 8
```

2. Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, **m** și **n**, și elementele unui tablou bidimensional cu **m** linii și **n** coloane, numere naturale distincte din intervalul $[0, 10^3]$. Programul modifică tabloul în memorie, înlocuind valorile tuturor elementelor aflate pe coloana care conține numărul minim memorat în tablou cu valoarea elementului aflat pe ultima linie și ultima coloană. Tabloul obținut este afișat pe ecran, câte o linie a tabloului pe câte o linie a ecranului, elementele fiecărei linii fiind separate prin câte un spațiu.

Exemplu: pentru **m=3**, **n=4** și

17	12	5	19
10	16	2	4
11	21	8	9

se obține

17	12	9	19
10	16	9	4
11	21	9	9

tabloul

(10p.)

3. La o centrală solară se monitorizează **stocul zilnic** de energie produsă și **stocul total** calculat pentru fiecare perioadă. Zilele de monitorizare sunt numerotate cu valori naturale consecutive, în ordine cronologică, începând cu ziua 1. O perioadă este formată din cel puțin două zile de monitorizare, consecutive, iar stocul total calculat pentru ea este suma stocurilor zilnice corespunzătoare. O **zi este validată** dacă stocul zilnic este cel puțin egal cu limita zilnică, **minz**; o **perioadă este validată** dacă stocul total calculat pentru ea este cel puțin egal cu limita stabilită pentru perioade, **minp**, fiecare zi a perioadei este validată, iar perioada este maximală în raport cu această proprietate (nu i se mai poate adăuga nicio zi validată).

Fișierul text **bac.in** conține cel mult 10^6 numere naturale din intervalul $[1, 10^3]$: pe prima linie **minz** și **minp**, reprezentând limitele precizate pentru validare, iar pe a doua linie stocurile zilnice de energie produse în zile consecutive, în ordinea monitorizării. Numerele aflate pe aceeași linie a fișierului sunt separate prin câte un spațiu.

Se cere să se afișeze pe ecran, pentru fiecare perioadă de producție validată, corespunzătoare datelor din fișier, câte un triplet de numere, reprezentând prima și ultima zi a perioadei, respectiv stocul total calculat pentru ea. Valorile din fiecare triplet se afișează pe câte o linie a ecranului, separate prin câte un spațiu, iar dacă nu există nicio astfel de perioadă, se afișează mesajul **nu exista**. Proiectați un algoritm eficient din punctul de vedere al memoriei utilizate și al timpului de executare.

Exemplu: dacă fișierul conține valorile alăturate,

3 6 71

13 15 90

10 40

65 9 20 25 12 14 7 3 11 15 12 8 19 50 21

- a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia.

(2p.)

- b. Scrieți programul C/C++ corespunzător algoritmului proiectat.

(8p.)


```

        if(a[0][j]==a[1][j-1] && a[0][j]==a[1][j+1] && a[0][j]==a[2][j]){
            ok=1;
        }
    }
}

```

Subiectul III.

1. void rest(int x, int y, int n, int &k)

```

{
    int i;
    k=0;
    for (i=n; i>=2; i--)
        if (i%x==2 && i%y==2)
            {
                k=i;
                break;
            }
}

```

2. #include <iostream>

#include <cstring>

using namespace std;

int main()

{

char s[251], x[31], *p, cuv[31];

int ok=0;

cin.getline(s, 251);

cin>>x;

p= strtok(s, " ");

while(p)

{

strcpy(cuv, p);

p= strtok(NULL, " ");

if (strcmp (cuv,x)==0)

{

cout<<p<<" ";

ok=1;

}

p= strtok(NULL, " ");

}

if(ok==0)

cout<<"NU";

return 0;

}

3. a. **Soluția intuitivă**

Este cunoscut de la matematică că descompunerea în factori primi ai numărului 45 este: $45=3^2 \cdot 5$.

Pentru ca numărul 45^p să fie divizor al produsului $1 \cdot 2 \cdot 3 \cdot \dots \cdot n = n!$ este necesar ca în descompunerea în factori primi a lui $n!$ să existe cel puțin $2 \cdot p$ factori egali cu 3 și cel puțin p factori egali cu 5.

Algoritmul descris parcurge pe rând numerele naturale din intervalul $[2, n]$. Pentru fiecare număr k , se determină numărul de apariții ale factorului prim 3 în descompunerea lui $n!$ în factori primi și se adaugă la suma st . În mod similar, se determină numărul de apariții ale factorului prim 5 și se adaugă la suma sc .

La finalul parcurgerii numerelor naturale din intervalul $[2, n]$, st reprezintă numărul total de factori egali cu 3, iar sc numărul total de factori egali cu 5 din descompunerea lui $n!$.

Deoarece fiecare putere a lui 45 necesită doi factori egali cu 3 și un factor egal cu 5, valoarea maximă a lui p , valoare care se scrie în fișierul `bac.txt` este calculată cu formula:

$$p = \min(\lfloor st/2 \rfloor, sc)$$

Algoritmul este eficient din punct de vedere al timpului de executare, deoarece parcurge o singură dată intervalul $[2, n]$, iar pentru fiecare număr efectuează numai împărțiri repetate la 3 și la 5, adică are o complexitate de timp $O(n \cdot \log n)$. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece algoritmul nu folosește structuri de tip tablou, adică are o complexitate de memorie $O(1)$.

b.

```
#include <fstream>
using namespace std;
ofstream g("bac.txt");
int n, k, st, sc, x;
int main()
{
    cin >> n;
    for (k = 2; k <= n; k++)
    {
        x = k;
        while (x % 3 == 0)
        {
            st++;
            x = x / 3;
        }
        x = k;
        while (x % 5 == 0)
        {
            sc++;
            x = x / 5;
        }
    }
    if (st / 2 < sc)
        g << st / 2;
    else
        g << sc;
    g.close();
    return 0;
}
```

Soluția optimă

Algoritmul descris utilizează formula lui Legendre pentru determinarea exponentului unui factor prim în descompunerea lui $n!$.

Această soluție determină valorile st și sc astfel:

$$st = [n/3] + [n/3^2] + [n/3^3] + [n/3^4] + \dots + 0$$

$$sc = [n/5] + [n/5^2] + [n/5^3] + [n/5^4] + \dots + 0$$

Valoarea cerută și scrisă în fișierul de ieșire este calculată cu formula: $\min([st/2], sc)$.

Algoritmul este eficient din punct de vedere al timpului de executare, deoarece are o complexitate de timp $O(\log n)$. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece algoritmul nu folosește structuri de tip tablou, adică are o complexitate de memorie $O(1)$.

```
#include <iostream>
#include <fstream>
using namespace std;
ofstream g("bac.txt");
int n, st, sc, i;
int main()
{
    cin >> n;
    for (i = 3; i <= n; i = i*3)
        st = st + n / i;
    for (i = 5; i <= n; i = i* 5)
        sc = sc + n / i;
    if (st / 2 < sc)
        g << st / 2;
    else
        g << sc;
    g.close();
    return 0;
}
```

Varianta 4 – Sesiune specială 2022

Subiectul I.

1.d 2.c 3. a 4.c 5.b

Subiectul II.

1. a. Răspuns corect: 37 23

b. Cele două numere conform cerinței sunt de forma $\overline{xyz}$, unde $\overline{xy}$ și $\overline{yz}$ sunt numere prime.
Exemplu de răspuns corect: 137 231.

```
c. #include <iostream>
using namespace std;
int main()
{
    int n, x, p;
    cin >> n;
    do
    {
        x = n % 100;
        if (x > 1)
        {
            p = (x % 2) * (x % 3) * (x % 5) * (x % 7);
            if (p != 0)
                cout << x << " ";
        }
    }
    while (x > 0);
}
```

```

    }
    n=n/10;
  } while(n>=10);
  return 0;
}

```

d.

citește n (număr natural)

 $x \leftarrow n \% 100$

```

dacă x>1 atunci
  p←(x%2)*(x%3)*(x%5)*(x%7)
  dacă p≠0 atunci
    scrie x, ' '
  ■
■
cât timp n≥10 execută
  x←n%100
  dacă x>1 atunci
    p←(x%2)*(x%3)*(x%5)*(x%7)
    dacă p≠0 atunci
      scrie x, ' '
    ■
  ■
■

```

2. [1,2], [1,3], [1,5], [2,3], [2,5], [3,5]

3. s=0;

for (i=1; i<=15; i++)

s=s+t[i].pret*t[i].nrLocuri;

Subiectul III.

1. Soluție propusă:

Relația data de problemă este: $x^2 \cdot y^2 = n$, relație din care rezultă că: $(x \cdot y)^2 = n$.

Pași de rezolvare:

- Se verifică dacă n este pătrat perfect. Dacă nu este, atunci nu există soluție.
- Dacă $m = \sqrt{n}$, atunci determinăm două numere x și y, care îndeplinesc următoarele condiții:
 - $x \cdot y = m$
 - $2 \leq x < y$
 - x să fie minim.
- Pentru a obține valoarea minimă a lui x, se parcurg divizorii lui m în ordine crescătoare, începând cu valoarea 2. Prima pereche găsită este și soluția cerută.
- Dacă nu există o astfel de pereche, $x=0$ și $y=0$.

```

void patrate(int n, int &x, int &y)
{

```

```

    int m, d;
    m= (int)sqrt(n);
    if (m * m != n)
    {
        x = 0;
        y = 0;
        return;
    }
    for (d = 2; d * d <= m; d++)
        if (m % d == 0)
        {
            x = d;
            y = m / d;
            if (x < y)
                return;
        }
    x = 0;
    y = 0;
}

```

```

2. #include <iostream>
using namespace std;
int main(){
    int n, i, j, a[101][101];
    cin>>n;
    for (i=1; i<=n; i++)
        for (j=1; j<=n; j++)
            cin>>a[i][j];
    for (i=1; i<=n; i++)
        for (j=i+1; j<=n; j++)
            a[j-1][i]=a[j][i];
    n--;
    for (i=1; i<=n; i++)
    {
        for(j=1; j<=n; j++)
            cout<<a[i][j]<<" ";

        cout<<endl;
    }
    return 0;
}

```

3. a. O soluție posibilă parcurge șirul din fișier, memorând în variabilele x și y ultimele două valori citite din fișier. Pentru fiecare nouă valoare citită din fișier:
- dacă este egală cu valoarea precedentă ($y=x$), se incrementează numărul de apariții consecutive (nr);
 - dacă este diferită, înseamnă că s-a încheiat secvența valorii precedente.
 - dacă numărul de apariții este egal cu valoarea respectivă ($nr=x$), atunci această valoare poate face parte dintr-o secvență progresivă, iar lungimea secvenței curente (lc) se mărește cu nr;
 - în caz contrar, dacă $nr < x$, secvența progresivă se întrerupe și lungimea secvenței curente se reinițializează cu 0, adică $lc=0$.

- după fiecare actualizare se compară lungimea secvenței curente cu lungimea maximă (lmax) și, dacă este cazul, aceasta se actualizează.

La sfârșitul fișierului se prelucrează în același mod și ultima secvență de valori egale, deoarece aceasta nu este urmată de o valoare diferită.

Algoritmul descris este eficient deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, adică are o complexitate $O(1)$.

b.

```
#include<iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
int x, y, nr = 1, lc, lmax;
int main()
{
    f>>x;
    while (f>> y)
    {
        if (x == y)
            nr++;
        else
        {
            if (nr == x)
            {
                lc =lc+ nr;
                if (lc > lmax)
                    lmax = lc;
            }
            else
                lc = 0;
            x = y;
            nr = 1;
        }
    }
    if (nr == x)
    {
        lc =lc+ nr;
        if (lc > lmax)
            lmax = lc;
    }
    cout << lmax;
    f.close();
    return 0;
}
```

Varianta 1 – Sesiunea iunie-iulie 2022**Subiectul I.**

1.c 2.d 3.a 4.b 5.c

Subiectul II.

1. a. Răspuns corect: 4

b. Orice număr prim din interval $[10, 99]$ este o soluție corectă.

Exemplu de răspuns corect: 13 17

c.

```

#include <iostream>
using namespace std;
int main( )
{
    int n, i, k;
    cin>>n;
    i=2; k=0;
    while (n>=i)
    {
        while (n%i==0)
        {
            k=k+1;
            n=n/i;
        }
        if (i==2)
            i=i+1;
        else
            i=i+2;
    }
    cout<<k;
    return 0;
}

```

d.

citește n (număr natural)

 $i \leftarrow 2; k \leftarrow 0$

```

cât timp  $n \geq i$  execută
┌─dacă  $n \% i = 0$  atunci
│   ┌─repetă
│   │    $k \leftarrow k + 1$ 
│   │    $n \leftarrow [n/i]$ 
│   └─până când  $n \% i \neq 0$ 
└─
┌─dacă  $i = 2$  atunci  $i \leftarrow i + 1$ 
└─altfel  $i \leftarrow i + 2$ 

```

scrie k

2. Oricare dintre numerele 2022, 2023, 2024, 2025 sunt răspunsuri corecte.
Exemplu de răspuns corect: 2022 2023

3. `strcpy(id, strchr(s, " ") + 1);`
`strcat(id, "2022");`

Subiectul III.

```
1. void secventa(int &n)
{
    nou = 0, p = 1, u, z;
    while (n > 0)
    {
        u = n % 10;
        z = (n / 10) % 10;
        if (z == 2 && u == 2)
        {
            nou = nou + 0 * p;
            p = p * 10;
            nou = nou + 2 * p;
            p = p * 10;
            n = n / 100;
        }
        else
        {
            nou = nou + u * p;
            p = p * 10;
            n = n / 10;
        }
    }
    n = nou;
}
```

```
2. #include <iostream>
using namespace std;
int a[105][105], m, n, i, j, minim, s ;
int main()
{
    cin >> m >> n;
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j = j++)
            cin >> a[i][j];
    for(i = 0; i <= m + 1; i++)
    {
        a[i][0] = 11;
        a[i][n + 1] = 11;
    }
    for(j = 0; j <= n + 1; j++)
    {
        a[0][j] = 11;
```

```

        a[m + 1][j] = 11;
    }
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j++)
        {
            minim = a[i - 1][j];
            if(a[i + 1][j] < minim)
                minim = a[i + 1][j];
            if(a[i][j - 1] < minim)
                minim = a[i][j - 1];
            if(a[i][j + 1] < minim)
                minim = a[i][j + 1];
            if(a[i][j] < minim)
                s = s + minim - a[i][j];
        }

    cout << s;
    return 0;
}

```

3. a. Algoritmul descris citește din fișier valorile x și y de pe primul rând al fișierului, apoi se parcurg o singură dată numerele de pe a doua linie a fișierului. Se memorează ultima valoare citită în variabila a . Pentru fiecare număr citit din fișier cu variabila b , dacă b aparține intervalului $[x, y]$ și este diferit de valoarea precedent, adică este diferit de a , atunci se incrementează contorul valorilor distincte, adică $nr++$ și valoarea curentă devine noua valoare precedentă. Algoritmul afișează contorul nr .

Algoritmul descris este eficient deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, adică are o complexitate $O(1)$.

```

b.
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
int x, y, nr, a, b;
int main()
{
    f >> x >> y;
    f >> a;
    if(a >= x && a <= y)
        nr = 1;
    while(f >> b)
    {
        if(b != a)
            if(b >= x && b <= y)
                nr++;
        a = b;
    }
    cout << nr;
    f.close();
}

```

```

    return 0;
}

```

Varianta 5 – Sesiunea august 2022

Subiectul I.

1.b 2.a 3.d 4.c 5.b

Subiectul II.

1. a. Răspuns corect: 45

b. Orice număr natural din intervalul $[10, 14]$ reprezintă un răspuns corect.

Exemplu de răspuns corect: 11 12

c.

```

#include <iostream>
using namespace std;
int main()
{
    int m, n, x, y;
    cin >> m >> n;
    x = 1;
    while (x == 1)
    {
        x = m; y = n; n = n - 1;
        while (x != y)
        {
            if (x > y)
                x = x - y;
            else
                y = y - x;
        }
    }
    cout << n + 1;
    return 0;
}

```

d.

citește m, n (numere naturale nenule, $m < n$)

$x \leftarrow 1$

dacă $x = 1$ atunci

repetă

$x \leftarrow m; y \leftarrow n; n \leftarrow n - 1$

cât timp $x \neq y$ execută

dacă $x > y$ atunci $x \leftarrow x - y$

altfel $y \leftarrow y - x$

■ până când $x \neq 1$

scrie n+1

2. Răspuns corect: 2 8

```
3. if(x.temperatura>11)
    cout<<"CALDUROS";
else
    if (x.temperatura<10)
        cout<<"RACOROS";
    else
        cout<<"NORMAL";
```

Subiectul III.

```
1. void schimb(int &n, int x, int p)
{
    int putere = 1, i, c;
    for(i = 1; i <= p; i++)
        putere = putere * 10;
    c = (n / putere) % 10;
    n = n - c * putere + x * putere;
}

2. #include <iostream>
#include <cstring>
using namespace std;
char s[51];
int i;
int main()
{
    cin >> s;
    for(i = 0; i < strlen(s); i++)
    {
        if(s[i] >= 'b' && s[i] <= 'd')
            s[i] = 'a';
        else
            if(s[i] >= 'f' && s[i] <= 'h')
                s[i] = 'e';
            else
                if(s[i] >= 'j' && s[i] <= 'n')
                    s[i] = 'i';
                else
                    if(s[i] >= 'p' && s[i] <= 't')
                        s[i] = 'o';
                    else
                        if(s[i] >= 'v' && s[i] <= 'z')
                            s[i] = 'u';
    }
    cout << s;
    return 0;
}
```

3. a. Algoritmul descris extrage în variabila x primul număr din fișier, apoi pe rând toate celelalte numere din fișier, cu ajutorul variabilei y . Astfel se păstrează, în fiecare moment, două valori aflate pe poziții consecutive în fișier.

Algoritmul memorează lungimea secvenței paritare curente (lg), lungimea maximă găsită ($lgmax$) și numărul secvențelor de lungime maximă (nr). Pentru fiecare număr citit (y) se compară paritatea sa cu paritatea numărului precedent (x) și dacă au aceeași paritate, secvența curentă continuă, iar în caz contrar, secvența curentă se încheie și se compară lungimea acesteia cu lungimea maximă. Dacă este mai mare, se actualizează lungimea maximă și numărul secvențelor devine 1. Dacă este egală cu lungimea maximă, se incrementează numărul secvențelor și se începe o nouă secvență. După terminarea citirii se prelucrează și ultima secvență din fișier.

Algoritmul descris este eficient deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, parcurge șirul o singură dată, fără a memora toate valorile, adică are o complexitate $O(1)$.

b.

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
int x, y, lg = 1, lgmax, nr;
int main()
{
    f >> x;
    while(f >> y)
    {
        if(x % 2 == y % 2)
            lg = lg + 1;
        else
        {
            if(lg > lgmax)
            {
                lgmax = lg;
                nr = 1;
            }
            else
                if(lg == lgmax)
                    nr = nr + 1;
            lg = 1;
        }
        x = y;
    }
    if(lg > lgmax)
    {
        lgmax = lg;
        nr = 1;
    }
    else
        if(lg == lgmax)
            nr = nr + 1;
}
```

```

    cout << nr << " " << lgmax;
    f.close();
    return 0;
}

```

Varianta – Simulare 2023

Subiectul I.

1.d 2.b 3.a 4.d 5.c

Subiectul II.

1. a. Răspuns corect: 474

b. Oricare dintre valorile 521, 531, 5021, 5031.

Exemplu de răspuns corect: 521 531 5021

c.

```

#include <iostream>
using namespace std;
int main( )
{
    int n, cn, p, c, x;
    cin>>n;
    x=5;
    do
    {
        cn=n; n=0; p=1;
        do
        {
            c=cn%10;
            if(c==x)
                c=5-c;
            n=c*p+n; cn=cn/10; p=p*10;
        } while (cn!=0);
        x=x-1;
    } while(x!=0);
    cout<<n;
    return 0;
}

```

d.

citește n (număr natural)

pentru $x \leftarrow 5, 1, -1$ execută

$cn \leftarrow n$; $n \leftarrow 0$; $p \leftarrow 1$;

repetă

$c \leftarrow cn \% 10$

dacă $c = x$ atunci $c \leftarrow 5 - c$

■

```

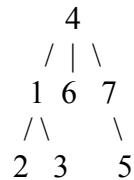
┌ n ← c * p + n; cn ← [cn/10]; p ← p * 10
└─▶ până când cn = 0
■
scrie n

```

2. Exemplu de răspuns corect: (1,2), (3,5), (4,6)

Pași de rezolvare:

- Se construiește arborele



- Se calculează gradele tuturor nodurilor
- Se identifică nodurile de grad impar
- Se construiesc muchii noi, astfel încât graful obținut să fie conex și toate nodurile sale să aibă grade pare (graf eulerian).

3.

```

for (i=1; i<=5; i++)
    for (j=1; j<=5; j++)
        if(i+j<=5)
            a[i][j]=5-j;
        else
            a[i][j]=i-1;

```

Subiectul III.

```

1. void NrImp(int x, int y, int &nr)
{
    int i, d, cnt;
    nr = 0;
    for(i = x; i <= y; i++)
    {
        cnt = 0;
        for(d = 1; d <= i; d = d + 2)
            if(i % d == 0)
                cnt = cnt + 1;
        if(cnt == 3)
            nr = nr + 1;
    }
}

```

```

2. #include <iostream>
#include <cstring>
using namespace std;
int main()
{
    char s[101], t[101], *p;
    cin.getline(s, 101);
    t[0] = '\0';
    p = strtok(s, " ");
    while(p)
    {

```

```

    if(strchr(p, '.') )
    {
        if(p[0] == 'F')
            strcat(t, "fam.");
        else
            if(p[0] == 'G')
                strcat(t, "gen.");
            else
                strcat(t, "spe.");
    }
    else strcat(t, p);
    p = strtok(NULL, " ");
    if(p)
        strcat(t, " ");
}
strcpy(s, t);
cout << s;
return 0;
}

```

3. a. Algoritmul descris citește pe rând temperaturile din fișier, fără a le memora, cu ajutorul variabilei x.

Algoritmul memorează temperatura maximă întâlnită până la pasul curent (maxi), temperatura de la pasul precedent (precedent) și numărul pasului (pas).

Un pas este reprezentativ dacă:

- temperatura curentă este mai mare sau egală cu temperatura maximă întâlnită anterior, valoare memorată în variabila precedent;
- nu este egală cu temperatura de la pasul precedent (pentru a elimina valorile egale aflate pe poziții consecutive).

Dacă sunt îndeplinite aceste condiții, se afișează numărul pasului și se actualizează valoarea maximă.

Algoritmul descris este eficient deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, parcurge șirul o singură dată, fără a memora toate valorile, adică are o complexitate $O(1)$.

b.

```

#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
int x, precedent, maxi, pas = 1;
int main()
{
    f >> x;
    cout << 1 << " ";
    maxi = x;
    precedent = x;
    while(f >> x)
    {
        pas++;
    }
}

```

```

        if(x >= maxi && x != precedent)
        {
            cout << pas << " ";
            maxi = x;
        }
        precedent = x;
    }
    f.close();
    return 0;
}

```

Varianta 6 – Sesiunea specială 2023

Subiectul I.

1.c 2.a 3.d 4.b 5.c

Subiectul II.

1. a. Răspuns corect: 233223222

b. Oricare dintre numerele 6, 7, 8, 9 sunt conform cerinței.

Exemplu de răspuns corect: 6 7

c.

```
#include <iostream>
```

```
using namespace std;
```

```
int main()
```

$$\{$$

```
int n, i, j;
```

```
cin>>n;
```

```
for(i=1; i<=n; i++)
```

```
for(j=1; j<=n; j++)
```

```

if(j<=i)

```

```
cout<<2;
```

else

```
cout<<3;
```

```
return 0;
```

}

d. Înlocuirea primei structuri pentru...execută cu o structură de tip cât timp...execută citește n (număr natural nenul)

$$i \leftarrow 1$$

$\vdash$ cât timp $i \leq n$ execută

pentru $j \leftarrow 1, n$ execută

└dacă $j \leq i$ atunci

serie 2

			altfel
--	--	--	--------

serie 3

```

└ i ← i+1
  └ ──

```

Înlocuirea celei de a doua structuri pentru...execută cu o structură de tip cât timp...execută citește n(număr natural nenul)

```

└ pentru i ← 1, n execută
  └ j ← 1
    └ cât timp j ≤ n execută
      └ dacă j ≤ i atunci
        └ scrie 2
          └ altfel
            └ scrie 3
              └ ──
            └ j ← j+1
              └ ──
          └ ──
        └ ──
      └ ──
    └ ──
  └ ──
└ ──

```

2. Răspuns corect: 4 5 6 7

```

3. for(k=strlen(s)-1; k >= 0; k--)
    if(s[k] == s[0])
        {
            s[k+1] = '\0';
            cout << s << " ";
        }

```

Subiectul III.

```

1. void Putere(int n, int &x, int &p)
{
    int d, k, cn;
    for(d = 2; d * d <= n; d++)
    {
        cn = n;
        k = 0;
        while(cn % d == 0)
        {
            cn = cn / d;
            k = k + 1;
        }
        if(cn == 1)
        {
            x = d;
            p = k;
            return;
        }
    }
    x = n;
    p = 1;
}

```

2. `#include <iostream>`

`using namespace std;`

`int a[21][21], n, i, j;`

`int main()`

`{`

`cin >> n;`

`for(j = 1; j <= n; j++)`

`cin >> a[1][j];`

`for(i = 2; i <= n; i++)`

`{`

`for(j = 1; j <= n; j++)`

`if(j <= n - i + 1)`

`a[i][j] = (a[i - 1][j] + a[i - 1][j + 1]) % 100;`

`else`

`a[i][j] = -1;`

`}`

`for(i = 1; i <= n; i++)`

`{`

`for(j = 1; j <= n; j++)`

`cout << a[i][j] << " ";`

`cout << endl;`

`}`

`return 0;`

`}`

3. a. Algoritmul descris citește pe rând valorile din fișier, cu ajutorul variabilei x, fără a le memora.

Algoritmul memorează valoarea minimă (mini) și valoarea maximă (maxi) dintre valorile citite până la un moment curent.

Pentru fiecare nouă valoare extrasă din fișier (x), dacă acesta modifică valoarea minimă sau valoarea maximă, atunci p-intervalul se modifică; altfel, dacă nu modifică nici minimul, nici maximul, atunci (p-1)-interval este identic cu p-interval, deci valoarea curentă a lui p este o soluție.

Algoritmul memorează prima valoare (prim) și ultima valoare (ultim) a lui p care îndeplinește această proprietate. Dacă nu există astfel de valori, adică prim=0, se afișează mesajul "nu exista".

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, parcurge șirul o singură dată, fără a memora toate valorile, adică are o complexitate $O(1)$.

b.

`#include <iostream>`

`#include <fstream>`

`using namespace std;`

```

ifstream f("bac.in");
int x, p = 1, mini, maxi, prim, ultim;
int main()
{
    f >> x;
    mini = x;
    maxi = x;
    while(f >> x)
    {
        p++;
        if(x < mini)
            mini = x;
        else
            if(x > maxi)
                maxi = x;
            else
            {
                if(prim == 0)
                    prim = p;
                ultim = p;
            }
    }
    if(prim == 0)
        cout << "nu exista";
    else
        cout << prim << " " << ultim;
    f.close();
    return 0;
}

```

Varianta 5 – Sesiunea iunie-iulie 2023

Subiectul I.

1.d 2.c 3.c 4.b 5.a

Subiectul II.

1. a. Răspuns corect: 9752

b. Oricare numere din intervalul [100, 999], cu cifrele în ordine strict descrescătoare sunt conform cerinței.

Exemplu de răspuns corect: 987 321

c.

```

#include <iostream>
using namespace std;
int main()
{
    int x, p, m, c;
    cin >> x;
    p=1; m=-1;
    while(p<=x)

```

```

    {
        c=x/p%10;
        if(c>m)
            {
                m=c;
                p=p*10;
            }
        else
            x=x/(p*10)*p+x%p;
    }
    if(m>=0)
        cout<<x;
    else
        cout<<"nul";
    return 0;
}

```

d.

citește x (număr natural)

 $p \leftarrow 1$; $m \leftarrow -1$

```

dacă  $p \leq x$  atunci
    repetă
         $c \leftarrow [x/p] \% 10$ 
        dacă  $c > m$  atunci
             $m \leftarrow c$ ;  $p \leftarrow p * 10$ 
        altfel
             $x \leftarrow [x/(p * 10)] * p + x \% p$ 
    până când  $p > x$ 
    dacă  $m \geq 0$  atunci scrie x
    altfel scrie "nul"

```

2. Răspuns corect: (mare, râu)
(ocean, ploaie, râu)

3. struct lalea

```

{
    char denumire[21];
    struct
    {
        int nrFire, pretFir;
    } stoc;
} f[10];

```

Subiectul III.

```

1. int abundant(int n)
{
    int i, j, sn=0, si;
    for(i = 1; i <= n; i++)
        if(n % i == 0)
            sn = sn + i;

    for(i = 1; i < n; i++)
    {
        si = 0;
        for(j = 1; j <= i; j++)
            if(i % j == 0)
                si = si + j;
        if(sn * i <= si * n)
            return 0;
    }
    return 1;
}

```

```

2. #include <iostream>
using namespace std;
int a[1001][51], ns, np, i, j, maxi, poz;
int main()
{
    cin >> ns >> np;
    for(i = 1; i <= ns; i++)
        for(j = 1; j <= np; j++)
            cin >> a[i][j];
    for(i = 1; i <= ns; i++)
    {
        maxi = a[i][1];
        poz = 1;
        for(j = 2; j <= np; j++)
            if(a[i][j] > maxi)
            {
                maxi = a[i][j];
                poz = j;
            }
        cout << i << ":" << poz << " ";
    }
    return 0;
}

```

3. a. Algoritmul descris citește pe rând numerele din fișier, folosind variabila x.
 Algoritmul construiește doi vectori de frecvență: p, respectiv s, care au cel mult 100 de elemente, numere naturale, unde:
- p memorează de câte ori fiecare număr de două cifre apare ca prefix
 - s memorează de câte ori fiecare număr de două cifre apare ca sufix
- Pentru fiecare număr x extras din fișier, algoritmul extrage sufixul de două cifre cu formula $x \% 100$, iar prefixul de două cifre se obține prin eliminarea cifrelor de la dreapta

până când numărul rămâne de două cifre (pentru determinarea prefixului lui x, algoritmul păstrează o copie a lui x în variabila cx).

Apoi, algoritmul parcurge valorile de la 10 la 99, adică numerele de două cifre, și se numără acele valori pentru care numărul de apariții ca prefix este egal cu numărul de apariții ca suffix, adică $p[i]=s[i]$, pentru orice $i=10, 99$.

Algoritmul parcurge fișierul o singură dată și utilizează două tablouri de dimensiune maximum 100 de elemente.

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei.

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
int p[100] , s[100] , x, cx, i, nr;
int main()
{
    while(f >> x)
    {
        s[n % 100] = s[n % 100] + 1;
        cx = x;
        while(cx > 99)
            cx = cx / 10;
        p[cx] = p[cx] + 1;
    }
    for(i = 10; i <= 99; i++)
        if(p[i] == s[i])
            nr++;
    cout << nr;
    f.close();
    return 0;
}
```

Varianta 7 – Sesiunea august 2023

Subiectul I.

1.b 2.c 3.a 4.b 5.d

Subiectul II.

1. a. Răspuns corect: 2

b. O soluție corectă este un șir de forma x, 0, y, 0, unde x și y sunt numere din intervalul $[0,9]$.

Exemplu de răspuns corect: 3 0 7 0

c.

```
#include <iostream>
using namespace std;
int main()
{
    int n, x, a, b, p, c;
    cin >> n;
```

```

x=0;
while(n>0)
{
    cin>>a>>b;
    c=a; p=1;
    while(a>9)
    {
        a=a/10;
        p=p*10;
    }
    a=a*p+b;
    if(a!=c)
        x=x+1;
    n=n-1;
}
cout<<x;
return 0;
}

```

d.

citește n (număr natural nenul)

x←0

```

┌ pentru i←n,1,-1 executa
│   citește a,b (numere naturale)
│   c←a; p←1
│   ┌ cât timp a>9 execută
│   │ a←[a/10]; p←p*10
│   │ ──┐
│   │   a←a*p+b
│   │   ┌ dacă a≠c atunci
│   │   │ x←x+1
│   │   └ ──┐
│   └ ──┐
└ ──┐
    scrie x

```

2. Răspuns corect: 2 3

```

3. if(p.pret<100)
    a=p.denumire[0];
else
    a='*';

```

Subiectul III.

```

1. void DNPI(int n)
{
    int d, i, prim;
    for(d = 1; d <= n; d++)
        if(n % d == 0 && d % 2 == 1)
        {
            if(d == 1)
                cout << d << " ";

```

```

else
{
    prim = 1;

    for(i = 2; i * i <= d; i++)
        if(d % i == 0)
            prim = 0;
    if(prim == 0)
        cout << d << " ";
}
}
}

```

```

2. #include <iostream>
using namespace std;
int a[21][21], m, n, i, j, s, maxi;
int main()
{
    cin >> m >> n;
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j++)
            cin >> a[i][j];
    for(i = 1; i < m; i++)
        for(j = 1; j < n; j++)
        {
            s = a[i][j] + a[i][j + 1] + a[i + 1][j] + a[i + 1][j + 1];
            if(s > maxi)
                maxi = s;
        }
    cout << maxi;
    return 0;
}

```

3. a. Algoritmul descris citește inițial dimensiunile celor două șiruri, folosind variabilele m , respectiv n . Apoi, citește pe rând cele două șiruri și numără de câte ori apare fiecare număr natural de două cifre în fiecare dintre cele două șiruri. Astfel, algoritmul folosește doi vectori de frecvență, de dimensiune maximum 100 de elemente, fa pentru primul șir de numere, respectiv fb pentru al doilea șir de numere.

Din punct de vedere matematic, pentru fiecare valoare de la 0 la 99 se poate forma un număr maxim de perechi egal cu minimul dintre numărul de apariții în cele două șiruri. Cu ajutorul variabilei s se însumează toate aceste minime, iar la final se afișează suma.

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(m+n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei.

b.

```

#include <iostream>
#include <fstream>
using namespace std;
ifstream f("date.in");
int fa[100], fb[100], m, n, x, i, s;
int main()

```

```

{
    f >> m >> n;
    for(i = 1; i <= m; i++)
    {
        f >> x;
        fa[x] = fa[x] + 1;
    }
    for(i = 1; i <= n; i++)
    {
        f >> x;
        fb[x] = fb[x] + 1;
    }
    for(i = 0; i <= 99; i++)
    {
        if(fa[i] < fb[i])
            s = s + fa[i];
        else
            s = s + fb[i];
    }
    cout << s;
    f.close();
    return 0;
}

```

Varianta – Simulare 2024

Subiectul I.

1.c 2.b 3.b 4.d 5.a

Subiectul II.

1. a. Răspuns corect: 37

b. Oricare dintre numerele 16, 32 sau 64 sunt conform cerinței.

Exemplu de răspuns corect: 16 32

c.

```

#include <iostream>
using namespace std;
int main()
{
    int n, s, d, p;
    cin >> n;
    s = 0;
    d = 2;
    while(d * d <= n)
    {
        p = 0;
        while(n % d == 0)
        {
            n = n / d;
            p = 1;
        }
        s = s + d * p;
    }
}

```

```

        d=d+1;
    }
    if(n!=1)
        s=s+n;
    cout<<s;
    return 0;
}

```

d.

citește n (număr natural nenul)

 $s \leftarrow 0$; $d \leftarrow 2$ cât timp $d \cdot d \leq n$ execută

```

    p ← 0
    dacă  $n \% d = 0$  execută
        repetă
             $n \leftarrow [n/d]$ ;  $p \leftarrow 1$ 
        până când  $n \% d \neq 0$ 
    s ← s + d * p; d ← d + 1

```

daca $n \neq 1$ atunci $s \leftarrow s + n$

scrie s

2. Răspuns corect: 1 3 8 9

```

3. for(i=0; i<5; i++)
    for(j=0; j<5; j++)
        if(i==0 || j==0)
            a[i][j]=0;
        else
            if(i<j)
                a[i][j]=2*i;
            else
                a[i][j]=2*j;

```

Subiectul III.

```

1. int Impare(int n)
{
    int c, p = 1, nr = 0, ok = 0;
    while(n != 0)
    {
        c = n % 10;
        if(c % 2 == 1)
        {
            nr = c * p + nr;
            p = p * 10;
            ok = 1;
        }
        nr = c * p + nr;
        p = p * 10;
    }
}

```

```

        n = n / 10;
    }
    if(ok == 0)
        return -1;

    return nr;
}

```

```

2. #include <iostream>
    #include <cstring>
    using namespace std;
    char v[101][11], s[101], t[101], *p;
    int n, i, ok = 1;
    int main()
    {
        cin >> n;
        cin.get();
        for(i = 1; i <= n; i++)
            cin.getline(v[i], 11);
        cin.getline(s, 101);
        t[0] = '\0';
        p = strtok(s, " ");
        while(p)
        {
            if(p[0] == '*')
            {
                ok = 0;
                for(i = 1; i <= n; i++)
                    if(strlen(v[i]) == strlen(p))
                    {
                        strcat(t, v[i]);
                        ok = 1;
                        break;
                    }
            }
            if(ok == 0)
            {
                cout << "imposibil";
                return 0;
            }
        }
        else
            strcat(t, p);
        p = strtok(NULL, " ");
        if(p)
            strcat(t, " ");
    }
    strcpy(s, t);
    cout << s;
    return 0;
}

```

3. a. Algoritmul descris citește pe rând numerele din fișier, cu ajutorul variabilei x , și construiește un vector de frecvență, cu dimensiune maximă 1000, în care se memorează numărul de apariții al fiecărei valori distincte din șir.

După extragerea tuturor numerelor din fișier, se parcurge tabloul de frecvență, și pentru fiecare valoare care apare în șir, cel puțin o dată, se verifică dacă numărul de apariții este mai mare decât valoarea respectivă sau dacă numărul de apariții al valorii are paritate diferită față de valoare, caz în care se oprește parcurgerea vectorului de frecvență (variabila ok ia valoarea 0).

Dacă una dintre condiții nu este îndeplinită, adică variabila ok are valoarea 0, atunci se afișează mesajul "NU". În caz contrar, după verificarea tuturor valorilor din vectorul de frecvență și variabila ok nu și-a modificat valoarea inițială, adică 1, atunci se afișează mesajul "DA".

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei.

b.

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int f[1001], x, i, ok=1;
int main()
{
    while(f >> x)
        f[x] = f[x] + 1;
    for(i = 1; i <= 1000 && ok==1; i++)
        if(f[i] != 0)
            if(f[i] > i || f[i] % 2 != i % 2)
                ok=0;
    if(ok==1) cout << "DA";
    else cout << "NU";
    f.close();
    return 0;
}
```

Varianta 4 – Sesiunea Specială 2024

Subiectul I.

1.a 2.b 3.d 4.d 5.c

Subiectul II.

1. a. Răspuns corect: 1 1 3 1 5

b. Oricare dintre numerele din intervalul $[16, 31]$ sunt soluții corecte.

Exemplu de răspuns corect: 17 18

c.


```

c=0;
for(i=1; i<=10; i++)
{
    cin>>s;
    if(strlen(s)==2)
        if((strchr("aeiou",s[0]) && !strchr("aeiou",s[1])) || (!strchr("aeiou",s[0])
&& strchr("aeiou",s[1]))) )
            c=1;
}

```

Subiectul III.

```

1. int fulg(int n)
{
    int f[5] = {0}, c, nr = 0, i;
    while(n != 0)
    {
        c = n % 10;
        if(c < 1 || c > 4)
            return 0;
        f[c] = f[c] + 1;
        nr = nr + 1;
        n = n / 10;
    }
    if(nr != 9)
        return 0;
    for(i = 1; i <= 4; i++)
        if(f[i] == 0)
            return 0;
    return 1;
}

2. #include <iostream>
using namespace std;
int a[50][50], m, n, i, j, p, x, y;
int main()
{
    cin >> m >> n;
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j++)
        {
            p = 1;
            x = j;
            while(x != 0)
            {
                p = p * 10;
                x = x / 10;
            }
            if(i * p + j < j * 10 + i)
                a[i][j] = i * p + j;
        }
}

```

```

        else
            a[i][j] = j * 10 + i;
    }
    for(i = 1; i <= m; i++)
    {
        for(j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
    return 0;
}

```

3. a. Algoritmul descris citește pe rând numerele din fișier, cu ajutorul variabilei x.

Algoritmul memorează cea mai mare cifră întâlnită până la un moment curent în variabila maxc, dar și primul număr (prim), respectiv ultimul număr (ultim) din șirul parcurs până în acel moment, numere care conțin această cifră maximă.

De asemenea, pentru fiecare număr x citit din fișier:

- se determină cifra sa maximă, mc (cifra maximă din numărul x se obține parcurgând o copie cx a lui x).
- dacă mc > maxc, atunci se actualizează maxc, iar primul și ultimul număr devin numărul curent; în caz contrar, se verifică dacă mc = maxc și dacă este adevărat, atunci se actualizează doar ultimul număr.

După extragerea tuturor numerelor din fișier, dacă primul și ultimul număr sunt diferite sau dacă aceeași valoare a apărut de cel puțin două ori, atunci se afișează cele două numere; în caz contrar se afișează mesajul "nu exista".

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, adică are o complexitate $O(1)$.

b.

```

#include <iostream>
#include <fstream>
using namespace std;
ifstream f("numere.in");
int n, x, c, mc, maxc = -1, prim, ultim, nr;
int main()
{
    while(f >> x)
    {
        cx = x;
        mc = 0;
        while(cx != 0)
        {
            c = cx % 10;
            if(c > mc)
                mc = c;
            cx = cx / 10;
        }
        if(mc > maxc)
        {

```

```

    maxc = mc;
    prim = x;
    ultim = x;
    nr = 1;
}
else
    if(mc == maxc)
    {
        ultim = x;
        nr++;
    }
}
if(nr >= 2)
    cout << prim << " " << ultim;
else
    cout << "nu exista";
f.close();
return 0;
}

```

Varianta 3 – Sesiunea iunie-iulie 2024

Subiectul I.

1.a 2.d 3.c 4.d 5.b

Subiectul II.

1. a. Răspuns corect: 9

b. Sunt soluții corecte oricare două numere aflate în reuniunea intervalelor de forma $[2 \cdot 3^k, 3^{k+1})$, unde k este număr natural, $k \in \{1, 2, 3, 4, 5\}$.

Exemplu de răspuns corect: 7 8, deoarece aparține intervalului $[6, 9)$

c.

```

#include <iostream>
using namespace std;
int main( )
{
    int n, p, i, x;
    cin >> n;
    p = 1;
    for(i = 1; i <= n; i++)
    {
        cin >> x;
        do
        {
            x = x / 3;
        } while(x > 3);
        if(x != 0)
            p = p * x;
    }
    cout << p;
    return 0;
}

```

}

d.

citește n (număr natural nenul)

p ← 1

i ← 1

cât timp $i \leq n$ execută

citește x (număr natural)

repetă

 $x \leftarrow \lfloor x/3 \rfloor$ ■ până când $x \leq 3$ dacă $x \neq 0$ atunci $p \leftarrow p * x$

■

 $i \leftarrow i + 1$

■

scrie p

2. Răspuns corect: 2 3 1 7

```

3. for(i=0; i<9; i++)
    for(j=0; j<9; j++)
    {
        a[i][j]=2;
        if(i+j<=3 || i+j>=13)
            a[i][j]=4;
    }

```

Subiectul III.

1. int majImp(int a, int b)

{

int i, d, si, sp;

for(i = a; i <= b; i++)

{

si = 0;

sp = 0;

for(d = 2; d < n; d++)

if(n % d == 0)

{

if(d % 2 == 0)

sp = sp + d;

else

si = si + d;

}

if(si > sp)

return i;

}

return 0;

}

2. #include <iostream>

```

#include <cstring>
using namespace std;
int main()
{
    char s[101], *p;
    int n, ok=1;

    cin.getline(s, 101);
    p = strtok(s, " ");
    n = strlen(p);
    p = strtok(NULL, " ");
    while(p && ok)
    {
        if(strlen(p) != n)
            ok=0;
        p = strtok(NULL, " ");
    }
    if(ok==1)
        cout << "DA " << n;
    else
        cout << "NU";
    return 0;
}

```

3. a. Algoritmul descris citește pe rând valorile din fișier, fără a le memora.

Algoritmul memorează lungimea scării curente în variabila lg , precum și lungimea maximă în variabila $lgmax$. Folosind două variabile x , respectiv y , algoritmul păstrează în fiecare moment două valori aflate în fișier pe poziții consecutive. Dacă $y-x=1$, atunci scara continuă și se mărește lungimea acesteia ($lg=lg+1$). În caz contrar, începe o nouă scară de lungime 1 ($lg=1$).

Dacă se obține o scară mai lungă decât cea maximă ($lg>lgmax$), atunci se memorează ultima valoare a acesteia în variabila $ultim$.

După extragerea tuturor numerelor din fișier, dacă lungimea maximă este cel puțin 2, se afișează toate valorile din scara respectivă, de la $ultim-lgmax+1$ până la $ultim$; altfel se afișează mesajul "nu exista".

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, fiecare valoare din fișier este prelucrată în momentul extragerii ei. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, adică are o complexitate $O(1)$.

b.

```

#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
int x, y, lg = 1, lgmax = 1, ultim, i;
int main()
{
    f >> x
    while(f >> y)
    {

```

```

    if(y == x + 1)
        lg = lg + 1;
    else
        lg = 1;
    if(lg > lgmax)
    {
        lgmax = lg;
        ultim = y;
    }
    x = y;
}
if(lgmax == 1)
    cout << "nu exista";
else
    for(i = ultim - lgmax + 1; i <= ultim; i++)
        cout << i << " ";
f.close();
return 0;
}

```

Varianta 8 – Sesiunea august 2024

Subiectul I.

1.b 2.d 3.a 4.b 5.c

Subiectul II.

1. a. Răspuns corect: 43141

b. Oricare două numere dintre 111, 1111, 11011 sunt conform cerinței.

Exemplu de răspuns corect: 111 1111

c.

```

#include <iostream>
using namespace std;
int main()
{
    int n, x, y;
    cin >> n;
    x = -1; y = -1;
    while(n > 9)
    {
        if(x == -1)
            x = n % 100;
        else
            y = n % 100;
        n = n / 10;
    }
    if(x < y)
        n = (n * 100 + x) * 100 + y;
    else
        n = (n * 100 + y) * 100 + x;
}

```

```

    cout<<n;
    return 0;
}

```

d.

citește n (număr natural nenul)

 $x \leftarrow -1; y \leftarrow -1$

```

dacă n>9 atunci
    repetă
        dacă x=-1 atunci  $x \leftarrow n \% 100$ 
        altfel  $y \leftarrow n \% 100$ 
        ■
         $n \leftarrow \lfloor n/100 \rfloor$ 
        ■ până când  $n \leq 9$ 
    ■
    dacă  $x < y$  atunci  $n \leftarrow (n*100+x)*100$ 
    altfel  $n \leftarrow (n*100+y)*100+x$ 
    ■
scrie n

```

2. Răspuns corect: (crin, gerbera, iris), (crin, iris, orhidee)

3.

```

if ((m.nota1=='C' && m.nota2=='E') || (m.nota1=='D' && m.nota2=='F') || (m.nota1=='E'
&& m.nota2=='G') || (m.nota1=='F' && m.nota2=='A') || (m.nota1=='G' && m.nota2=='B'))
    cout<<"TERTA";
else
    cout<<"NU";

```

Subiectul III.

1. int prim(int n)

```

{
    int d;
    if(n < 2)
        return 0;
    for(d = 2; d * d <= n; d++)
        if(n % d == 0)
            return 0;
    return 1;
}

int moderat(int n)
{
    int d, p1, p2, i;
    for(d = 2; d * d <= n; d++)
        if(n % d == 0)
        {
            p1 = d;
            p2 = n / d;
            if(prim(p1) && prim(p2))
            {
                i = p1 + 1;
            }
        }
    return i;
}

```

```

        while(!prim(i))
            i++;

        if(i == p2)
            return 1;
    }
}
return 0;
}

```

```

2. #include <iostream>
using namespace std;
int a[100][100], m, n, i, j, mini;
int main()
{
    cin >> m >> n;
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j++)
            cin >> a[i][j];
    for(i = 1; i <= m; i = i + 2)
    {
        mini = a[i + 1][1];
        for(j = 2; j <= n; j++)
            if(a[i + 1][j] < mini)
                mini = a[i + 1][j];
        for(j = 1; j <= n; j++)
            if(a[i][j] >= mini)
                a[i][j] = mini - 1;
    }
    for(i = 1; i <= m; i++)
    {
        for(j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
    return 0;
}

```

3. a. Algoritmul descris citește de la tastatură datele de intrare în variabilele x, y, respectiv z, ce reprezintă trei termeni aflați pe poziții consecutive în cadrul șirului.

Deoarece se cer termenii șirului, afișați în ordine descrescătoare, se afișează mai întâi z, apoi se determină succesiv termenii anteriori, folosind variabila t, prin relația obținută din formula de recurență:

$$t = \frac{4 \cdot y - 3 \cdot x - z}{2}$$

Astfel, cunoscând trei termeni consecutivi: x, y și z, se calculează t, termenul precedent, apoi variabilele își schimbă valorile astfel:

```

z ← y
y ← x
x ← t

```

Algoritmul se încheie atunci când x=y=0.

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate O(n)), parcurge fișierul o singură dată, fiecare valoare

din fișier este prelucrată în momentul extragerii ei. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, adică are o complexitate $O(1)$.

b.

```
#include <iostream>
#include <fstream>
using namespace std;
ofstream g("bac.txt");
int x, y, z, t;
int main()
{
    cin >> x >> y >> z;
    g << z << " ";
    while(x != 0 || y != 0)
    {
        g << y << " ";
        t = (4 * y - 3 * x - z) / 2;
        z = y;
        y = x;
        x = t;
    }
    g << 0 << " " << 0;
    g.close();
    return 0;
}
```

Varianta – Simulare 2025

Subiectul I.

1.c 2.d 3.a 4.b 5.a

Subiectul II.

1. a. Răspuns corect: 25 24 *22 21 *19 18 *

b. Oricare două numere dintre 3, 10, 11 sunt conform cerinței.

Exemplu de răspuns corect: 10 11

c.

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, i, k;
    cin >> m >> n;
    if(m < n)
    {
        n = n + m;
        m = n - m;
        n = n - m;
    }
}
```



```

1. void harsad(int k, int &n)
{
    int s, x;
    while(k >= 1)
    {
        s = 0;
        x = k;
        while(x > 0)
        {
            s = s + x % 10;
            x = x / 10;
        }
        if(k % s == 0)
        {
            n = k;
            return;
        }
        k--;
    }
    n = 0;
}

```

```

2. #include <iostream>
#include <cstring>
using namespace std;
char s[100][21];
int v[100], n, i, j, nr;
int main()
{
    cin >> n;
    for(i = 0; i < n; i++)
    {
        cin >> s[i];
        v[i] = 0;
        for(j = 0; j < strlen(s[i]); j++)
            if(s[i][j] == 'a' || s[i][j] == 'e' || s[i][j] == 'i' || s[i][j] == 'o' || s[i][j] == 'u')
                v[i]++;
    }
    for(i = 0; i < n - 1; i++)
        if(v[i] == v[n - 1] && strcmp(s[i], s[n - 1]) != 0)
        {
            cout << s[i] << " ";
            nr++;
        }
    if(nr == 0)
        cout << "nu exista";
    return 0;
}

```

1. a. Algoritmul descris citește, mai întâi, primele două valori din fișier în variabilele nc, respectiv np. Apoi, pe rând, citește etichetele cerceilor folosind variabila x și, pentru fiecare x, se rețin într-un tablou caracteristic (format doar din valori de 0 și 1) de dimensiune

maximă 100 de elemente perechile formate din primele două cifre, fără a ține cont de ordine, astfel:

- se determină prima cifră a lui x, adică c1
- se determină a doua cifră a lui x, adică c2
- dacă $c1 > c2$, atunci se interschimbă valorile, astfel încât perechea să fie memorată în ordine crescătoare
- se marchează perechea în tabloul ap, adică $a[c1][c2]=1$.

În mod asemănător, se citesc și prelucreează numerele de pe rândul al treilea din fișier, adică etichetele pandantelor. Pentru fiecare etichetă se determină aceeași pereche de cifre $a[c1][c2]$. Variabilă ok, care inițial avea valoarea 0, devine 1, dacă se verifică că perechea de cifre există deja în tabloul caracteristic. Astfel, dacă se citesc toate pandantivele fără a găsi o pereche comună, variabilă ok rămâne neschimbată, adică cu valoarea 0.

După extragerea tuturor numerelor din fișier, algoritmul afișează mesajul "DA" sau "NU", în funcție de valoarea variabilei ok, adică se afișează mesajul "DA", dacă $ok=1$, sau se afișează mesajul "NU", dacă $ok=0$.

Algoritmul descris este eficient din punct de vedere al timpului de executare deoarece este liniar (are complexitate $O(nc+np)$), parcurge fișierul o singură dată, moment în care se realizează și prelucrarea valorilor extras.

b.

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bijuterii.in");
int ap[10][10], nc, np, x, c1, c2, aux, ok;
int main()
{
    f >> nc >> np;
    for(int i = 1; i <= nc; i++)
    {
        f >> x;
        while(x >= 100)
            x = x / 10;
        c1 = x / 10;
        c2 = x % 10;
        if(c1 > c2)
        {
            aux = c1;
            c1 = c2;
            c2 = aux;
        }
        ap[c1][c2] = 1;
    }
    for(int i = 1; i <= np; i++)
    {
        f >> x;
        while(x >= 100)
            x = x / 10;
        c1 = x / 10;
        c2 = x % 10;
        if(c1 > c2)
        {
```

```

        aux = c1;
        c1 = c2;
        c2 = aux;
    }
    if(ap[c1][c2] == 1)
        ok=1;
    }
    if (ok==0)
        cout << "NU";
    else
        cout<<"DA";
    f.close();
    return 0;
}

```

Varianta 6 – Sesiunea specială 2025

Subiectul I.

1.d 2.c 3.a 4.b 5.c

Subiectul II.

1. a. Răspuns corect: 88 77

b. Oricare două numere naturale din intervalul [2222, 3332] sunt conform cerinței.

Exemplu de răspuns corect: 2222 2223

c.

```

#include <iostream>
using namespace std;
int main()
{
    int m, n, i, x, c;
    cin>>m>>n;
    for(i=n; i>=m; i--)
    {
        x=i;
        c=x%10;
        do
        {
            x=x/10;
        }while(x%10==c);
        if(x==0)
            cout<<i<<" ";
    }
    return 0;
}

```

d.

citește m, n (numere naturale nenule, $m \leq n$)

$i \leftarrow n$

```

cât timp i ≥ m execută
x ← i
c ← x % 10
    repetă
        x ← [x/10]
        ■ până când x % 10 ≠ c
    dacă x = 0 atunci
        scrie i, ' '
        ■
    i ← i - 1
    ■

```

2. Oricare două numere dintre numerele 1,3,7 sunt conform cerinței.
Exemplu de răspuns corect: 1 3
3. `if(abs(d.A.x-d.B.x)==abs(d.A.y-d.B.y))`
`cout<<"DA";`
`else`
`cout<<"NU";`

Subiectul III.

```

1. int diviz(int n)
{
    int d = 2, p = 1, nr;
    while(n > 1)
    {
        nr = 0;
        while(n % d == 0)
        {
            nr++;
            n = n / d;
        }
        if(nr % 2 == 1)
            nr--;
        for(int i = 1; i <= nr; i++)
            p = p * d;
        d++;
        if(d * d > n && n > 1)
            d = n;
    }
    return p;
}

```

```

2. #include <iostream>
#include <cstring>
using namespace std;

```

```

char s[101], t[201], *p;
int i, j = 0;
int main()
{
    cin.getline(s, 101);
    t[0]='\0';
    p = strtok(s, " ");
    while(p)
    {
        strcat(t, p);
        p = strtok(NULL, " ");
        if(p )
            strcat(t, " - ");
    }
    strcpy(s, t);
    cout << s;
    return 0;
}

```

3. a. Algoritmul descris citește fiecare triplet de numere în variabilele a, b, respectiv c.

Se determină maximul dintre a, b și c și se memorează în variabila ip, adică ipotenuza triunghiului. Celelalte două valori, catetele triunghiului, se memorează în variabilele auxiliare x, respectiv y.

Algoritmul verifică dacă triunghiul este dreptunghic folosind relația (teorema lui Pitagora): $x^2 + y^2 = ip^2$.

Dacă relația este adevărată, atunci se mărește frecvența ipotenuzei într-un tablou de frecvență fr de dimensiune maximă 100 de elemente. În același timp, se memorează valoarea maximă a frecvenței.

După extragerea tuturor valorilor din fișier se afișează frecvența maximă.

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(n)$, unde n reprezintă numărul de triplete din fișier), parcurge fișierul o singură dată, moment în care se realizează și prelucrarea valorilor extrase.

b.

```

#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
int a, b, c, x, y, ip, fr[101], maxi;
int main()
{
    while(f >> a >> b >> c)
    {
        ip = a;
        x = b;
        y = c;
        if(b > ip)
        {
            ip = b;

```

```

        x = a;
        y=c;
    }
    if(c > ip)
    {
        ip = c;
        x = a;
        y = b;
    }
    if(x * x + y * y == ip * ip)
    {
        fr[ip]++;
        if(fr[ip] > maxi)
            maxi = fr[ip];
    }
}
cout << maxi;
f.close();
return 0;
}

```

Varianta 6 – Sesiunea iunie-iulie 2025

Subiectul I.

1.d 2.a 3.b 4.c 5.b

Subiectul II.

1. a. Răspuns corect: 9

b. Oricare două numere naturale din intervalul [17, 25] sunt conform cerinței.

Exemplu de răspuns corect: 18 19

c.

```

#include <iostream>
using namespace std;
int main()
{
    int m, n, nr, i, x;
    cin >> m >> n;
    nr=0; i=m;
    do
    {
        x=1;
        while(x*x<i)
        {
            x=x+1;
        }
        if(x*x==i)
            nr=i;
    }
}

```

```

        else
            i=i+1;

    }while(i<=n && nr==0);
    cout<<nr;
    return 0;
}

```

d.

citește m, n (numere naturale nenule, $m \leq n$) $nr \leftarrow 0$; $i \leftarrow m$

```

repetă
x ← 1
    dacă  $x * x < i$  atunci
        repetă
            x ← x + 1
        ■ până când  $x * x \geq i$ 
    ■
    dacă  $x * x = i$  atunci  $nr \leftarrow i$ 
    altfel  $i \leftarrow i + 1$ 
    ■
■ până când  $i > n$  sau  $nr \neq 0$ 
scrie nr

```

2. Mulțimea nodurilor: {1, 2, 3}

Mulțimea muchiilor: {(1,2), (1, 3), (2, 3)}

Explicație: S-a ales un subgraf format din nodurile 1, 2 și 3, unite prin muchiile [1,2], [1,3] și [2,3]. Subgraful este un graf eulerian, deoarece este conex, nu are noduri izolate și fiecare nod are gradul par.

3. struct prajitura

```

{
    int cod, informatii[3];
    float pret;
}p;

```

Subiectul III.

1. int ascendent(int n, int x, int y)

```

{
    int s = 0, ok, i, c, ci;
    c = n % 10;
    for (i= x; i <= y; i++)
    {
        ok = 1;
        if(i == 0)
        {
            if(0 < c)
                ok = 0;
        }
        else
        {

```

```

        ci = i;
        while(ci > 0 && ok == 1)
        {
            if(ci % 10 < c)
                ok = 0;
            ci = ci / 10;
        }
    }
    if(ok == 1)
        s = s + i;
}
return s;
}

```

```

2. #include <iostream>
#include <cstring>
using namespace std;
char s[201], t[201], *p;
int ok, lg, k, i, l;
int main()
{
    cin.getline(s, 201);
    t[0] = '\0';
    p = strtok(s, " ");
    while(p)
    {
        lg = strlen(p);
        if(lg % 2 == 0)
        {
            ok = 1;
            k = lg / 2;
            for(i = k; i < lg; i++)
            {
                l = strlen(t);
                t[l] = p[i];
                t[l + 1] = '\0';
            }
            for(i = 0; i < k; i++)
            {
                l = strlen(t);
                t[l] = p[i];
                t[l + 1] = '\0';
            }
        }
        else
            strcat(t, p);
        p = strtok(NULL, " ");
        if(p)
            strcat(t, " ");
    }
    if(ok == 0)
        cout << "nu exista";
}

```

```

else
{
    strcpy(s, t);
    cout << s;
}
return 0;
}

```

1. a. Algoritmul descris citește primul interval, care reprezintă timpul disponibil al tânărului în variabilele a și b . Apoi, cu ajutorul variabilelor x și y se citesc, pe rând, intervalele fiecărui muzeu. Pentru fiecare muzeu se determină intervalul comun dintre cele două intervale, adică se realizează intersecția intervalelor $[a,b]$ și $[x,y]$, astfel:

- începutul intervalului comun este $\max(a, x)$;
- sfârșitul intervalului comun este $\min(b, y)$.

Dacă lungimea intervalului comun este de cel puțin o oră, adică:

$$\min(b, y) - \max(a, x) \geq 1$$

atunci muzeul este convenabil. Se mărește numărul muzeelor convenabile, adică $nr++$ și se memorează numărul de ordine al acestui muzeu în variabila `ultim`.

La final se afișează numărul muzeelor convenabile și numărul de ordine al ultimului muzeu convenabil.

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, moment în care se realizează și prelucrarea valorilor extrase. Algoritmul este eficient din punct de vedere al memoriei utilizate, deoarece nu utilizează structuri de tip tablou, adică are o complexitate $O(1)$.

```

b. #include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int a, b, x, y, inceput, sfarsit, nr, ultim, poz;
int main()
{
    f >> a >> b;
    while(f >> x >> y)
    {
        poz++;
        if(a > x)
            inceput = a;
        else
            inceput = x;
        if(b < y)
            sfarsit = b;
        else
            sfarsit = y;
        if(sfarsit - inceput >= 1)
        {
            nr++;
            ultim = poz;
        }
    }
    cout << nr << " " << ultim;
}

```

```

        f.close();
        return 0;
    }

```

Varianta 4 – Sesiunea august 2025

Subiectul I.

1.a 2.d 3.c 4.b 5.b

Subiectul II.

1. a. Răspuns corect: NU 2 5

b. Oricare două numere naturale din intervalul $[10, 99]$ care îndeplinesc proprietatea că au suma cifrelor egală cu 9 sau cu 18 sunt conform cerinței.

Exemplu de răspuns corect: 18 27

c.

```

#include <iostream>
using namespace std;
int main()
{
    int x, y, cx, cy;
    cin >> x >> y;
    cx = 0; cy = 0;
    while (x >= 10 || y >= 10)
    {
        while (x + y != 0)
        {
            cx = cx + x % 10; x = x / 10;
            cy = cy + y % 10; y = y / 10;
        }
        x = cx; cx = 0; y = cy; cy = 0;
    }
    if (x == y)
        cout << "DA " << x;
    else
        cout << "NU " << x << " " << y;
    return 0;
}

```

d.

citește x, y (numere naturale distincte)

$cx \leftarrow 0; cy \leftarrow 0$

dacă $x \geq 10$ sau $y \geq 10$ atunci

 repetă

 cât timp $x + y \neq 0$ execută

$cx \leftarrow cx + x \% 10; x \leftarrow [x / 10]$

$cy \leftarrow cy + y \% 10; y \leftarrow [y / 10]$

 ■

$x \leftarrow cx; cx \leftarrow 0; y \leftarrow cy; cy \leftarrow 0$

 ■ până când $x < 10$ și $y < 10$

■

```

┌dacă x=y atunci scrie "DA ",x
└altfel scrie "NU ",x," ",y
■

```

2. $f(3, 9)=3$
 $f(1, 1000)=25$

3. struct bijuterie
 {
 struct
 {
 char denumire[31];
 int pret;
 } metal;
 int greutate;
 }b;

Subiectul III.

```

1. void teren(int x, int y)
{
    int aria = x * y, a, ok = 0, i;
    for(i = 2; i <= aria; i = i + 2)
        if(aria % i == 0)
        {
            a = aria / i;
            if(i < a)
            {
                cout << i << " parcele de arie " << a << endl;
                ok = 1;
            }
        }
    if(ok == 0)
        cout << "nu exista";
}

```

```

2. #include <iostream>
using namespace std;
int a[101][101], m, n, i, j, mini, c;
int main()
{
    cin >> m >> n;
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j++)
            cin >> a[i][j];
    mini = a[1][1];
    c = 1;
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j++)
            if(a[i][j] < mini)
            {
                mini = a[i][j];
            }
}

```

```

        c = j;
    }
    for(i = 1; i <= m; i++)
        a[i][c] = a[m][n];

    for(i = 1; i <= m; i++)
    {
        for(j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
    return 0;
}

```

1. a. Algoritmul descris citește, mai întâi, valorile minZ și minP, apoi se parcurg pe rând stocurile zilnice, folosind variabila x.

Algoritmul construiește secvențe maxime de zile consecutive pentru care stocul fiecărei zi este cel puțin minZ, astfel:

- se memorează prima zi în variabila prim
- se calculează suma stocurilor în variabila s
- se memorează ultima zi în variabila ultim.

La întâlnirea unei zile cu stoc mai mic decât minZ sau la sfârșitul fișierului, secvența se încheie. Dacă secvența are cel puțin două zile și suma este cel puțin minP, se afișează prima zi, ultima zi și suma.

Dacă nu s-a afișat nicio perioadă, adică ok=0, se afișează mesajul ”nu exista”.

Algoritmul descris este eficient din punct de vedere al timpului de executare, deoarece este liniar (are complexitate $O(n)$), parcurge fișierul o singură dată, moment în care se realizează și prelucrarea valorilor extrase, nu se folosesc structuri de tip tablou.

b.

```

#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int minZ, minP, x, nr, prim, ultim, s, lg, ok;
int main()
{
    f >> minZ >> minP;
    while(f >> x)
    {
        nr++;
        if(x >= minZ)
        {
            if(lg == 0)
            {
                prim = nr;
                s = 0;
            }

            s = s + x;
            ultim = nr;
            lg++;
        }
    }
}

```

```
    }
    else
    {
        if(lg >= 2 && s >= minP)
        {
            cout << prim << " " << ultim << " " << s << endl;
            ok = 1;
        }
        lg = 0;
        s = 0;
    }
}
if(lg >= 2 && s >= minP)
{
    cout << prim << " " << ultim << " " << s << endl;
    ok = 1;
}
if(ok == 0)
    cout << "nu exista";
f.close();
return 0;
}
```

CUPRINS

PREFAȚĂ.....	3
Partea I – ENUNȚURI.....	4
Varianta - Simulare 2022.....	4
Varianta 4 – Sesiunea specială 2022.....	6
Varianta 1 – Sesiunea iunie-iulie 2022.....	8
Varianta 5 – Sesiunea august 2022.....	10
Varianta - Simulare 2023.....	12
Varianta 6 – Sesiunea specială 2023.....	14
Varianta 5 – Sesiunea iunie-iulie 2023.....	16
Varianta 7 – Sesiunea august 2023.....	18
Varianta – Simulare 2024.....	20
Varianta 4 – Sesiunea specială 2024.....	22
Varianta 3 – Sesiunea iunie-iulie 2024.....	24
Varianta 8 – Sesiunea august 2024.....	26
Varianta – Simulare 2025.....	28
Varianta 6 – Sesiunea specială 2025.....	30
Varianta 1 – Sesiunea iunie-iulie 2025.....	32
Varianta 4 – Sesiunea august 2025.....	34
Partea a II-a - INDICAȚII DE REZOLVARE.....	36
Varianta – Simulare 2022.....	36
Varianta 4 – Sesiune specială 2022.....	39
Varianta 1 – Sesiunea iunie-iulie 2022.....	43
Varianta 5 – Sesiunea august 2022.....	46
Varianta – Simulare 2023.....	49
Varianta 6 – Sesiunea specială 2023.....	52
Varianta 5 – Sesiunea iunie-iulie 2023.....	55
Varianta 7 – Sesiunea august 2023.....	58
Varianta – Simulare 2024.....	61
Varianta 4 – Sesiunea Specială 2024.....	64
Varianta 3 – Sesiunea iunie-iulie 2024.....	68
Varianta 8 – Sesiunea august 2024.....	71
Varianta – Simulare 2025.....	74
Varianta 6 – Sesiunea specială 2025.....	78
Varianta 6 – Sesiunea iunie-iulie 2025.....	81
Varianta 4 – Sesiunea august 2025.....	85

CUPRINS.....	89
--------------	----