MODEL TEST 1

- *Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.*
- *Timpul efectiv de lucru este de 3 ore.*
- *În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).*

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 **a,b** şi **z** sunt întregi, iar **a≤b.** Care dintre expresiile **C/C++** următoare are valoarea **1** dacă şi numai dacă valoarea variabilei **z** este pară și nu aparţine intervalului închis determinat de valorile variabilelor **a** şi **b** ?

a. z%2==0 && z>a || z>b **b.!(z<a && z>b) && z%2==0**

c. z<a && z>b && z%2==0 **d.!(z>=a && z<=b) && z%2==0**

2. Se consideră secvența de instrucțiuni următoare:

```
int n,i=1,k=1;
cin>>n;
while(k*k<=n)
{
    i++;
    k=k+i;
}
cout<<k;
```

Ce valoare se afișează dacă pentru n se citește valoarea 99.

a. 0 **b.20** **c.5** **d.10**

3. Care dintre următoarele variante de instruncțiuni inserează cifra 2 înaintea ultimei cifre a unui număr natural n.

1. n=(n%10*10+2)*10+n/10; **2. n=(n/10*10+2)*10+n%10;**

3. n=n/10+2*10+n%10; **4. n=(n/10+2)*10+n%10;**

a. 1 **b. 2** **c. 2, 3** **d. 3, 4**

4. Care dintre expresiile următoarele are valoarea 1 dacă și numai dacă valorile variabilelor a și b sunt numere întregi pare consecutive.

a. **(a%2)&&(b%2)&&(a-b==2)** **b.(a%2)&&(a-b==2||b-a==2)**

c. (a%2==0)&&!(abs(a-b)==2) **d.(a%2==0)&&(abs(a-b)==2)**

5. Pentru tabloul unidimensional **(4,6,14,25,61,73,82,87,95,96,98)** numărul minim de elemente ale tabloului care trebuie verificate până este găsit elementul **82** este:

a. 7 **b.2** **c.3** **d.4**

Subiectul II (40 de puncte)

1. Se consideră algoritmul alăturat, descris în pseudocod.
S-a notat cu **x%y** restul împărţirii numărului natural **x** la numărul natural nenul **y** şi cu **[z]** partea întreagă a numărului real **z**.
a) Scrieţi valorile care se vor afişa dacă se citesc în ordine numerele 5 15 45 33 81 66 44 87. **(6p.)**
b) Daca pentru n, a și b se citesc valorile 5 50 100 completați setul de date cu valori care pot fi citite astfel încât, în urma executării algoritmului, valoarea afișată pentru **m** să fie **4**. **(6p.)**
c) Scrieţi programul **C/C++** corespunzător algoritmului dat. **(10p.)**
d) Scrieţi în pseudocod un algoritm echivalent cu cel dat în care structura **pentru …execută** să fie înlocuită cu o structură repetitivă cu test inițial. **(6p.)**

```
citeste  n, a, b (numere naturale nenule)
m←0
┌pentru i←1,n execută
| citește x
| ┌dacă x≥a și x≤b atunci
| | ┌dacă x%10=[x/10] atunci
| | |     m←m+1
| | └■
| └■
└■
scrie m
```

2. Tabloul unidimensional A, cu 5 elemente având valori distincte, memorează cele mai mici 5 numere naturale nenule pătrate perfecte. Tabloul unidimensional B, cu 4 elemente având valori distincte, memorează cele mai mici 4 numere naturale prime. Tablourile A și B sunt sortate descrescător. Se interclasează descrescător cele două tablouri A și B în tabloul unidimensional C. Precizați care sunt elementele tabloului C. **(6p.)**

3. Variabilele întregi i și j memorează numere naturale. Precizați ce se afișează după executarea instrucțiunilor de mai jos. **(6p.)**

```
for(i=0; i<=3; i++)
        for(j=3; j>=i; j--)
                if (j%3==2)
                  cout<<i+j;
```

Subiectul III (30 de puncte)

1. Un șir bicolor este reprezentat sub forma unui tablou unidimensional cu **n** elemente ce conţine numere naturale din mulțmea **{0,1}**. spunem că șirul este **perfect** dacă există o singură secvență cu indici în intervalul **i1, i2** cu toate elementele sale egale cu **0** iar șirul nu conține alte elemente egale cu **0**. Scrieți programul C/C++ care citeşte de la tastatură numerele naturale **n (2≤n≤200)** și apoi n valori din mulțimea **{0,1**} (cel puțin 2 valori distincte) reprezentând elementele șirului și afișează pe ecran mesajul **DA**, dacă șirul este perfect sau mesajul **NU** în caz contrar. **(10p)**

Exemplu: pentru **n=5** şi tabloul alăturat, **1 0 0 0 1,**se va afişa **DA**

Exemplu: pentru **n=5** şi tabloul alăturat, **1 0 1 0 1,**se va afişa **NU**

2. Un număr natural nenul se numește **echilibrat** dacă suma cifrelor de pe poziții pare este egală cu suma cifrelor de pe poziții impare. Cifrele se numerotează de la dreapta la stânga începând cu valoarea **0**. Exemplu: **121** este număr echilibrat pentru că **2=1+1**. Scrieți un program în pseudocod care citește două numere naturale **a** și **b (2≤a<b≤10^9, b-a≤10000)** și afișează pe ecran, în ordine descrescătoare, separate prin câte un spațiu, toate numerele **echilibrate** din intervalul **[a,b]**. Dacă în interval nu există astfel de numere, se afișează pe ecran mesajul **nu exista**. **(10p)**

Exemplu: pentru **a=100 b=150**, se afișează pe ecran: **143 132 121 110**.

3. O secvență de **K** elemente a unui șir de numere naturale este numită secvență **RK**, dacă elementele din secvență dau resturi distincte la împărțirea cu K. Fișierul **bac.txt** conține pe prima linie un număr natural **K** din intervalul **[1,10]**, iar pe a doua linie conține un șir de cel puțin **K** și cel mult $\mathbf{10^3}$ numere naturale din intervalul **[0,10^4]**, separate prin câte un spațiu. Se cere să se afișeze pe ecran suma elementelor primei secvențe **RK** din șirul de pe a doua linie a fișierului, dacă în șir există secvențe **RK** sau mesajul **NU EXISTA** dacă șirul de pe a doua linie a fișierului nu conține nicio secvența **RK**. Proiectați un algoritm eficient din punctul de vedere al timpului de executare.

Exemplul 1. Dacă fișierul conține numerele

3

10 ***10 11 3*** **4** **2 49 21** **27 12 13** atunci se afișează pe ecran 24

(secvența **RK 11 3 4**)

Exemplul 2. Dacă fișierul conține numerele

3

10 11 13 16 11 10 atunci se afișează pe ecran **NU EXISTĂ** (în șir nu există nicio secvență **RK**).

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

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

REZOLVARE SUBIECTE TEST 1

Subiectul I (20 de puncte)

1. d.!(z>=a && z<=b)&&z%2==0
2. d.10

3. b. 2

4. d.(a%2==0)&&(abs(a-b)==2)

5. c.3

Subiectul II (40 de puncte)

1.a) 2 **(6p.)**
b) n=5 a=50 b=100 putem citi valorile 55 66 77 88 90 sau orice set format din 5 valori din care 4 sunt din mulțimea {55, 66, 77, 88, 99} **(6p.)**
c)
```
#include <iostream>
using namespace std;
int main()
{
   int n, a, b, i, x, m;
   cin>>n>>a>>b;
   m=0;
   for(i=1;i<=n;i++)
   {
      cin>>x;
      if(x>=a&&x<=b)
       if(x%10==x/10)
          m++;
   }
   cout<<m;
   return 0;
}
```
d. structura repetitivă cu test initial este structura cat timp … executa **(6p.)**

```
citeste  n, a, b (numere naturale nenule)
m←0;i←1
┌cat timp i≤n execută
| citește x
|  ┌dacă x≥a și x≤b atunci
| |  ┌dacă x%10=[x/10] atunci
| | |      m←m+1
| |  └■
|  └■
| i←i+1
 └■
scrie m
```

2. A=(25, 16, 9, 4, 1) B=(7, 5, 3, 2) C=(25, 16, 9, 7, 5, 4, 3, 2, 1). (6p.)

3. Se afișează 234 **(6p.)**

Subiectul III (30 de puncte)

1. Vom memora tabloul și vom determina prima și ultima apariție a unui element egal cu 0. Verificăm dacă elementele cuprinse între cele două apariții sunt toate egale cu 0.

```
#include <iostream>
using namespace std;
int main()
{
   int n, v[201], i,i1=0,i2=0, ok=1,primul=0;
   cin>>n;
   for(i=1;i<=n;i++)
    {
       cin>>v[i];
       if(v[i]==0&&primul==0)
            {i1=i;primul=1;}
       if(v[i]==0)i2=i;
    }
   for(i=i1;i<=i2;i++)
     if(v[i]!=0)ok=0;
   if(primul==0||ok==0)cout<<"NU";
   else cout<<"DA";
   return 0;}
```

2. Căutăm numerele echilibrate numai printre numerele divizibile cu 11.

```
întreg a, b, ok,si,sp,ci,r;
citeste  a, b (numere naturale nenule)
ok←0
 ┌pentru i←b,a,-1 execută
 | ┌dacă i%11=0 atunci
 | |ci←i; si←0; sp←0; r←0;
 | | ┌cat timp ci>0 executa
 | | |   ┌dacă r%2=0 atunci
 | | |   |  sp←sp+ci%10;
 | | |   |altfel
 | | |   |  si←si+ci%10;
 | | |   └■
 | | |   r←r+1;   ci←[ci/10]
 | | └■
 | | ┌dacă si=sp atunci
 | | |    scrie i,' ';
 | | |      ok←1
 | | └■
 | └■
 └■

 ┌dacă ok=0 atunci
 |  scrie "nu exista"
 └■
```

3. a) O soluție eficientă utilizează un vector cu **k** elemente cu ajutorul căruia vom memora secvența curentă cu k elemente prin utilizarea vectorului circular, pentru a vedea daca resturile la împărțirea cu k sunt distincte folosim un vector de aparitii, dar și contorizăm câte resturi distincte avem în secvența curentă cu ajutorul variabile **nr**, iar cu variabila **s** determinăm suma secvenței curente. Algoritmul este eficient din punct de vedere a spațiului de memorie deoarece utilizează doar variabile simple și 2 vectori a câte 10 elemente (k≤10). Eficiența timp este data de faptul că este un algoritm liniar, la o singura trecere prin fișier determină valoarea cerută, complexitatea algoritmului depinde doar de numărul de valori din fișier.

4. b)

```
#include <iostream>
#include<fstream>
using namespace std;
ifstream fin("bac.txt");
int k, i, x, v[10], s, ap[10],nr,ok,y;
int main()
{   fin>>k;
   for(i=0;i<k;i++)
   {
      fin>>v[i];
      ap[v[i]%k]++;
      if(ap[v[i]%k]==1)nr++;
      s=s+v[i];
   }
   if(nr==k){ok=1;cout<<s; return 0;}
   while(fin>>x)
   {
      y=v[i%k];
      s=s-y;
      if(ap[y%k]==1)nr--;
      ap[y%k]--;
      s=s+x;v[i%k]=x;
      ap[x%k]++;
      if(ap[x%k]==1)nr++;
      if(nr==k){cout<<s;ok=1;return 0;}
      i++;
   }
   if(ok==0)cout<<"NU EXISTA";
   return 0;
}
```

BAREM DE EVALUARE ȘI DE NOTARE TEST 1

- *Se punctează oricare alte modalităţi de rezolvare corectă a cerinţelor.*
- *Nu se acordă punctaje intermediare, altele decât cele precizate explicit prin barem. Nu se acordă fracţiuni de punct. Se acordă 10 puncte din oficiu. Nota finală se calculează prin împărţirea punctajului total acordat pentru lucrare la 10.*
- *Utilizarea unui tip de date care depăşeşte domeniul de valori precizat în enunţ este acceptată dacă acest lucru nu afectează corectitudinea în funcţionarea programului.*

SUBIECTUL I (20 de puncte)

1. d 2. d 3. b 4. d 5. c	**5x4p.**

SUBIECTUL al II - lea (40 de puncte)

1.	**a)** Răspuns corect: 2	**6p.**	
	b) Pentru răspuns corect	**6p.**	orice set format din 5 valori din care 4 sunt din mulțimea {55, 66, 77, 88, 99}
	c) Pentru program corect -declarare variabile -citire date -afişare date -instrucţiune de decizie -instrucţiunea repetitivă -atribuiri -corectitudine globală a programului[1)]	**10p.** 1p. 1p. 1p. 2p. 3p. 1p. 1p.	
	d) Pentru algoritm pseudocod corect -echivalenţă a prelucrării realizate, conform cerinţei (*) -corectitudine globală a algoritmului[1)]	**6p.** 5p. 1p.	(*) Se acordă numai 2p. dacă algoritmul are o structură repetitivă conform cerinţei, principial corectă, dar nu este echivalent cu cel dat. Se va puncta orice formă corectă de structură repetitivă conform cerinței.
2.	**Pentru rezolvare corectă** **A=(25, 16, 9, 4, 1) B=(7, 5, 3, 2)** **C=(25, 16, 9, 7, 5, 4, 3, 2, 1)**. (*)	**6p.** (**)	(*) Se acordă câte 2p. pentru fiecare aspect (determinarea corecta al fiecărui vector) (**) Se acordă întreg punctajul dacă vectorul final este corect, dar nu au fost precizați vectorii intermediari
3.	**Pentru rezolvare corectă** 234 (*)	**6p.** .	(*) Se acordă numai 3p. dacă doar o parte dintre valorile afișate sunt conform cerinței.

SUBIECTUL al III - lea (30 de puncte)

1.	**Pentru program corect** -declararea variabilelor -citire a datelor -verificarea proprietăților cerute, (*) -corectitudine a globală a programului[1)]	**10p.** 1p. 2p. 6p. 1p	(*) Se acordă câte 2p. pentru fiecare aspect al cerinței (valorile egale cu 0 în același interval, valorile egale cu 1 în afara intervalului, cazul cand nu sunt îndeplinite condițiile din enunț).
2.	**Pentru algoritm corect** - citire date, inițializare (*) - determinare valorilor cerute (**) - verificarea existenței valorilor cerute -corectitudine globală a programului[1)]	**10p.** 2p. 6p. 1p. 1p.	(*) Se acordă câte 1p. pentru fiecare aspect conform cerinței. (**) Se acordă câte 3p. pentru fiecare aspect al cerinței (calculul celor două sume, condiția de afișare).
3.	a) Pentru răspuns corect -coerența descrierii algoritmului (*) -justificare a unor elemente de eficiență **b) Pentru program corect** -operaţii cu fişiere: declarare, pregătire în vederea citirii, citire din fişier -determinarea valorilor cerute (*),(**) -utilizarea unui algoritm eficient (***) -declarare a variabilelor, afișare a datelor -corectitudine globală a programului [1)]	**2p.** 1p. 1p. **8p.** 1p. 5p. 1p. 1p.	(*) Se acordă punctajul chiar dacă algoritmul ales nu este eficient. (**) Se acordă numai 3p. dacă algoritmul este principial corect, dar nu oferă rezultatul cerut pentru toate seturile de date de intrare. (***) Se acordă punctajul numai pentru un algoritm liniar. O soluție eficientă utilizează un vector cu k elemente cu ajutorul căruia vom memora secvența curentă cu k elemente prin utilizarea vectorului circular, pentru a vedea daca resturile la împărțirea cu k sunt distincte folosim un vector de apariții, dar și contorizăm câte resturi distincte avem în secvența curentă cu ajutorul variabile nr, iar cu variabila s determinăm suma secvenței curente. Algoritmul este eficient din punct de vedere a spațiului de memorie deoarece utilizează doar variabile simple și 2 vectori a câte 10 elemente (k≤10). Eficiența timp este data de faptul că este un algoritm liniar, la o singura trecere prin fișier determină valoarea cerută, complexitatea algoritmului depinde doar de numărul de valori din fișier.