MODEL TEST 5

- *Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.*
- *Timpul de lucru efectiv este de 3 ore.*
- *Identificatorii utilizați în rezolvări trebuie să respecte precizările din enunț (bold), iar în lipsa unor precizări explicite, notațiile trebuie să corespundă cu semnificațiile asociate acestora (eventual în formă prescurtată). Datele de intrare se consideră corecte, validarea lor nefiind necesară.*
- *În grafurile din cerințe oricare arc/muchie are extremități distincte și oricare două arce/muchii diferă prin cel puțin una dintre extremități.*

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 se notează cu 4 puncte.

1. Variabila **n** memorează un număr natural cu 5 cifre. Care dintre expresiile C/C++ de mai jos schimbă cifra din mijloc cu cifra 0?

a) n=n%1000+n/100*100
b) n=n%10+n/10*10+n/100*100
c) n=n%100+n/1000*1000
d) n=n/10%100*1000+n/100

2. Fiind dat un graf complet, se știe că pentru a obține 3 componente conexe, trebuie să eliminăm minim 9 muchii. Care este numărul minim de noduri pe care ar trebui să îl aibă graful inițial?

a) 8
b) 9
c) 7
d) 6

3. Utilizând metoda backtracking, se generează toate șirurile de câte patru operatori din mulțimea {'+', '-', '*', '/', '%'}, șiruri în care nu pot fi alăturate primul și ultimul operator. Primele 8 șiruri sunt: **+-+-**, **+-+***, **+-+/**, **+--+**, **+---**, **+--***, **+--/**, **+--%**. Câte dintre aceste șiruri generate încep cu '-' și se termină cu '%'?

a) 13
b) 15
c) 10
d) 20

4. Se consideră un arbore cu 9 noduri, numerotate de la 1 la 9, cu rădăcina nodul 1 în care avem muchiile: [1,2], [1, 3], [1,4], [4, 5], [4, 6], [5, 7], [5, 8], [7, 9]. Care sunt ascendenții nodului 7?

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

5. Se consideră subprogramul **f** definit alăturat. În urma cărui apel valoarea returnată de subprogram va fi **8**?

a) f (34)
b) f (10)
c) f (128)
d) f (256)

```
int f(int n){
    if(n==1) return 0;
    if(n%2==0)
        return 1+f(n/2);
    else
        return 1+f(3*n+1);
}
```

1. Algoritmul alăturat este reprezentat în pseudocod.

S-a notat cu a%b restul împărțirii numărului natural a la numărul natural b.

a) Scrieți valoarea care se afișează în urma executării algoritmului dacă se citesc, în această ordine, numerele **4**, **3**, **12**, **36**, **45**, **51**, **27**, **87**, **17**, **25**. **(6p.)**

b) Dacă pentru **n** se citește 2, iar pentru **p** se citește 5, scrieți patru numere distincte care pot fi citite, astfel încât în urma executării algoritmului, valoarea afișată să fie **0**. **(6p.)**

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

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

```
citește n, p
(numere naturale nenule)
nr←0
repetă
   citește x,y
   (numere naturale)
   cât timp y≠0 execută
      z←x%y
      x←y
      y←z
   dacă (x=p) atunci
      nr←nr+1
   n←n-1
până când n=0
scrie nr
```

2. Fie următoarea definiție și declarație:

```
struct examen{
    char nume[30], prenume[30], initiala;
    float nota[3], medie;
}e[300];
```

Să se scrie secvența de program care citește pentru **n** candidați la examenul de bacalaureat următoarele date: numele, inițiala tatălui, prenumele, notele la cele trei probe și calculează media obținută la examen. În cazul în care candidatul a promovat cele trei probe și examenul de bacalaureat să se afișeze media sau mesajul **respins** în caz contrar (un candidat este promovat dacă are cel puțin nota 5 la fiecare probă și media cel puțin 6).

(6p.)

3. În secvența de instrucțiuni de mai jos variabilele i și j sunt de tip întreg, iar variabila **A** memorează un tablou bidimensional cu 4 linii și 6 coloane, numerotate începând de la 1. Elementele tabloului sunt numere întregi. Fără a utiliza alte variabile, scrieți una sau mai multe instrucțiuni care pot înlocui punctele de suspensie astfel încât, în urma executării secvenței obținute, tabloul memorat în variabila **A** să aibă elementele din figura de mai jos. **(6p.)**

```
for(i=1;i<=4;i++)
  for (j=1;j<=6;j++)
     ..............
```

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

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

1. Un număr natural **n**, având un număr de **k** cifre, se numește număr **Armstrong** dacă este egal cu suma cifrelor sale ridicate la puterea k.
Subprogramul **armstrong** are un singur parametru, **n**, prin care primește un număr natural (n∈ [0, 10^9]). Subprogramul returnează **1** dacă **n** este număr Armstrong și **0** în caz contrar. Scrieți definiția completă a subprogramului. **(10p.)**
Exemplu: dacă **n=153** subprogramul returnează 1 ($153=1^3+5^3+3^3$).

2. Într-un text cu cel mult **100** 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 menționat și afișează pe ecran, pe linii separate, toate cuvintele care încep și se termină cu o consoană, iar în rest nu conțin decât vocale. Dacă nu există niciun astfel de cuvânt, se afișează pe ecran mesajul **nu exista**. Se consideră vocale literele din mulțimea {**a**, **e**, **i**, **o**, **u**}. **(10p.)**
Exemplu: pentru textul **fetita a desenat pe caiet un deal acoperit cu verdeata si un cal cu coama neagra** se afișează pe ecran, nu neapărat în această ordine, cuvintele de mai jos:
caiet
deal
cal

3. Numerele întregi pozitive cu proprietatea că, prin însumarea iterativă a pătratelor cifrelor lor, se ajunge în cele din urmă la numărul 1, se numesc **numere fericite**. Numărul 7 este un număr fericit pentru că $7^2=49$, $4^2 + 9^2=97$, $9^2 + 7^2=130$, $1^2 + 3^2 + 0^2=10$, $1^2 + 0^2=1$. Prelucrând astfel orice număr, în cele din urmă se va ajunge doar la unul dintre următoarele numere posibile: 0, 1, 4, 16, 20, 37, 42, 58, 89 sau 145.
Fișierul **bac.in** conține un șir de cel mult 10^6 numere naturale din intervalul [0,10^9], separate prin câte un spațiu. Se cere să se afișeze în fișierul **bac.out,** pe câte un rând, numerele din șir care sunt *fericite* urmate de numărul de iterații necesare pentru a ajunge la numărul 1. Dacă în șir nu există niciun număr fericit se va afișa mesajul **nu exista**. Proiectați un algoritm eficient din punct de vedere al timpului de executare și al memoriei utilizate.
Exemplu:

bac.in	**bac.out**
7 13 95 104 86 17 379 226 445 33	**7 5** **13 2** **86 2** **379 6** **226 5**

a) Scrieți programul C/C++ corespunzător algoritmului proiectat. **(8p.)**
b) Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. **(2p.)**

Subiectul I

1. c

2. d

3. b

4. c

5. d

Subiectul II

1. Rezolvare:

a) 2

n	p	nr	x	y	z
4	3	0	12 36 12	36 12 0	12 0
3		1	45 51 45 6 3	51 45 6 3 0	45 6 3 0
2		2	27 87 27 6 3	87 27 6 3 0	27 6 3 0
1			17 25 17 8 1	25 17 8 1 0	17 8 1 0
0					

b) Algoritmul numără câte seturi de numere x, y au cel mai mare divizor comun numărul p.

Un set de numere posibile: 64, 26, 13, 29

c) Programul C++:

```
#include <iostream>
using namespace std;
int main(){
 int n,p,nr,x,y,z;
 cin>>n>>p;
 nr=0;
 do{
     cin>>x>>y;
     while(y!=0){
         z=x%y;
         x=y;
         y=z;
     }
     if(x==p)
         nr++;
      n--;
    }while(n!=0);
    cout<<nr;
 return 0;
}
```

d) Pseudocod echivalent:

```
citește n, p (numere naturale)
nr←0
pentru i←1,n,1 execută
    citește x,y (numere naturale)
    cât timp y≠0 execută
        z←x%y
        x←y
        y←z
    dacă (x=p) atunci
        nr←nr+1
scrie nr
```

2.

```
int i,n;
for(i=1;i<=n;i++){
      cin.getline(e[i].nume,30);
      cin.getline(e[i].prenume,30);
      cin>>e[i].initiala>>e[i].nota[0]>>e[i].nota[1]>>e[i].nota[2];
      if(e[i].nota[0]>=5 && e[i].nota[1]>=5 && e[i].nota[2]>=5){
            e[i].medie=(e[i].nota[0]+e[i].nota[1]+e[i].nota[2])/3;
            if(e[i].medie>=6)
                  cout<<e[i].medie;
            else
                  cout<<"respins";
      }
      else cout<<"respins";
 }
```

3.

```
if(i%2==1)
      a[i][j]=i%j;
else  a[i][j]=j%i;
```

Subiectul III

1. Rezolvare:

```
int armstrong(int n){
    int k, aux,s,i,p;
    k=s=0;
    aux=n;
    while(aux!=0){
        k++;
        aux=aux/10;
    }
    aux=n;
    while(aux!=0){
        p=1;
        for(i=1;i<=k;i++)
            p=p*(aux%10);
        s=s+p;
        aux=aux/10;
    }
    if(n==s)
        return 1;
    else
        return 0;
}
```

2. Rezolvare:

```
#include <iostream>
#include<string.h>
using namespace std;
int main(){
    char s[100], *p;
    int i,ok,n,k=0;
    cin.getline(s,100);
    p=strtok(s," ");
    while(p){
        n=strlen(p);
        if(strchr("aeiou",p[0])==NULL && strchr("aeiou",p[n-1])==NULL){
            ok=1;
            for(i=1;i<strlen(p)-1;i++)
                if(strchr("aeiou",p[i])==NULL)
                    ok=0;
            if(ok==1){
                cout<<p<<endl;
                k++;
            }
        }
        p=strtok(NULL," ");
    }
    if(k==0)
        cout<<"nu exista";
    return 0;
}
```

3. Rezolvare:

```
#include <iostream>
#include<fstream>
using namespace std;
int main(){
    ofstream fout("bac.out");
    ifstream fin ("bac.in");
    int s,x,it,aux,ok;
    ok=0;
    while(fin>>x){
        aux=x; it=0;
        do{
            s=0; it++;
            while(x!=0){
                s=s+x%10*(x%10);
                x=x/10;
            }
            x=s;
        }while(s!=1 && s!=0 && s!=4 && s!=16 && s!=20 && s!=37 && s!=42
&& s!=58 && s!=89 && s!=145);
        if(s==1){
            fout<<aux<<' '<<it<<endl;
            ok=1;
        }
    }
    if(ok==0)
        fout<<"nu exista";
    return 0;
}
```

BAREM DE EVALUARE ȘI DE NOTARE TEST 5

- *Se punctează oricare alte modalități de rezolvare corectă a cerințelor.*
- *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)

1c 2b 3b 4c 5d	**5 x 4p.**

SUBIECTUL al II-lea (40 de puncte)

1.	**a) Răspuns corect:** 2	**6p.**	
	b) Pentru răspuns corect	**6p.**	Se acordă câte 3p. pentru fiecare set de numere x, y conform cerinței (un set de numere x, y este corect dacă c.m.m.d.c (x,y) este diferit de 5).
	c) Pentru program corect -declarare variabile -citire date -afișare date -instrucțiune de decizie -instrucțiuni repetitive (*) -atribuiri -corectitudine globală a programului[1)]	**10p.** 1p. 1p. 1p. 2p. 3p. 1p. 1p.	(*) Se acordă numai 2p. dacă doar una dintre instrucțiunile repetitive este conform cerinței.
	d) Pentru algoritm pseudocod corect -echivalență a prelucrării realizate, conform cerinței (*) -corectitudine globală a algoritmului[1)]	**6p.** 5p. 1p.	(*) Se acordă numai 2p. dacă algoritmul are o structură repetitivă conform cerinței, principial corectă, dar nu este echivalent cu cel dat. Se va puncta orice formă corectă de structură repetitivă conform cerinței.
2.	**Pentru rezolvare corectă** -acces la câmpurile înregistrării -citire date -verificare a condiției impuse (*) -corectitudine globală a expresiei[1)]	**6p.** 1p. 1p. 3p. 1p.	(*) Se acordă câte 1p. pentru fiecare aspect al cerinței referitor la condiția impusă (note, medie, operatori logici utilizați conform cerinței).
3.	**Pentru rezolvare corectă** -acces la un element al tabloului -atribuire a valorilor indicate elementelor tabloului (*) -corectitudine globală a secvenței[1)]	**6p.** 1p. 4p. 1p.	(*) Se acordă câte 2p. pentru fiecare aspect specific (atribuire valori, identificare linii cu indice par/linii cu indice impar). O soluție posibilă este atribuirea valorii expresiei i%j elementelor aflate pe linii cu indice impar, respectiv j%i elementelor aflate pe linii cu indice par.

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

1.	**Pentru subprogram corect** -antet subprogram (*) -verificare a proprietății cerute (**) -instrucțiune/instrucțiuni de returnare a rezultatului -declarare a tuturor variabilelor locale, corectitudine globală a subprogramului[1)]	**10p.** 2p. 6p. 1p. 1p.	(*) Se acordă câte 1p. pentru fiecare aspect al antetului (structură, declarare parametru de intrare) conform cerinței. (**) Se acordă câte 2p. pentru fiecare aspect al cerinței (identificare a unei cifre, ridicarea unui număr la o putere, calcul sumă).
2.	**Pentru program corect** -declarare a unei variabile care să memoreze un șir de caractere -citire a datelor -determinare a cuvintelor cerute (*) -afișare a datelor în formatul cerut și tratare a cazului **nu exista** -declarare a variabilelor simple, corectitudine globală a programului[1)]	**10p.** 1p. 1p. 6p. 1p. 1p.	(*) Se acordă câte 2p. pentru fiecare aspect al cerinței (obținere a unui cuvânt, localizare a consoanelor pe prima și ultima poziție a unui cuvânt, localizare a vocalelor).
3.	**a) Pentru program corect** -operații cu fișiere: declarare, pregătire în vederea citirii/scrierii, citire/scriere din/în fișier -determinare a valorilor cerute (*), (**) -utilizare a unui algoritm eficient (***) -declarare a variabilelor, citire a datelor, corectitudine globală a programului	**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 care utilizează eficient memoria. O soluție posibilă parcurge șirul din fișier, memorând valoarea curentă, calculează iterativ suma pătratelor cifrelor cât timp nu s-a ajuns la o valoare particulară și numără iterațiile efectuate.
	b) Pentru răspuns corect -coerență a descrierii algoritmului (*) -justificare a elementelor de eficiență	**2p.** 1p. 1p.	

[1)] Corectitudinea globală vizează structura, sintaxa, alte aspecte neprecizate în barem.