SUBIECTUL I **(30 de puncte)**

Pentru itemul 1, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect.

1. Variabilele **x** și **y** sunt de tip întreg și memorează câte un număr natural nenul. O expresie echivalentă cu cea alăturată poate fi: **(4p.)**

```
!(x%y!=0 || y<2)
```

a. `x%y==y%x && y>1`

b. `(x+y)%y==0 || y>=1`

c. `(x/y)*y==x && y>=2`

d. `x%y==0 || y>2`

2. Algoritmul alăturat este reprezentat în pseudocod.

S-a notat cu **a%b** restul împărţirii numărului natural **a** la numărul natural nenul **b** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte n
   (număr întreg nenul)
┌dacă n<0 atunci
│ n←-n
└■
 s←0
┌repetă
│ x←n%10
│┌pentru i←1,x execută
│└ s←s+x
└■
 n←[n/10]
└până când n=0
scrie s
```

a) Scrieţi numărul afişat dacă se citeşte valoarea **2018**. 69 **(6p.)**

b) Scrieţi patru numere distincte din intervalul **[10, 10^3]** care pot fi citite astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, să se afișeze valoarea **100**. **(4p.)**

6oricare patru numere din
68, 86, 608, 806, 680, 860

c) Scrieţi în pseudocod un algoritm echivalent cu cel dat, înlocuind adecvat structura **pentru...execută** cu o atribuire. **(6p.)**

d) Scrieţi programul C/C++ corespunzător algoritmului dat. **(10p.)**

c)

```
citeşte n
   (număr întreg nenul)
┌dacă n<0 atunci
│ n←-n
└■
 s←0
┌repetă
│ x←n%10
│   s=s+x*x
│ n←[n/10]
└până când n=0
scrie s
```

d)

```
#include <iostream>
using namespace std;

long int n,s; int i,x;

int main()
{
  cin>>n;
  if(n<0) n=-n;
  s=0;
  do{
     x=n%10;
     for(i=1;i<=x;i++)
        s=s+x;
   n=n/10;
  }while(n!=0);
cout<<s;
return 0;
}
```

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

Pentru fiecare dintre itemii 1 şi 2 scrieţi pe foaia de examen litera corespunzătoare răspunsului corect.

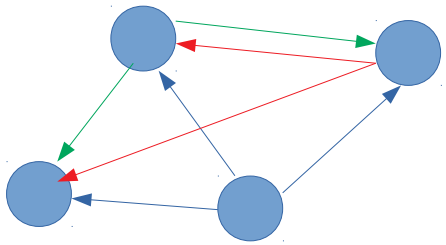
1. În declararea alăturată, variabila **m** memorează, pentru fiecare dintre cele **20** de medicamente dintr-o farmacie, prețul, precum și date despre substanța activă specifică: doza și codul acesteia. O expresie a cărei valoare reprezintă codul substanţei active specifice din primul medicament este: **(4p.)**

```
struct medicament
{ float pret;
  struct
  { int cod, doza;
  }substanta;
}m[20];
```

a. `m[0].cod.substanta`
b. `m[0].substanta.cod`
c. `m.cod.substanta[0]`
d. `m.substanta.cod[0]`

2. Într-un graf orientat cel puțin două vârfuri au gradul intern **2**, cel puțin un vârf are gradul intern **3** și cel puțin un vârf are gradul extern **3**. Numărul minim de vârfuri ale grafului este: **(4p.)**

a. 3
b. 4
c. 5
d. 6



Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

3. Un arbore are **9** noduri, numerotate de la **1** la **9**, şi muchiile **[1,2]**, **[1,6]**, **[1,8]**, **[1,9]**, **[2,3]**, **[2,7]**, **[4,5]**, **[5,7]**. Scrieți trei noduri care ar putea fi alese drept rădăcină astfel încât nodul **2** să aibă un număr minim de descendenți. **(6p.)**

oricare 3 dintre nodurile 1, 6, 8, 9

4. Variabilele **i** şi **j** sunt de tip întreg, iar variabila **a** memorează un tablou bidimensional cu **9** linii şi **9** coloane, numerotate de la **0** la **8**, având iniţial toate elementele egale cu **-1**.
Fără a utiliza alte variabile, scrieţi secvenţa de instrucţiuni de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenţei obţinute, variabila **a** să memoreze tabloul alăturat.

```
for(i=0;i<9;i++)
  for(j=0;j<9;j++)
        a[i][j]=(i+j)%8;
```

(6p.)

0	1	2	3	4	5	6	7	0
1	0	3	4	5	6	7	0	1
2	3	0	5	6	7	0	1	2
3	4	5	0	7	0	1	2	3
4	5	6	7	0	1	2	3	4
5	6	7	0	1	0	3	4	5
6	7	0	1	2	3	0	5	6
7	0	1	2	3	4	5	0	7
0	1	2	3	4	5	6	7	0

5. Fiind dat un cuvânt **s**, format numai din litere, și un cod **c**, de aceeași lungime cu **s**, format numai din cifre, numim **codificare** a lui **s** pe baza codului **c** operația de construire a unui nou șir, în care inițial se copiază prima literă din **s**, apoi, parcurgând de la stânga la dreapta restul șirului **s**, se adaugă litera curentă la începutul noului șir, dacă cifra corespunzătoare de pe aceeași poziție în **c** este pară, sau la finalul noului șir, în caz contrar.
Exemplu: dacă șirul **s** este **etalon**, iar codul este **025843** se obține cuvântul **oltean** (inițial șirul conține litera **e**, apoi se adaugă, în ordinea parcurgerii lui **s**, literele **t**, **l** și **o** la început, iar restul literelor la final).
Scrieţi un program C/C++ care citeşte de la tastatură două cuvinte, notate cu **s** și **c**, fiecare având cel mult 10^2 caractere, **s** fiind format doar din litere mici ale alfabetului englez, iar **c** fiind format doar din cifre. După primul cuvânt se tastează Enter. Programul construiește în memorie și afișează pe ecran cuvântul obținut prin codificarea lui **s** pe baza lui **c**, dacă cele două cuvinte au aceeași lungime, sau mesajul **cod incorect**, în caz contrar.

Exemplu: dacă se citesc cuvintele alăturate, se afişează pe ecran cuvântul **oltean** **(10p.)**

```
etalon
025843
```

Prima varianta de rezolvare (cu rezolvare in memorie)

consta in adaugarea de elemente in fata sirului dat sau dupa sirul dat si apoi a stergerii sirului initial din sirul obtinut (fara prima litera). Pentru a putea sa realizam stergerea trebuie sa stim cate cifre pare avem in cod pentru a stii peste cate litere trebuie sa facem saltul si sa incepem stergerea.

```
#include <iostream>
#include <string.h>
using namespace std;
char s[101], c[101],i,n,m,x[201];

void adaugare_fata(char x)
{
for(int i=strlen(s);i>0;i--)
        s[i]=s[i-1];
s[0]=x;
}

void adaugare_spate(char x)
{
        int n=strlen(s);
        s[n]=x;
        s[n+1]='\0';
}
int main()
{

        cin.get(s,100);
        cin.get();
        cin.get(c,100);
        if(strlen(s)!=strlen(c)) {
                                cout<<"cod incorect";return 0;}
        m=0;//numarul de cifre pare
```

```
        strcpy(x,s);
        for(i=1;i<strlen(c);i++)
                if((c[i]-'0')%2==0)
                        {m++;adaugare_fata(x[i]);}
                else
                        adaugare_spate(x[i]);

 cout<<endl;
 strcpy(s+m+1,s+m+strlen(c));
cout<<s;
return 0;
}
```

Alta varianta de rezolvare:

```
#include <iostream>
#include <string.h>
using namespace std;

int main()
{
        char s[101], c[101],i,n,m,x[201];
        cin.get(s,100);
        cin.get();
        cin.get(c,100);
        if(strlen(s)!=strlen(c)) {
                                cout<<"cod incorect";return 0;}

   x[strlen(s)-1]=s[0];
   int fata=strlen(c)-1;
   int spate=strlen(c)-1;
   for(i=1;i<strlen(s);i++)
      if((c[i]-'0')%2==0){
                fata--;
                x[fata]=s[i];
                /*cout<<fata<<' '<<x[fata]<<endl;*/
                }
                else
                {
                spate++;
                x[spate]=s[i];
                /*cout<<spate<<' '<<x[spate]<<endl;*/
                }
spate++;
x[spate]='\0';
strcpy(x,x+fata);
cout<<x;
return 0;
}
```

Am folosit un alt sir de caractere pentru care am rezervat un spatiu dublu data de s si c.
Prima litera din sir am pus-o in x, la mijloc, pe pozitia egala cu numarul de caractere din s si c. Apoi am inceput sa construiesc sirul: daca e litera para in c am pus la staga (in fata), daca e litera impara am pus la dreapta (adica la stanga).
Dupa ce am terminat de pus literele in noul sir x am marcat sfarsitul de sir pentru x.
Insa, in x pot exista elemente care nu s-au completat, in partea stanga. Stergem aceste elemente prin deplasarea elementelor in fata.

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

Pentru itemul 1, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect.

1. O companie organizează cursuri de programare în limbaje din mulțimea {**PHP**, **Java**, **Python**, **C#**, **SQL**}, astfel încât o persoană poate opta pentru un curs în care se studiază un număr **par** de limbaje, dar nu poate alege Java și Python în același curs. Utilizând metoda backtracking se generează toate posibilitățile unei persoane de a opta pentru un curs în cadrul ofertei companiei. Două cursuri sunt distincte dacă diferă prin cel puțin un limbaj sau prin ordinea în care se studiază limbajele. Primele cinci soluții generate sunt, în această ordine: **(PHP, Java)**, **(PHP, Java, C#, SQL)**, **(PHP, Java, SQL, C#)**, **(PHP, Python)**, **(PHP, Python, C#, SQL)**.
Soluția generată imediat după **(Java, PHP, SQL, C#)** este: **(4p.)**

a. **(Java, C#)** b. **(Java, PHP, C#, SQL)**
c. **(SQL, Python)** d. **(SQL, Java, C#, PHP)**

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

2. Subprogramul **f** este definit alăturat. Scrieți trei valori naturale pe care le poate avea variabila întreagă **x** astfel încât, în urma apelului de mai jos, pentru fiecare dintre acestea, să **NU** se afișeze niciun caracter *
f('e',x);
orice numar natural >=5 **(6p.)**

```
void f(char ch, int x)
{ cout<<ch; | printf("c",ch);
  if(x==0)
    cout<<'*'; | printf("*");
  else if(ch=='a')
         cout<<x; | printf("%d",x);
       else
         f(ch-1,x-1);
}
```

3. Subprogramul **resturi** are patru parametri, **n**, **x**, **y** și **r**, prin care primeşte câte un număr natural din intervalul **[1,10⁹]**, **r<x<y<n**. Subprogramul returnează numărul de valori naturale din intervalul **[1,n]** pentru care atât restul împărțirii la **x**, cât și restul împărțirii la **y**, sunt egale cu **r**.
Scrieţi definiţia completă a subprogramului.
Exemplu: pentru **n=200**, **x=5**, **y=14** și **r=2**, subprogramul returnează numărul **3** (pentru numerele **2**, **72** și **142** atât restul împărțirii la **5**, cât și restul împărțirii la **14**, este **2**). **(10p.)**

```
int resturi (long n, long x, long y, long r)
{
        int c=0;
        for(int i=1;i<=n;i++)
                if(i%x==r&& i%y==r) c++;
   return c;
}
```

4. Numim **secvență neuniformă** a unui șir de numere naturale un subșir al acestuia, format din termeni aflați pe poziții consecutive în șirul dat, cu proprietatea că oricare trei termeni aflați pe poziții consecutive sunt diferiți. Lungimea secvenței este egală cu numărul de termeni ai acesteia.
Fișierul **bac.txt** conține un șir de cel mult 10^6 numere naturale din intervalul [**0**,**9**]. Numerele sunt separate prin câte un spațiu, iar în șir există cel puțin trei termeni diferiți pe poziții consecutive.
Se cere să se afișeze pe ecran lungimea maximă a unei secvențe neuniforme a șirului aflat în fișier. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al memoriei utilizate.
Exemplu: dacă fişierul **bac.txt** conţine numerele
7 7 1 3 7 7 5 3 3 3 7 8 9
atunci pe ecran se afișează valoarea **4**
a) Descrieţi în limbaj natural algoritmul proiectat, justificând eficienţa acestuia. **(2p.)**
b) Scrieţi programul C/C++ corespunzător algoritmului descris. **(8p.)**

```
#include <iostream>
#include <fstream>
using namespace std;

ifstream f("bac.txt");
int main()
{
int max, maxim=0,x,y,z;
f>>x>>y;
if(x!=y) max=2;
   else max=0;
while(f>>z)
{
        if(x!=y && y!=z && z!=x)
        {
                max++;
        }
        else
   {
                if(max>maxim) maxim=max;
                if(z!=y) max=2;
                  else max=0;
        }
   x=y;
   y=z;
}
cout<<maxim;
return 0;}
```

Am folosit triplete de numere (x,y,z) carora le-am schimbat valoarile in mod corespunzator pentru fiecare valoare noua citita si construirea urmatorului triplet.
Am folosit doua variabile: max- pentru a retine numarul maxim de elemente curent cu proprietatea ceruta si maxim in care retin lungimea celei mai mari secvente de numere cu proprietatea ceruta.
Initial, citim doua elemente. Daca valorile lor sunt diferite atunci numarul maxim de elemente dinferite din triplet este 2 altfel este 0.
Daca numerele dintr-un triplet sunt diferite doua cate doua atunci numarul de elemente din secventa ceruta creste cu 1 altfel verificam daca numarul de elemente din secventa curenta este mai mare decat numarul maxim de elemente gasit pana acum si daca da, vom retine o noua valoare pentru valoarea maxima ceruta.
Programul foloseste putine variabile deci foloseste putina memorie.
Programul nu contine instructiuni repetitive imbricate, deci, timpul de executie este mic.