

**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 valoarea 1.
  - $21/3*19==3/19*21$
  - $21/19*3==19/3*21$
  - $19/3*21==3/21*19$
  - ☒  $19/21*3==3/19*21$
- Utilizând metoda backtracking, se generează toate drapelele formate din câte 3 culori distincte din mulțimea {alb, galben, negru, roșu, verde}. Două drapele sunt distincte dacă diferă prin cel puțin o culoare sau prin ordinea culorilor. Primele patru soluții obținute sunt, în această ordine: (alb, galben, negru), (alb, galben, roșu), (alb, galben, verde) și (alb, negru, galben). Indicați soluția generată imediat înainte de (galben, verde, alb).
  - (negru, roșu, verde)
  - (negru, alb, galben)
  - (galben, verde, roșu)
  - ☒ (galben, roșu, verde)
- Subprogramul `f` alăturat este incomplet definit. Indicați expresia cu care pot fi înlocuite punctele de suspensie, astfel încât valoarea lui `f(2019,1)` să fie egală cu numărul divizorilor pozitivi ai lui 2019.
 

```
int f(int n, int d)
{ if(.....) return 0;
  if(d*d==n) return 1;
  if(n%d==0) return 2+f(n,d+1);
  return f(n,d+1);
}
```

  - ☒  $d*d>n$
  - $d>n/2$
  - $d>n$
  - $d>2*n$
- Un graf neorientat cu 8 noduri, numerotate de la 1 la 8, are muchiile [1,2], [1,6], [4,6], [3,6], [6,5], [5,3], [3,4], [7,8], [8,2]. Trei noduri care nu aparțin niciunui ciclu în acest graf pot fi:
  - 1, 3, 4
  - ☒ 2, 7, 8
  - 3, 5, 6
  - 5, 6, 8
- Un arbore cu 19 noduri, numerotate de la 1 la 19, are ca rădăcină nodul numerotat cu 1; nodul 1 are un singur fiu, și anume nodul 2, iar fiecare nod  $i$  ( $i \geq 2$ ) fie este frunză, fie are drept fiu (descendenți direcți) noduri numerotate cu valori din intervalul  $[i+1, 2 \cdot i - 1]$ . Numărul maxim de frunze ale arborelui este:
  - 9
  - 12
  - ☒ 13
  - 16

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

S-a notat cu  $a \leq b$  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, în această ordine, numerele 5, 19 și 4. **12** (6p.)
- Dacă pentru variabila  $b$  se citește numărul 2019, iar pentru variabila  $k$  se citește numărul 5, scrieți cea mai mică și cea mai mare valoare care pot fi citite pentru variabila  $a$  astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, valoarea afișată să fie 0. **2016, 2019** (6p.)
- Scrieți programul C/C++ corespunzător algoritmului dat. (10p.)

citește  $a, b, k$ (numere naturale,  $1 \leq a \leq b, k > 1$ ) $pm \leftarrow 0; y \leftarrow 0; i \leftarrow b$ cât timp  $i \geq a$  execută $x \leftarrow i; p \leftarrow 0$ cât timp  $x \% k = 0$  execută $x \leftarrow [x/k]; p \leftarrow p+1$ 

■

dacă  $p \neq 0$  și ( $p < pm$  sau  $pm = 0$ ) atunci $pm \leftarrow p; y \leftarrow i$ 

■

 $i \leftarrow i-1$ 

■

scrie  $y$ 

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

d)

citește  $a, b, k$ (numere naturale,  $1 \leq a \leq b, k > 1$ ) $pm \leftarrow 0; y \leftarrow 0; i \leftarrow b$ cât timp  $i \geq a$  execută

```

     $x \leftarrow i; p \leftarrow 0$ 
    cât timp  $x \% k = 0$  execută
         $x \leftarrow [x/k]; p \leftarrow p+1$ 
    ■
    dacă  $p \neq 0$  și ( $p < pm$  sau  $pm = 0$ ) atunci
         $pm \leftarrow p; y \leftarrow i$ 
    ■
     $i \leftarrow i-1$ 
    ■

```

scrie  $y$ pentru  $i=b, a$  pas -1 executa $x \leftarrow i; p \leftarrow 0$ cât timp  $x \% k = 0$  execută $x \leftarrow [x/k]; p \leftarrow p+1$ 

■

dacă  $p \neq 0$  și ( $p < pm$  sau  $pm = 0$ ) atunci $pm \leftarrow p; y \leftarrow i$ 

■

■

```

c)
#include <iostream>
using namespace std;
int a, b, k;

int main()
{
    cin>>a>>b>>k;
    int pm=0, y=0,i,p,x;
    i=b;
    while(i>=a)
    {
        x=i;p=0;
        while(x%k==0)
        {
            x=x/k;
            p=p+1;
        }
        if(p!=0 && (p<pm || pm==0))
        {
            pm=p;
            y=i;
        }
        i=i-1;
    }
    cout<<y;
    return 0;
}

```

2. Expresiile de mai jos au ca valori numere naturale, reprezentând următoarele informații memorate pentru un eveniment din anul 2019: data desfășurării sale (ziua și luna) și un identificator. Scrieți definiția unei structuri cu eticheta **eveniment**, care să permită memorarea informațiilor menționate pentru un eveniment, și declarați corespunzător variabila **e**, de acest tip.

**e.data.zi      e.data.luna      e.id**

(6p.)

|                                                                             |  |                                                                                   |
|-----------------------------------------------------------------------------|--|-----------------------------------------------------------------------------------|
| <pre> struct data_ev {     unsigned int luna;     unsigned int zi; } </pre> |  | <pre> struct eveniment {     unsigned int id;     struct data_ev data; }e; </pre> |
|-----------------------------------------------------------------------------|--|-----------------------------------------------------------------------------------|

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 0 la 4, 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=0; i<5; i++)
    for (j=0; j<5; j++)
        if(i==0) a[i][j]=j;
        else a[i][j]=a[i-1][j]+2;

```

(6p.)

|   |   |    |    |    |
|---|---|----|----|----|
| 0 | 1 | 2  | 3  | 4  |
| 2 | 3 | 4  | 5  | 6  |
| 4 | 5 | 6  | 7  | 8  |
| 6 | 7 | 8  | 9  | 10 |
| 8 | 9 | 10 | 11 | 12 |

### SUBIECTUL al III-lea

(30 de puncte)

1. Subprogramul **inserare** are un singur parametru, **n**, prin care primește un număr natural ( $n \in [10, 10^5]$ ). Subprogramul furnizează prin același parametru numărul obținut din **n** prin inserarea, între oricare două cifre alăturate ale sale, a valorii absolute a diferenței acestora. Scrieți definiția completă a subprogramului.

**Exemplu:** dacă **n=7255**, atunci, după apel, **n=7523505**.

(10p.)

```

void inserare(unsigned long long &n)
{
    unsigned long long p=10;int r1=n%10,r2,x;
    unsigned long long m=r1;
    n=n/10;
    while(n>0)
    {
        r2=n%10;
        if(r2<r1) x=r1-r2;
        else x=r2-r1;
        m=m+p*x;
        p=p*10;
        m=m+p*r2;
        p=p*10;
        n=n/10;
        r1=r2;
    }
    n=m;
}

```

2. Într-un text de cel mult 50 de caractere cuvintele sunt separate prin câte un spațiu și sunt formate din litere mari ale alfabetului englez, urmate eventual de caracterul . (punct), dacă sunt scrise prescurtat. Textul reprezintă numele unei instituții de învățământ și doar cuvintele din mulțimea {COLEGIUL, LICEUL, NATIONAL, TEORETIC} pot fi prescurtate, eliminându-se ultimele 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, numele instituției scris fără prescurtări.

**Exemplu:** dacă se citește textul COLEG. NATIONAL DE INFORMATICA sau textul  
COLEG. NAT. DE INFORMATICA se obține COLEGIUL NATIONAL DE INFORMATICA (10p.)

```

#include <iostream>
#include <string.h>
#include <stdlib.h>

```

```
using namespace std;
```

```

int main()
{
    char str[101];
    char *p;
    int i,n; char v[50][20],k;
    cin.get(str,100);
    n=0;
    p = strtok(str, " ");
    while( p != NULL )
    {
        n++; strcpy(v[n],p);
        p = strtok(NULL, " ");
    }
    for(i=1;i<=n;i++)
    {
        k=strlen(v[i]);
        if(v[i][k-1]!='.')
        {
            v[i][k-1]='\0';
            if(strstr("COLEGIUL",v[i])!=0)
                strcpy(v[i],"COLEGIUL");
            if(strstr("TEORETIC",v[i])!=0)
                strcpy(v[i],"TEORETIC");
            if(strstr("NATIONAL",v[i])!=0)
                strcpy(v[i],"NATIONAL");
            if(strstr("TEORETIC",v[i])!=0)
                strcpy(v[i],"TEORETIC");
        }
    }
    for(i=1;i<=n;i++) cout<<v[i]<<' ';
    cout<<endl;
    return 0; }

```

3. Un șir format din  $2 \cdot n$  numere naturale se numește **paritar** dacă fiecare dintre primii săi  $n$  termeni fie are aceeași paritate cu oricare dintre ultimii săi  $n$  termeni, fie este strict mai mic decât oricare număr de paritate diferită aflat printre aceștia.

Fișierul **bac.txt** conține numere naturale din intervalul  $[0, 10^6]$ : pe prima linie un număr nenul,  $n$ , iar pe a doua linie un șir de  $2 \cdot n$  numere, separate prin câte un spațiu. Se cere să se afișeze pe ecran mesajul **DA**, în cazul în care șirul aflat în fișier este paritar, sau mesajul **NU**, în caz contrar. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al memoriei utilizate.

**Exemplu:** dacă fișierul are unul dintre conținuturile de mai jos, se afișează pe ecran mesajul **DA**.

|                             |  |                             |
|-----------------------------|--|-----------------------------|
| 5                           |  | 5                           |
| 20 3 11 4 15 25 49 18 53 16 |  | 20 3 11 4 15 25 49 81 53 61 |

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

```
#include <fstream>
#include <math.h>
using namespace std;

ifstream f("bac.txt");
int main()
{
    long n,x,i,min_st_par=pow(10,6),min_st_imp=pow(10,6),max_dr_par=-1,max_dr_imp=-1;
    f>>n;
    //prima jumătate
    for(i=1;i<=n;i++)
    {
        f>>x;
        if(x%2==0)
            if(x<min_st_par) min_st_par=x;
        if(x%2==1)
            if(x<min_st_imp) min_st_imp=x;
    }
    //a doua jumătate
    for(i=1;i<=n;i++)
    {
        f>>x;
        if(x%2==0)
            if(x>max_dr_par) max_dr_par=x;
        if(x%2==1)
            if(x>max_dr_imp) max_dr_imp=x;
    }

    if(min_st_par<max_dr_imp && min_st_imp<max_dr_par)        cout<<"DA";
    else
        if((min_st_par!=pow(10,6))&&(max_dr_par!=-1) && (min_st_imp<max_dr_par))cout<<"DA";
    else
        if((min_st_par<max_dr_imp)&&(min_st_imp!=pow(10,6))&&(max_dr_imp!=-1)) cout<<"DA";
        else cout<<"NU";

    return 0;
}
```

Din primele  $n$  elemente ale sirului (partea stanga) vom retine valoarea minima pentru elementele pare si impare. Din cea de-a doua parte a sirului (partea dreapta) vom retine valoarea maxima pentru elementele pare si impare.

Vom afisa DA doar daca:

- valoarea minima para este mai mica ca valoarea maxima impara si valoarea minima impara este mai mica ca valoarea maxima para (acest lucru inseamna ca exista elemente pare si impare in prima)
- daca exista elemente pare atat in partea dreapta cat si in partea stanga
- daca exista elemente impare atat in partea dreapta at si in partea stanga

Algoritmul foloseste putine variabile, nu foloseste vectori cu numar variabil de elemente deci vom folosi putina memorie pentru retinerea datelor.

Algoritmul nu contine structuri repetitive imbricate care se se execute de un numar variabil de ori in functie de datele de intrare ceea ce inseamna ca timpul de executie este scurt.