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** şi **y** sunt întregi și memorează câte un număr natural. Indicați o expresie C/C++ care are valoarea **1** dacă şi numai dacă numărul memorat în **x** aparține intervalului **[0,2019]**, iar numărul memorat în **y** aparține intervalului **[2019,2020]**.

   a. `x<=2019 || y==2019 && y==2020` b. `x<=2019 && y==2019 || y==2020`

   c. `!(x>2019 || y!=2019) && !(y>2020)` **d.** `!(x>2019) && !(y<2019 || y>2020)`

2. Utilizând metoda backtracking se generează toate posibilitățile de a planta în trei ghivece, aşezate de-a lungul unui pervaz, plante distincte din mulțimea {**azalea**, **begonia**, **vriesea**, **busuioc**, **ferigă**}, astfel încât în oricare două ghivece alăturate să nu fie două plante cu flori sau două plante fără flori; primele trei plante din mulțime sunt cu flori, iar celelalte sunt fără flori. Două soluții diferă prin cel puțin o plantă sau prin ordinea plantelor. Primele şase soluţii generate sunt, în această ordine, (**azalea**, **busuioc**, **begonia**), (**azalea**, **busuioc**, **vriesea**), (**azalea**, **ferigă**, **begonia**), (**azalea**, **ferigă**, **vriesea**), (**begonia**, **busuioc**, **azalea**), (**begonia**, **busuioc**, **vriesea**). Indicați a noua soluţie generată.

   a. (**begonia**, **ferigă**, **vriesea**) b. (**ferigă**, **azalea**, **begonia**)

   c. (**busuioc**, **azalea**, **ferigă**) **d.** (**vriesea**, **busuioc**, **azalea**)

3. Subprogramul **f** este definit alăturat. Indicați valoarea lui **f(2020,2)**.

```
int f(int x, int y)
{ if (y<1) return 0;
  else if (x%y==0) return 1+f(x/y,y);
       else return 2020;
}
```

   a. 2022 b. 2020 c. 2002 d. 2000

4. Indicați numărul de noduri ale unui arbore cu **16** muchii.

   a. 8 **b.** 17 c. 64 d. 136

5. Un graf neorientat are **20** de noduri și **9** muchii. Indicați numărul maxim de componente conexe din care poate fi format graful.

   a. 14 b. 15 **c.** 16 d. 17

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**.

```
citeşte n (număr natural)
m←0
dacă n=0 atunci
  m←10
altfel
  repetă
    c←n%10; n←[n/10]
    dacă c≥m atunci
      m←c
    altfel
      m←10
    ■
  până când n=0
■
scrie m
```

   a. Scrieți numărul afișat în urma executării algoritmului dacă se citește valoarea **45530**. **10** **(6p.)**

   b. Scrieți trei numere din intervalul **[10³,10⁴)** care pot fi citite astfel încât, pentru fiecare dintre acestea, în urma executării algoritmului, să se afișeze **1**. **3 numere din {1000, 1100, 1110, 1111}** **(6p.)**

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

   d. Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind adecvat structura **repetă...până când** cu o structură repetitivă cu test inițial. **(6p.)**

c)

```
#include <iostream>
using namespace std;
unsigned long n; unsigned int c,m;
int main()
{
cin>>n;
m=0;
if(n==0) m=10;
        else
        {
        do {
        c=n%10; n=n/10;
        if(c>=m) m=c;
           else m=10;
        }while(n!=0);
}
cout<<m;
return 0;
}
```

d)

```
citeşte n (număr natural)
m←0
dacă n=0 atunci
 m←10
altfel
 cat timp (n != 0)
 c←n%10; n←[n/10]
 dacă c≥m atunci
   m←c
 altfel
   m←10
scrie m
```

2. Variabila **s** memorează date specifice despre o seră: numărul de specii de flori (număr natural din intervalul **[3,10²]**) și date despre fiecare specie (denumire științifică și denumire populară, șiruri de cel mult **20** de caractere). Expresiile C/C++ de mai jos au ca valori numărul de specii de flori, denumirea științifică, respectiv denumirea populară a celei de a patra specii. Scrieți definiția unei structuri cu eticheta **sera**, care permite memorarea datelor despre o seră, și declarați corespunzător variabila **s**.
s.numar s.specie[3].denumireStiintifica s.specie[3].denumirePopulara **(6p.)**

```
struct den{
char denumireStiintifica[21];
char denumirePopulara[21];
}
```

```
struct sera{
struct den specie[101];
int numar;
}s;
```

3. Variabilele **i** și **p** sunt de tip întreg, iar variabila **a** memorează un tablou bidimensional cu **7** linii și **7** coloane, cu elemente numere întregi din intervalul **[0,10²]**.

Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni astfel încât, în urma executării acesteia, să se afişeze, separate prin câte un spațiu, produsul celor **7** elemente situate pe diagonala principală a tabloului, urmat de produsul elementelor situate simultan pe diagonala principală și pe primele **6** linii ale tabloului, şi aşa mai departe, astfel încât ultimul număr afișat să fie elementul situat simultan pe diagonala principală și pe prima linie a tabloului.
Exemplu: pentru tabloul alăturat se afișează **1680 840 420 84 21 7 1**

1	8	3	9	6	5	5
0	7	4	2	5	5	4
8	6	3	5	1	2	3
2	4	9	4	8	3	4
2	1	7	5	5	5	6
7	4	2	4	9	2	7
0	9	6	3	1	7	2

(6p.)

```
long p=1; int i;
for(i=1;i<=7;i++)
        p=p*a[i][i];
for(i=7;i>=1;i--)
{
        cout<<p<<' ';
        p=p/a[i][i];
}
```

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

1. Subprogramul **patrate** are doi parametri, **x** și **y**, prin care primește câte un număr natural (**1≤x≤y≤10⁹**). Subprogramul afişează pe ecran o expresie aritmetică reprezentând suma numerelor din intervalul **[x,y]** care au proprietatea că sunt pătrate perfecte, urmate de valoarea acestei sume. Termenii sumei sunt într-o ordine oarecare și sunt separați prin câte un simbol plus (**+**), iar valoarea sumei este precedată de simbolul egal (**=**), ca în exemplu. Dacă nu există niciun astfel de termen, se afișează pe ecran mesajul **nu exista**.
Scrieți definiția completă a subprogramului.
Exemplu: dacă **x=10** și **y=50** se poate afișa pe ecran **16+25+36+49=126** **(10p.)**

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

void patrate(unsigned long x, unsigned long y)
{
	unsigned long long s=0;
	unsigned long i;
	//gasim primul numar patrat perfect
	i=x;
	while(i<=y && (int)sqrt(i) != sqrt(i))
		i++;
	//urmatorii termeni ai sirului
	if(i>y) cout<<"nu exista";
	else
	{
	  cout<<i;s=i;i++;
	  while(i<=y)
	   {
		if((int)sqrt(i) == sqrt(i))
			{cout<<'+'<<i;s=s+i;}
		i++;
	   }
	   cout<<'='<<s;
	}

}
int main()
{
	unsigned long x,y;
	cin>>x>>y;
	patrate(x,y);
	return 0;
}
```

2. Într-un text cu cel mult 10^2 caractere, cuvintele sunt formate din litere mici și mari 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, astfel încât fiecare cuvânt să aibă prima literă mare, și toate celelalte litere mici. Textul obținut se afișează pe ecran.
Exemplu: dacă de la tastatură se introduce textul **`ABIA aSTept sa Merg lA scoala`**
se obține textul **`Abia Astept Sa Merg La Scoala`** **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;
int main()
{
	int i;
	char s[101];
	cin.get(s,100);
		//transformam prima litera din sir
	if((s[0]>='a'&& s[0]<='z')) s[0]=s[0]-'a'+'A';
		//transformam prima litera dupa fiecare spatiu
	i=1;
	while(i<strlen(s))
	{
		if(s[i-1]==' ' && s[i]>='a'&& s[i]<='z')
			s[i]=s[i]-'a'+'A';
		if(s[i-1]!=' ' && s[i]>='A'&& s[i]<='Z')
			s[i]=s[i]-'A'+'a';
		i++;
	}
	cout<<s;
	return 0;}
```

3. Fişierul **bac.txt** conţine un şir **crescător** de cel mult 10^6 numere naturale din intervalul $[0,10^9]$, separate prin câte un spaţiu. Se cere să se afişeze pe ecran fiecare număr distinct din şir, urmat de numărul de apariţii ale acestuia în şir. Numerele afișate sunt separate prin câte un spațiu. Proiectați un algoritm eficient din punctul de vedere al memoriei utilizate și al timpului de executare.

Exemplu: dacă fişierul **bac.txt** conține numerele **0 0 0 5 5 5 5 7 7 11 20 20**
se afișează **0 3 5 4 7 2 11 1 20 2**

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

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

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

Pentru citirea numerelor din fisier vom folosi doua varibile x si y care vor retine, in ordine, valoarea precedenta si valoarea curente citite din fisier.
Daca valorile variabilelor x si y sunt egale inseamna ca suntem intr-o secventa de numere egale si vom mari numarul de elemente egale din acea secventa (k++). Daca cele doua valori nu sunt egale inseamna ca suntem la inceputul unei secvente si vom initializa valoarea numarului de elemente egale k cu 1 pentru ca, deja, avem un prim element in secventa.
Avantajul major il constitue faptul ca elementele sunt in ordine cresatoare deci nu putem avea surprize cu verificarile explicate mai sus.
Datorita faptului ca programul contine un numar mic de variabile, spatiul alocat pentru variabile va fii mic.
Programul nu contine structuri repetitive imbricate, deci, timpul de executie pentru program este mic, complexitatea algoritmului este O(n) – unde n reprezinta numarul de elemente din fisier.