SUBIECTUL I **(30 de puncte)**

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

1. Variabila **x** este de tip întreg. Indicați o expresie care are valoarea **1** dacă și numai dacă expresia C/C++ alăturată are valoarea **1**. **(4p.)**

```
x<=3 || x>10
```

a. `!(x<3 && x<10)`

b. `x>=3 && !(x>=10)`

c. `!(x<3 || x<=10)`

d. `!(x>3) || !(x<=10)`

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, a
   (numere naturale nenule)
nr←0
i←1
┌cât timp i≤n execută
| citeşte b
|    (număr natural nenul)
| c←0
| ┌cât timp b%2=0 execută
| | b←[b/2]
| | c←c+1
| └■
| ┌dacă c=a atunci
| | nr←nr+1
| └■
| i←i+1
└■
scrie nr
```

a) Scrieţi valoarea afişată dacă se citesc, în această ordine, numerele **5**, **3**, **9**, **206**, **200**, **80**, **24**. **(6p.)**

b) Dacă pentru variabila **n** se citeşte numărul **4**, iar pentru variabila **a** se citeşte numărul **2**, scrieţi un set de numere distincte din intervalul **[10,99]** care pot fi citite în continuare astfel încât, în urma executării algoritmului, să se afișeze valoarea **4**. **(4p.)**

Numere din intervalul cerut, valori de forma 4·k, unde k este număr impar

c) Scrieţi în pseudocod un algoritm echivalent cu cel dat, înlocuind adecvat prima structură **cât timp...execută** cu o structură de tip **pentru...execută**. **(6p.)**

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

c)

```
citeşte n, a
   (numere naturale nenule)
nr←0
[redacted]pentru i=1,n executa
| citeşte b
|    (număr natural nenul)
| c←0
| ┌cât timp b%2=0 execută
| | b←[b/2]
| | c←c+1
| └■
| ┌dacă c=a atunci
| | nr←nr+1
| └■
| [redacted]
└■
scrie nr
```

d)

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

unsigned int n, a, i, nr, b, c;

int main()
{
    cin>>n>>a;
    nr=0;
    i=1;
    while(i<=n)
    {
        cin>>b;
        c=0;
        while(b%2==0)
        {
            b=b/2;
            c=c+1;
        }
        if(c==a)
            nr=nr+1;
        i=i+1;
    }
    cout<<nr;
    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. Un arbore cu **9** noduri, numerotate de la **1** la **9**, este reprezentat prin vectorul de „taţi" (**3**, **3**, **0**, **5**, **2**, **5**, **2**, **5**, **8**). Descendenții direcți ("fii") ai nodului cu eticheta **5** sunt: **(4p.)**

a. `2 7` **b.** `2 8` **c.** `3 3 0` **d.** `4 6 8`

2. Numărul de noduri ale unui graf neorientat fără cicluri, cu **26** de muchii și **12** componente conexe este: **(4p.)**

a. `18` **b.** `28` **c.** `38` **d.** `48`

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

3. Variabila **d**, declarată alăturat, memorează în câmpul **mic** cel mai mic divizor, strict mai mare decât **1**, al numărului natural din intervalul **[2,10²]**, memorat în câmpul **nr**.

```
struct divizor
{ int nr, mic;
} d;
```

Scrieţi o secvență de instrucțiuni în urma executării căreia, pentru numărul memorat în câmpul **nr** al variabilei **d**, se afișează pe ecran mesajul **prim**, dacă numărul este prim, mesajul **patrat** dacă numărul este pătratul unui număr prim, sau două numere naturale, separate printr-un spațiu, reprezentând cel mai mic și cel mai mare dintre divizorii proprii pozitivi ai săi. Divizorii proprii pozitivi ai unui număr sunt divizori pozitivi diferiţi de **1** şi de el însuşi.

Exemplu: dacă în câmpul **nr** se memorează numărul **12**, iar în câmpul **mic** se memorează numărul **2**, se afișează pe ecran

```
2 6
```

iar dacă în câmpul **nr** se memorează numărul **9**, iar în câmpul **mic** se memorează numărul **3**, se afișează pe ecran mesajul

```
patrat
```

(6p.)

```
#include <iostream>
#include <math.h>
using namespace std;
struct divizor
{
   int nr, mic;
}d;

int prim(int n)
{
   if(n<2) return 0;
   for(int i=2;i<=n/2;i++)
      if(n%i==0) return 0;
   return 1;}
int patrat(int n)
{
   if((int)sqrt(n) == sqrt(n))
      return 1;
   else
      return 0;}
int main()
{
   cin>>d.nr;
   cin>>d.mic;
   //(floor)(sqrt(d.nr)) poate fi inlocuita cu prim(d.mic)
   if(patrat(d.nr)&&prim((floor)(sqrt(d.nr)))) cout<<"patrat";
   else
       if(prim(d.nr)) cout<<"prim";
       else
       cout<<d.mic<<" "<<d.nr/d.mic;

   return 0;}
```

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 **1** la **9**, având iniţial toate elementele nule.
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=1;i<=9;i++)
  for(j=1;j<=9;j++)
        if(j>=i){
                    a[i][j]=i;
                    a[j][i]=i;
                  }
```

(6p.)

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

5.

5. Un text are cel mult **100** de caractere și este format din cuvinte, numere naturale şi spaţii. Cuvintele sunt formate numai din litere mici ale alfabetului englez. Cuvintele şi numerele sunt separate prin câte un spaţiu, ca în exemplu.
Scrieţi un program C/C++ care citeşte de la tastatură un text de tipul menţionat mai sus şi afişează pe ecran numărul din text care începe cu cea mai mare cifră, ca în exemplu. Dacă există mai multe astfel de numere, se afișează doar unul dintre acestea, iar dacă textul nu conține niciun număr, se afișează pe ecran mesajul **nu exista**.
Exemplu: pentru textul
am 190 de nuci si 70 de castane
se afişează
70

(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];
  cin.get(str,100);
  n=0;
  p = strtok(str, " ");
  while( p != NULL ) {
    n++; strcpy(v[n],p);
    p = strtok(NULL, " ");
  }

  char max='0'; //initializam valoarea maxima
  int poz=-1; //poz=-1 daca nu s-a gasit un numar si poz!=-1 daca s-a gasit cel putin un numar
  for(i=1;i<=n;i++)
     if(v[i][0]>='0' && v[i][0]<='9' && v[i][0]>max)
     {
     max=v[i][0];
     poz=i;
     }


if(poz!=-1) cout<<v[poz]<<endl;
    else cout<<"nu exista";

  return(0);
}
```

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

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

1. Subprogramele **f1** şi **f2** sunt definite mai jos.

```
int f1 (int x, int y)
{ if(x%2!=0 || y%2!=0)return 1;
  else return 2*f1(x/2,y/2);
}
```

```
int f2 (int x, int y)
{ if (x==y) return x;
  else
    if(x>y) return f2(x-y,y);
    else return f2(x, y-x);
}
```

Cel mai mare divizor comun al lui **30** și **50** se obține în urma apelului: **(4p.)**

a. **f1(30,50)** b. **f2(30,50)** c. **f1(30/2,50)** d. **f2(30/2,50)**

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

2. Utilizând metoda backtracking, se generează toate posibilitățile de a forma cutii cu bomboane de tipuri distincte din mulțimea {**fondante**, **caramele**, **dropsuri**, **acadele**}. Într-o cutie sunt cel puțin două tipuri de bomboane, dar nu pot fi și dropsuri și acadele simultan. Două cutii sunt distincte dacă ele conțin cel puțin un tip diferit de bomboane.
Primele patru soluţii generate sunt, în această ordine, (**fondante**, **caramele**), (**fondante**, **caramele**, **dropsuri**), (**fondante**, **caramele**, **acadele**), (**fondante**, **dropsuri**). Scrieţi a cincea și a șasea soluție, în ordinea generării acestora. **(6p.)**

(fondante, acadele)
(caramele, dropsuri)

3. Un număr natural este numit **echilibrat** dacă suma cifrelor sale de pe poziții pare este un număr par, iar suma cifrelor sale de pe poziţii impare este un număr impar. Pozițiile cifrelor sunt numerotate de la dreapta la stânga, astfel: cifra unităților este pe poziția **0**, cifra zecilor este pe poziția **1** ș.a.m.d.
Subprogramul **echilibrat** are un singur parametru, **n**, prin care primeşte un număr natural (**n∈[10,10^9]**). Subprogramul returnează valoarea **1** dacă **n** este echilibrat sau valoarea **0** în caz contrar.
Scrieţi definiţia completă a subprogramului.
Exemplu: dacă **n=25163912**, subprogramul returnează valoarea **1**, iar dacă **n=11211**, subprogramul returnează valoarea **0**. **(10p.)**

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

int echilibrat (unsigned long int n)
{
        int poz=0, spare=0, simpare=0, cifra;
        while(n!=0)
        {
        cifra=n%10;
        if(poz%2==0) spare=spare+cifra;
            else simpare=simpare+cifra;
        poz++;
        n=n/10;
        }
if(spare%2==0 && simpare%2==1) return 1;
    else return 0;
}

int main()
{
        unsigned long n;
        cin>>n;
        cout<<echilibrat(n);
        return 0;
}
```

4. Numim **secvență încadrată** a unui șir de numere naturale un subșir al acestuia, format din termeni aflați pe poziții consecutive în șirul dat, subșir care începe și se termină cu aceeași valoare. Lungimea secvenței este egală cu numărul de termeni ai acesteia.
Fișierul **bac.txt** conține un șir de cel puțin două și cel mult 10^6 numere naturale din intervalul [**0**, **9**]. Numerele sunt separate prin câte un spațiu. În șir există cel puțin doi termeni egali.
Se cere să se determine secvențele încadrate din acest șir care au lungime maximă și să se afișeze pe prima linie a ecranului lungimea maximă determinată, iar pe următoarea linie, pentru fiecare astfel de secvenţă, valoarea primului său termen. Numerele de pe a doua linie sunt afişate în ordine strict crescătoare, separate prin câte un spaţiu.
Proiectați un algoritm eficient din punctul de vedere al timpului de executare.
Exemplu: dacă fişierul **bac.txt** conţine numerele
3 1 5 2 4 5 5 2 5 9 5 7 4 6 8 0 8
atunci pe ecran se afișează valorile:
9
4 5

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 prim[11], ultim[11],x,dif;

int main()
{
int i, max=0,nr;
for(i=0;i<=9;i++){prim[i]=0;ultim[i]=0;}

i=0;
max=-1;
while(f>>x)
{
   i++;
              if(prim[x]==0) prim[x]=i;
                 else
                                        ultim[x]=i;
   if(max<ultim[x]-prim[x]+1) max=ultim[x]-prim[x]+1;
}
cout<<max<<endl;

for(i=0;i<=9;i++)
              if(max==ultim[i]-prim[i]+1)
                        cout<<i<<' ';
return 0;
}
```