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. Indicați o expresie C/C++ echivalentă cu cea alăturată. `(x>5) && (x<20) || (x!=y)`

a. `(x>5 || x<20) && (x==y)`
b. `!(x<=5 || x>=20) || (x!=y)`
c. `(x>5 || x<20) && (x!=y)`
d. `!(x<5 || x>20) && (x!=y)`

2. Subprogramul `f` este definit alăturat. Indicați valoarea `f(4770777,7)`.

```
int f (int n, int k)
{ if (n!=0)
    if(n%10==k) return 1+f(n/10,k);
  return 0;
}
```

a. 2 b. 3 c. 4 d. 5

f(4770777,7)=1+f(477077) =1+2=3
f(477077)=1+f(47707) =1+1=2
f(47707)=1+f(4770) =0+1=1
f(4770)=0

3. Variabila `x` este declarată alăturat. Indicați secvența care, în urma executării, memorează în variabila `x` ziua, luna și anul corespunzătoare unei date calendaristice citite de la tastatură.

```
struct data
      {
        int zi, luna, an;
      }x;
```

a. `cin>>x.zi>>x.luna>>x.an;` | `scanf("%d%d%d",&x.zi,&x.luna,&x.an);`
b. `cin>>zi.x>>luna.x>>an.x;` | `scanf("%d%d%d",&zi.x,&luna.x,&an.x);`
c. `cin>>x(zi,luna,an);` | `scanf("%d%d%d",&x(zi,luna,an));`
d. `cin>>x(zi)>>x(luna)>>x(an);` | `scanf("%d%d%d",&x(zi),&x(luna),&x(an));`

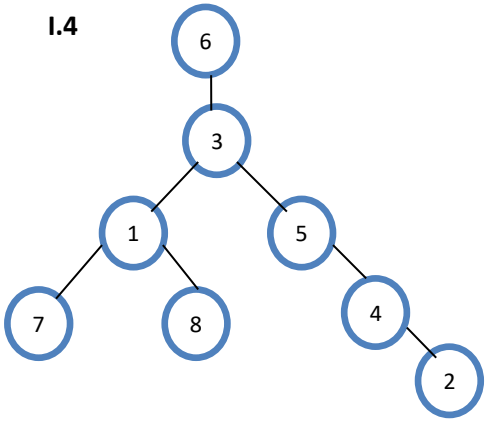
4. Un arbore cu rădăcină are `8` noduri, numerotate de la `1` la `8`, și muchiile `[1,3]`, `[1,7]`, `[1,8]`, `[2,4]`, `[3,5]`, `[3,6]`, `[4,5]`. Știind că rădăcina arborelui este nodul numerotat cu `6`, indicați nodurile de tip frunză ale arborelui dat.

a. `6,8` b. `2,6` c. `4,7,8` d. `2,7,8`

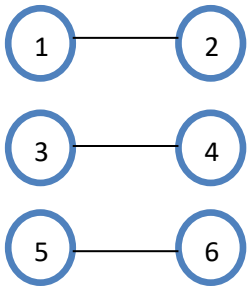
5. Un graf neorientat are `6` noduri și fiecare dintre acestea are gradul egal cu `1`. Indicați numărul de componente conexe ale grafului.

a. 1 b. 2 c. 3 d. 4

I.4



I.5



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`.

```
citește n (număr natural)
m←10
dacă n=0 atunci
 m←0
altfel
 repetă
  c←n%10; n←[n/10]
  dacă c<=m atunci m←c
  altfel m←-1
 până când n=0
scrie m
```

a. Scrieți numărul afișat în urma executării algoritmului dacă pentru `n` se citește valoarea `205579`. -1 **(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 `7`. 7777, 7789, 7999 **(6p.)**
numere din intervalul mentionat care
- au cifrele in ordine crescatoare
- cea mai mica cifra este 7

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 de alt tip. **(6p.)**

II.1.c

```
#include <iostream>
using namespace std;
int main()
{
   unsigned long int n;
   int m,k,c;
   cin>>n;
   m=10;
   if(n==0)  m=0;
    else
      do{
         c=n%10;
         n=n/10;
         if(c<=m)m=c;
         else m=-1;
        }while(n!=0);
   cout<<m;
   return 0;
}
```

II.1.d

```
citeste n
m=10
daca (n==0)
   atunci m=0
   altfel
      c=n%10
      n=n/10
      cat timp (n!=0) executa
         c=n%10
         n=n/10
         daca (c<=m)m=c
            altfel m=-1
scrie m
```

2. Utilizând metoda backtracking, sunt generate toate numerele din intervalul `[100,999]`, cu proprietatea că au cifrele în ordine crescătoare, cifrele aflate pe poziții consecutive sunt de paritate diferită, iar suma lor este egală cu `14`. Scrieți toate numerele generate, în ordinea obținerii lor. **(6p.)**

149, 167, 347

3. Variabilele `s1` și `s2` pot memora câte un șir de cel mult `50` de caractere. Scrieți ce se afișează în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s1,"bac2021");
cout<<strlen(s1)<<endl; | printf("%d\n",lengtf(s1));
strcpy(s2,s1+3); strcpy(s2+2,"20-");
strcat(s2,s1+3);
cout<<s2; | printf("%s",s2);
```

```
strcpy(s1,"bac2021");
cout<<strlen(s1)<<endl;
strcpy(s2,s1+3); strcpy(s2+2,"20-");
strcat(s2,s1+3);
cout<<s2; | printf("%s",s2);
```

s1="bac2021"
scrie 7

s2="2021" s2="2020-"
s2="2020-2021"
scrie 2020-2021

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

1. Subprogramul **divX** are doi parametri, **n** și **x**, prin care primește câte un număr natural din intervalul **[2,50]**. Subprogramul afișează pe ecran, în ordine descrescătoare, separate prin câte un spațiu, primele **n** numere naturale nenule divizibile cu **x**.
Scrieți definiția completă a subprogramului.
Exemplu: dacă **n=4** și **x=15** în urma apelului se afișează numerele **60 45 30 15** **(10p.)**

```
void divX(int n, int x)
{
  long int k=n*x;
  while(k>=x)
  {
    cout<<k<<' ';
    k=k-x;
  }
}
```

2. Scrieți un program C/C++ care citește de la tastatură numărul natural **n** ($n \in [2, 10^2]$) și elementele unui tablou bidimensional cu **n** linii și **n** coloane, numere naturale din intervalul $[0, 10^9]$.

Programul afișează pe ecran, separate prin câte un spațiu, elementele primului pătrat concentric, parcurs în sens invers al acelor de ceasornic, începând din colțul său stânga-sus, ca în exemplu. Primul pătrat concentric este format din prima și ultima linie, prima și ultima coloană a tabloului.
Exemplu: pentru **n=5** și tabloul alăturat, se afișează pe ecran numerele
1 2 3 4 5 6 7 8 9 0 2 4 6 8 1 3 **(10p.)**

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

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

unsigned long int a[100][100];
int n;
void citire()
{
  int i,j;
  cout<<"n=";cin>>n;
  for(i=0;i<n;i++)
     for(j=0;j<n;j++)
  {
    cout<<"a["<<i<<","<<j<<"]=";
    cin>>a[i][j];
  }
}
```

```
void parcurgere()
{
  int i;
  for(i=0;i<n;i++) cout<<a[i][0]<<' ';
  for(i=1;i<n;i++) cout<<a[n-1][i]<<' ';
  for(i=n-2;i>=0;i--) cout<<a[i][n-1]<<' ';
  for(i=n-2;i>0;i--)cout<<a[0][i]<<' ';
}
int main()
{
  citire();
  parcurgere();
  return 0;
}
```

3. Fișierul **bac.in** conține cel mult 10^6 numere naturale din intervalul $[0, 10^9]$, separate prin câte un spațiu. Se cere să se afișeze pe ecran, în ordine descrescătoare, cele mai mari două numere de două cifre distincte care **NU** se află în fișier. Numerele afișate sunt separate printr-un spațiu, iar dacă nu există două astfel de numere, se afișează pe ecran mesajul **nu exista**. Proiectați un algoritm eficient din punctul de vedere al timpului de executare.
Exemplu: dacă fișierul **bac.in** conține numerele **12 235 123 67 98 6 96 94 123 67 98 100** se afișează pe ecran, în această ordine, numerele **97 95**.
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.in");
int main()
{
   int v[100],k,x;
   unsigned long int y,x1,x2;
   for(x=0;x<=99;x++)v[x]=0;
   while(f>>y)
      if(y>=10 && y<=99)v[y]=1;
   k=2;
   x=98;
   while(x>=10 && k>=1)
   {
   if(v[x]==0 && x/10!=x%10)
      {
         if(k==2)x1=x;
         if(k==1)x2=x;
         k--;
      }
   x--;
   }
   if(k==0)cout<<x1<<' '<<x2;
   else cout<<"nu exista";
   return 0;
}
```

Am folosit un vector de aparitii v pentru fiecare numar din [10,98], unde:
v[i]=0 daca numarul i nu s-a gasit in sirul de numere citit; $i \in [10,98]$
v[i]=1 daca numarul I s-a gasit in sirul de numere citit; $i \in [10,98]$
98 este cel mai mare numar cu doua cifre distincte.
Pentru fiecare numar citit din fisier ce corespunde cerintelor (este format din doua cifre distincte) am completat, dupa caz, vectorul v pe pozitia data de valoarea numarului.
k=1 daca s-a gasit doar un numar in conditiile cerute.
k=0 daca s-au gasit doua numere in conditiile cerute.
Initial, k=2 pentru ca nu s-au gasit cele doua numere cerute; atunci cand se gaseste un numar ce indeplineste conditiile valoarea lui k se micsoreaza cu 1; in cazul in care am gasit cele doua valori nu mai cautam numere din [10,98] si trecem la afisarea celor doua numere retinute in variabilele x1 si respective x2, numere ce indeplinesc conditiile cerute; pozitiile din vector vor fi parcurse de la cea mai mare la cea mai mica, astfel, cele doua numere gasite sunt cele mai mari.
In cazul in care nu s-au gasit doua numere care sa indeplineasca conditiile se afiseaza mesajul corespunzator.

Pentru rezolvarea problemei am folosit un vector cu un numar constant si mic de elemente; elementele din fisier au fost citite doar o data; vectorul a fost parcurs o singura data, total sau partial, pentru a gasi cele doua elemente cerute. Neavand instructiuni repetitive imbricate, timpul necesar pentru executia programului este mic.
In program s-au folosit putine variabile; numarul acestora putea sa fie mai mic in cazul in care am fi folosit o variabila in mai multe scopuri; nu a fost necesar sa retinem toate numerele din fisier; numarul variabilelor folosite fiind mic si spatiul de memorie ocupat de acestea este mic.