Instrucțiunea FOR – Probleme cu divizorii unui număr

Forma generală

```
for (var1=expresie1; expresie2; expresie3)
     instrucțiuni;
```

Semnificație

Variabila **var1** primește valoarea **expresie1**. Cât timp **expresie2** este adevărată, se execută **instrucțiuni** și apoi se execută **expresie3.**

Exemplu: afișați toate numerele de la **1** la **n.**

```
for (i=1; i<=n; i++)
     cout<<i<<" ";
```

Valoarea inițială a lui **i** este **1.** Dacă condiția **i<=n** este adevărată, se afișează valoarea lui **i** și apoi se execută instrucțiunea **i ++.** Se continuă până când condiția **i<=n** nu mai este adevărată.

Probleme rezolvate

1. Se dă un număr natural **n**. Afișați valoarea sumei:
   S=1*2 + 2*3 + ... + **n*(n+1)**

```
#include <iostream>
using namespace std;
int main()
{
    int n,i;
    long long S=0;
    cin>>n;
    for(i=1;i<=n;i++)
        S=S+i*(i+1);
    cout<<S;
    return 0;
}
```

2. **Afișarea divizorilor unui număr.** Se citește un număr natural **n**. Afișați divizori acestuia.
   Exemplu: pentru **n=6** se va afișa: **1 2 3 6**

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    int d;
    cin>>n;
    for(d=1;d<=n;d++)
        if(n%d==0)
            cout<<d<<" ";
    return 0;
}
```

3. **Suma divizorilor unui număr.** Se citește un număr natural **n.** Afișați suma divizorilor acestuia.
   Exemplu: pentru **n=6,** se va afișa **12 (12=1+2+3+6).**

Varianta 1:

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    int d;
    long long S=0;
    cin>>n;
    for(d=1;d<=n;d++)
        if(n%d==0)
            S=S+d;
    cout<<S;
    return 0;
}
```

O variantă mai eficientă...

Pentru **n=12** vom avea următorii divizori: **1, 2, 3, 4, 6, 12.** Putem observa că, dacă **d** este divizor al lui **12**, atunci și **n/d** este divizor al lui **12.**

Un număr care este pătrat perfect are un număr impar de divizori. În consecință, după calculul sumei va trebui să avem în vedere acest caz particular, ceea ce înseamnă că trebuie să scădem din sumă valoarea radicalului lui **n.**

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

int main()
{
    long long n,s,d;
    cin>>n;
    s=0;
    for (d=1;d*d<=n;d++)
        if (n%d==0)
        {
            s=s+d+n/d;
        }
    //dacă n este pătrat perfect
    if (sqrt(n)==int(sqrt(n)))
           s=s-int(sqrt(n));
    cout <<s;
    return 0;
}
```

4. **Număr perfect.** Se citește un număr natural **n** cu cel mult 9 cifre. Afișați mesajul corespunzător dacă numărul citit este perfect sau nu**. Un număr este perfect dacă este egal cu suma divizorilor săi.**
Exemplu: dacă **n=6,** suma divizorilor este 1 + 2 + 3 = 6, deci se va afișa **6 este perfect.**

Numerele perfecte cunoscute se termină în 6 sau 8 și sunt foarte rare? Existența numerelor perfecte impare este încă o problemă nerezolvată a matematicii.

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    int d;
    long long S=0;
    cin>>n;
    for(d=1;d<=n/2;d++)
        if(n%d==0)
            S=S+d;
    if(S==n) cout<<n<<" este perfect";
    else cout<<n<<" nu este perfect";
    return 0;
}
```

5. Se citeşte un număr natural **n** (**n<=5).** Afişaţi primele **n** numere perfecte.
   Exemplu: pentru **n=4** se vor afişa: **6 28 496 8128.**

```
#include <iostream>
using namespace std;
int main()
{
    int n,x,i;
    int d;
    long long S=0;
    cin>>n;
    i=1;x=5;
    while(i<=n)
    {
        S=0;
        for(d=1;d<=x/2;d++)
            if(x%d==0)
                S=S+d;
        if(S==x)
        {
            i++;
            cout<<x<<" ";
        }
        x++;
    }
```

```
    return 0;
}
```

6. **Algoritmul lui Euclid pentru determinarea numerelor perfecte.** Se citește un număr natural **n** (**n<=5).** Afișați primele **n** numere perfecte.
   Euclid a constatat că, dacă 2^n-1 este număr prim atunci numărul $2^{n-1}*(2^n-1)$ este perfect.

n	2^n-1	$2^{n-1}*(2^n-1)$
2	3	6
3	7	28
4	15	120
5	31	496
6	63	2016
7	127	8128

număr prim număr perfect

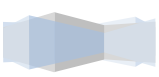
```
#include <iostream>
using namespace std;
int main()
{
    int n,p=1,d,ok;
    cin>>n;
    while(n)
    {
        ok=1;
        if(p-1<2) ok=0;
        else
            for(d=2;d*d<=p-1;d++)
                if((p-1)%d==0)
                    ok=0;
        if(ok==1)
        {
            n--;
            cout<<(p-1)*(p/2)<<" ";
        }
        p=p*2;
    }
    return 0;
}
```

7. **Număr prim. Un număr este prim dacă are exact doi divizori, unu și el însuși.** Se citește un număr întreg **n** cu cel mult 8 cifre**.** Afișați un mesaj dacă numărul este prim.

```
#include <iostream>
using namespace std;
int main()
{
    int n,nr=0;
    int d;
    cin>>n;
    for(d=1;d<=n;d++)
        if(n%d==0)
            nr++;
    if(nr==2) cout<<n<<" este prim";
    else cout<<n<<" nu este prim";
    return 0;
}
```

8. **Număr prim. Un număr este prim dacă nu are alți divizori în afară de unu și el însuși.** Se citește un număr întreg **n** cu cel mult 8 cifre**.** Afișați un mesaj dacă numărul este prim.

```
#include <iostream>
using namespace std;
int main()
{
    int n,ok=1;
    int d;
    cin>>n;
    if(n<2) ok=0;
    else
        for(d=2;d*d<=n;d++)
            if(n%d==0)
                ok=0;
    if(ok==1) cout<<n<<" este prim";
    else cout<<n<<" nu este prim";
    return 0;
}
```



9. **Model Bacalaureat 2019.** Subprogramul **CifrePrime** are un singur parametru, **n**, prin care primeşte un număr natural (**n**Є[0,109]). Subprogramul returnează suma cifrelor prime ale lui **n**.
   Scrieţi definiţia completă a subprogramului.
   Exemplu: dacă **n=1235405**, atunci subprogramul returnează **15**, iar dacă n=140, atunci subprogramul returnează 0..

```
int CifrePrime(int n)
{
    int s=0;
    while(n)
    {
        if(n%10==2 or n%10==3 or n%10==5 or n%10==7)
            s=s+n%10;
        n=n/10;
    }
    return s;
}
```

Probleme propuse

1. Se citește un număr natural nenul **n**. Afișați un mesaj dacă oglinditul acestuia este un număr prim.
2. Se dau **n** numere naturale nenule. Afișați media aritmetică a numerelor perfecte, dacă există, în caz contrar afișați mesajul **nu exista.**
3. Se citesc **n** numere naturale nenule. Afișați câte dintre numere au suma cifrelor un număr prim.
4. Se citesc **n** numere naturale nenule. Afișați cel mai mare număr prim și de câte ori apare printre numerele citite. Dacă nu există niciun număr prim se va afișa **-1.**
5. Afișați primele **n** numere prime.
6. Afișați numerele prime mai mici sau egale decât **n.**
7. Se dă un număr natural **n (n>2).** Afișați cel mai mic număr prim, mai mare strict decât **n** și cel mai mare număr prim, mai mic strict decât **n.**
8. Se dau două numere naturale **a** și **b.** Afișați numerele prime din intervalul [**a, b].** Dacă nu există numere prime în acest interval, se va afișa mesajul **nu sunt numere prime.**
9. **Pseudoperfect.** Se dă un număr natural **n.** Afișați un mesaj dacă acesta este pseudoperfect. Un număr este pseudoperfect dacă este divizor al sumei divizorilor săi.

10. **Numere prietene.** Se dau două numere **a** și **b**. Afișați un mesaj dacă acestea sunt prietene. Două numere se numesc prietene dacă fiecare număr este egal cu suma divizorilor celuilalt număr.
11. Se dă un număr natural nenul **n** cu maxim 9 cifre. Afișați dacă numărul obținut după eliminarea cifrelor pare ale sale este prim.
12. Se dă un număr natural nenul **n** cu maxim 9 cifre. Afișați dacă numărul obținut după eliminarea cifrelor impare ale sale este perfect.
13. **Aproape prim.** Se dă un număr natural nenul **n.** Afișați dacă el este aproape prim. Un număr este aproape prim dacă se poate scrie ca produs de două numere prime distincte.