Cel mai mare divizor comun. Cel mai mic multiplu comun

Algortimul lui Euclid prin scăderi repetate

Se scade numărul mai mic din numărul mai mare până când acestea devin egale.

```
if(a*b==0)cmmmdc=a+b;
else
{
     while(a!=b)
        if (a>b)
          a=a-b;
        else
          b=b-a;
     cmmdc=a;
}
```

Exemplu:

a=12	b=14
12	2
10	2
8	2
6	2
4	2
2	2

cmmdc=2

Observații:

1. Dacă unul dintre numere este 0, celălalt este cmmdc.
2. Două numere sunt prime între ele dacă au cmmdc 1.
3. Algoritmul prin împărțiri repetate este, de cele mai multe ori, mai rapid decât cel prin scăderi repetate.

Aplicație

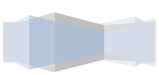
Se citesc două numere naturale **a** și **b.** Afișați cmmdc al celor două numere.

Variantă iterativă:

```
#include <iostream>
using namespace std;
int main()
{
    int a,b;
    cin>>a>>b;
    if(a*b==0)
        cout<<a+b;
    else
    {
        while(a!=b)
            if(a>b) a=a-b;
            else
                b=b-a;
        cout<<a;
    }
    return 0;
}
```

Variantă recursivă

```
#include <iostream>
using namespace std;
int cmmdc(int a, int b)
{
    if(a*b==0) return a+b;
    if(a>b) return cmmdc(a-b,b);
    else return cmmdc(a,b-a);
}
int main()
{
    int a,b;
    cin>>a>>b;
    cout<<cmmdc(a,b);
    return 0;
}
```



Algortimul lui Euclid prin împărțiri repetate

Se calculează restul împărțirii lui **a** la **b**. Primul număr ia valoarea celui de-al doilea, iar acesta valoarea restului calculat anterior. Se calculează un nou rest. Se repetă aceste instrucțiuni până când se obține un rest nul.

```
r=a%b;
while (r!=0)
{
      a=b;
      b=r;
      r=a%b;
}
```

Exemplu:

a=12	b=14	r=a%b
12	14	12
14	12	2
12	2	0

cmmdc=2

Aplicație

Se citesc două numere naturale **a** și **b.** Afișați cmmdc al celor două numere.

Variantă iterativă:

```
#include <iostream>
using namespace std;
int main()
{
    int a,b;
    cin>>a>>b;
    if(a*b==0)
        cout<<a+b;
```

```
    else
    {
        while(a!=b)
            if(a>b) a=a-b;
            else
                b=b-a;
        cout<<a;
    }
    return 0;
}
```

Variantă recursivă

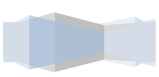
```
#include <iostream>
using namespace std;
int cmmdc(int a, int b,int r)
{
    if(a*b==0) return a+b;
    if(r==0) return b;
    else return cmmdc(b,r,a%b);
}
int main()
{
    int a,b,r;
    cin>>a>>b;
    r=a%b;
    cout<<cmmdc(a,b,r);
    return 0;
}
```

Cel mai mic multiplu comun a două numere

$$cmmmc(a,b) = \frac{a * b}{cmmdc(a,b)}$$

Probleme rezolvate

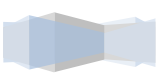
1. Se dau două numere naturale nenule **a** și **b**. Afișați divizorii comuni ai celor două numere.
   Observație: Divizorii comuni pe care îi pot avea în comun **a** și **b** sunt divizorii celui mai mare divizor comun al celor două numere.



```
#include <iostream>
using namespace std;
int cmmdc(int a, int b,int r)
{
    if(a*b==0) return a+b;
    if(r==0) return b;
    else return cmmdc(b,r,a%b);
}
int main()
{
    int a,b,r,x;
    cin>>a>>b;
    r=a%b;
    x=cmmdc(a,b,r);
    for(int d=1;d<=x;d++)
        if(a%d==0 and b%d==0)
            cout<<d<<" ";
    return 0;
}
```

2. Se dau două fracții de forma **a/b** și **c/d.** Afișați suma celor două fracții sub formă ireductibilă.
   Observație: pentru a aduce o fracție la forma ireductibilă, se simplifică numărătorul și numitorul fracției prin valoarea cmmmdc (numărător, numitor).

```
#include <iostream>
using namespace std;
int cmmdc(int a, int b)
{
    if(a*b==0) return a+b;
    if(a>b) return cmmdc(a-b,b);
    else return cmmdc(a,b-a);
}

int main()
{
    int a,b,c,d;
    cin>>a>>b;
    cin>>c>>d;
    int numitor=b*d/cmmdc(b,d);
```



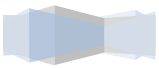
```
    int numarator=a*(numitor/b)+c*(numitor/d);
    int cmd=cmmdc(numarator,numitor);
    cout<<numarator/cmd<<"/"<<numitor/cmd;
    return 0;
}
```

3. Se dă un număr natural nenul **n** mai mic decât 2.000.000.000. Afișați câte perechi de numere prime între ele există printre primele **n** numere naturale.

```
#include <iostream>
using namespace std;
int cmmmdc(int a,int b, int r)
{
    if(a*b==0) return a+b;
    while(r!=0)
    {
        a=b;
        b=r;
        r=a%b;
    }
    return b;
}

int main()
{
    int a,b,n;
    cin>>n;
    int s=n;
    for(a=2;a<n;a++)
        for(b=a+1;b<=n;b++)
            if(cmmmdc(a,b, a%b)==1) s++;
    cout<<s;
    return 0;
}
```

4. Se citesc numere naturale până la întâlnirea lui 0. Afișați câte perechi de numere citite consecutiv au cmmmdc egal cu un **k** dat (0 nu se calculează!).

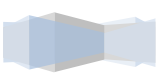
```
#include <iostream>
using namespace std;
int cmmmdc(int a,int b, int r)
```



```
{
    if(a*b==0) return a+b;
    while(r!=0)
    {
        a=b;
        b=r;
        r=a%b;
    }
    return b;
}

int main()
{
    int a,b,s=0,k;
    cin>>k;
    cin>>a;
    do{
        cin>>b;
        if(b>0)
        {
            if(cmmmdc(a,b,a%b)==k)
                s++;
        }
        a=b;
    }while(b!=0);

    cout<<s;
    return 0;
}
```

5. Se citesc numere naturale până la întâlnirea lui 0. Afișați câte dintre numerele citite sunt prime cu suma cifrelor lor.

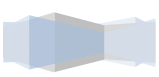
```
#include <iostream>
using namespace std;
int cmmmdc(int a,int b, int r)
{
    if(a*b==0) return a+b;
    while(r!=0)
    {
        a=b;
```



```
        b=r;
        r=a%b;
    }
    return b;
}
int sumcif(int n)
{
    int s=0;
    while(n)
    {
        s=s+n%10;
        n=n/10;
    }
    return s;
}
int main()
{
    int a,b,s=0;
    do{
        cin>>a;
        if(a>0)
        {
            b=sumcif(a);
            if(cmmmdc(a,b,a%b)==1)
                s++;
        }
    }while(a!=0);

    cout<<s;
    return 0;
}
```

6. Se citesc numere naturale până la întâlnirea lui 0. Afișați cmmmdc al acestor numere (0 nu intră în calcul!)..

```
# include <iostream>
#include <iomanip>
using namespace std;
int cmmmdc(int a,int b, int r)
{
    if(a*b==0) return a+b;
```



```
    while(r!=0)
    {
        a=b;
        b=r;
        r=a%b;
    }
    return b;
}
int sumcif(int n)
{
    int s=0;
    while(n)
    {
        s=s+n%10;
        n=n/10;
    }
    return s;
}
int main()
{
    int a,b,c;
    cin>>a;
    c=a;
    do{
        cin>>a;
        if(a>0)
        {
            c=cmmmdc(a,c,a%b);
        }
    }while(a!=0);

    cout<<c;
    return 0;
}
```

7. **Degustare de ciocolată – Concurs de admitere, Universitatea Babeș Bolyai Cluj, iulie, 2017.** O companie de publicitate face reclamă la un nou sortiment de ciocolată și intenționează să distribuie mostre de ciocolată la **n** ($10 \leq \mathbf{n} \leq 10000000$) copii care sunt așezați într-un cerc. Angajații companiei își dau seama că distribuirea de mostre tuturor copiilor ar costa foarte mult. În consecință, decid să distribuie mostre fiecărui al **k** lea ($0 < \mathbf{k} < \mathbf{n}$) copil din cei **n**, numărând copiii din **k** în **k** (atunci când numărătoarea ajunge la

ultimul copil, ea continuă cu primul copil și așa mai departe). În numărătoare se vor considera toți copiii, fie că au primit sau nu ciocolată. Numărătoarea se oprește atunci când o ciocolată ar trebui distribuită unui copil care deja a primit. Scrieți un subalgoritm care determină numărul copiilor (nr) care nu primesc mostre de ciocolată. Parametrii de intrare sunt numerele naturale **n** și **k**, iar parametrul de ieșire va fi numărul natural nr. Exemplu 1: dacă n = 12 și k = 9, atunci nr = 8 (primul, al 2-lea, al 4-lea, al 5-lea, al 7-lea, al 8-lea, al 10-lea, al 11- lea copil nu primesc ciocolată). Exemplu 2: dacă n = 15 și k = 7, atunci nr = 0 (toți copiii primesc ciocolată).

Răspuns: Ne imaginăm copiii așezați într-un șir lung, multiplu de **n.** Fiecare al **k**-lea copil primește ciocolată. În consecință, va exista o poziție din acest șir care va fi multiplu de **n** și de **k.** Acesta este **cmmmc (n,k).** Un număr de **cmmmc(n,k)/k** de copii va primi ciocolată. Copiii care nu primesc ciocolată vor fi: **n-cmmmc(n,k)/k.**

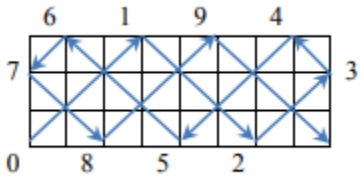
```
#include <iostream>
using namespace std;
int cmmdc(int a, int b)
{
    if(a*b==0) return a+b;
    if(a>b) return cmmdc(a-b,b);
    else return cmmdc(a,b-a);
}

int main()
{
    int n,k,c;
    cin>>n>>k;
    c=n*k/cmmdc(n,k);
    cout<<n-c/k;
    return 0;
}
```

Probleme propuse

1. Se dau două fracții de forma **a/b** și **c/d.** Afișați diferența celor două fracții sub formă ireductibilă.
2. Se citesc perechi de numere naturale nenule **a, b** până când sunt introduse două numere prime între ele. Afișați suma fracțiilor constituite cu fiecare pereche **a/b** (exclusiv ultima pereche citită!).

3. În portul Constanţa sunt ancorate două vapoare pline cu marfă. Ele fac curse repetate către două destinaţii diferite. Se ştie că primul vapor ajunge la destinaţia stabilită după un număr *X* de săptămâni, iar al doilea vapor după un număr *Y* de săptămâni. Drumul înapoi ia acelaşi timp. Armatorul celor *2* vapoare vrea să ştie după câte zile cele *2* vapoare pleacă din nou concomitent din port. Se mai ştie faptul că pentru manevrarea mărfurilor primului vapor îi sunt necesare *z1* zile, iar celui de al doilea *z2* zile. Scrieţi un program care determină numărul de zile după care cele *2* vapoare pleacă din nou concomitent din portul din care au plecat. (*Olimpiada Municipală de Informatică Iași, clasa a 6-a, 2015 – enunț preluat de pe site-ul www.pbinfo.ro, problema cu id-ul #2114).*

4. **Rază – Concurs UBB Cluj, 2017, Facultatea de Matematică și Informatică.** Avem la dispoziție un chenar dreptunghiular format din oglinzi. O rază de lumină pornește din colțul stânga jos al dreptunghiului sub un unghi de 45º față de latura de jos a dreptunghiului și lovește latura de sus sau latura din dreapta. Aici se reflectă (pornește spre o altă latură tot sub un unghi de 45º față de latura de care s-a lovit). Își continuă drumul până când ajunge într-un colț al dreptunghiului.



Scrieți un subalgoritm care calculează de câte ori **(nrSchimb**) raza își schimbă direcția de mers până când se oprește într-un colț. Punctul de pornire nu se numără. Parametri de intrare ai subalgoritmului sunt lungimea (**1 < a < 10 000**) și lățimea (**1 < b < 10 000**) dreptunghiului, iar **nrSchimb** va fi parametru de ieșire.
Exemplu 1: dacă a = 8 și b = 3, atunci nrSchimb = 9. Exemplu 2: dacă a = 8 și b = 4, atunci nrSchimb = 1.