📕前置芝士 —— 莫比乌斯函数

🎈定义

$$\mu(n)=\begin{cases}1 & n=1\\ (-1)^k & n\text{无平方因数，}n=p_1p_2p_3...p_k\\ 0 & n\text{有大于1的平方因数}\end{cases}$$

可以简化为：
在n无平方因数时：$\mu(n)=(-1)^{n\text{的不同质因子的个数}}$
其他情况：$\mu(n)=0$

🎈性质

正常情况下在n有x数个不同质因子，m有y数个不同质因子时
1.x奇，y奇，n * m的质因子个数 = x + y = 偶，$\mu(n)*\mu(m)=(-1)*(-1)=1$

2.x奇，y偶，n * m的质因子个数 = x + y = 奇，$\mu(n) * \mu(m) = (-1) * 1 \quad = -1$
3.x偶，y奇，n * m的质因子个数 = x + y = 奇，$\mu(n) * \mu(m) = \quad 1 \quad * (-1) = -1$
4.x偶，y偶，n * m的质因子个数 = x + y = 偶，$\mu(n) * \mu(m) = \quad 1 \quad * 1 \quad = 1$
可以看出莫比乌斯函数是个积性函数

但是特殊情况例如 $n = m = 2$ 时
$\mu(n) = \mu(m) = -1$
$\mu(n * m) = 0 \ != (-1) * (-1) = \mu(n) * \mu(m)$
所以莫比乌斯函数不是完全积性函数

🎈利用

$$\sum_{d|n} \mu(d) = [n = 1]$$

例如 n = 12时
$\sum_{d|12} \mu(12) = \mu(1) + \mu(2) + \mu(3) + \mu(4) + \mu(6) + \mu(12) = 1 + (-1) + 1 + 0 + 1 + 0$

🎈程序

线性筛打表：

```
const int maxn = 2005;

bool isprime[maxn];
ll mu[maxn];//Mobius函数表
vector<ll> prime;

inline void Mobius(){//线性筛
        isprime[0] = isprime[1] = 1;
        mu[1] = 1;//特判mu[i] = 1
```

```
        for(ll i = 2; i <= maxn; i ++){
                if( !isprime[i] ) mu[i] = -1,
prime.push_back(i);//质数的质因子只有自己，所以为-1
                for(ll j = 0; j < prime.size() && i * prime[j]
<= maxn; j ++){
                        isprime[i * prime[j]] = 1;
                        if(i % prime[j] == 0) break;
                        mu[i * prime[j]] = -mu[i];//积性函数性质:
(i * prime[j])多出来一个质数因数(prime[j])，修正为 (-1) * mu[i]
                }
        }
        //剩余的没筛到的是其他情况，为0
}
```

📕概述

🎈概念

莫反是一种利用莫比乌斯函数的积性性质，对方程进行计算用时简化的一种方法

🎈思想

（上文中性质的利用）

🎈反演定理

设有两个方程 $f(x)$ 和 $F(x)$ ，有以下两种反演方式

1. $$F(n)=\sum_{d|n}f(d)$$

$$\Downarrow$$

$$f(n)=\sum_{d|n}\mu(d)F(\frac{n}{d})$$

> 证明：
>
> $$\sum_{d|n}\mu(d)F(\frac{n}{d})=\sum_{d|n}\mu(d)\sum_{i|\frac{n}{d}}f(i)=\sum_{i|n}f(i)\sum_{d|\frac{n}{i}}\mu(d)=f(n)$$

2. $$F(n)=\sum_{n|d}f(d)$$

$$\Downarrow$$

$$f(n)=\sum_{n|d}\mu(\frac{d}{n})F(d)$$

> 证明
>
> $$\sum_{n|d}\mu(\frac{d}{n})F(d)=\sum_{n|d}\mu(\frac{d}{n})\sum_{d|i}f(n)=(d'=\frac{d}{n})\sum_{n|i}f(i)\sum_{d'|\frac{i}{n}}\mu(d')=f(n)$$

📕实例

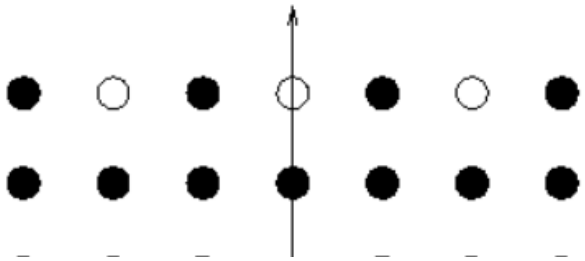
🎈题目

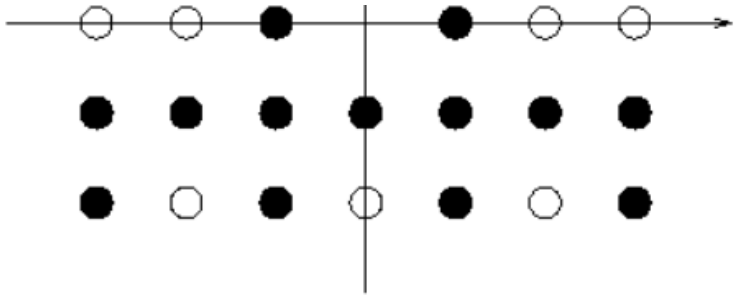
UVA10214 《Trees in a Wood.》传送门

The saying *"You can't see the wood for the trees"* is not only a cliche, but is also incorrect. The real problem is that you can't see the trees for the wood. If you stand in the middle of a wood, the trees tend to obscure each other and the number of distinct trees you can actually see is quite small. This is especially true if the trees are planted in rows and columns, because they tend to line up. The purpose of this problem is to find how many distinct trees one can see if one were standing on the position of a tree in the middle of the wood.

For a mathematically more precise description we assume that you are situated at the origin of a coordinate system in the plane.

Trees are planted at all positions $(x, y) \in \mathbb{Z} \times \mathbb{Z} \backslash \{(0, 0)\}$, with $|x| \leq a$ and $|y| \leq b$.





A tree at position (x, y) can be seen from the origin if there are no other trees on the straight line from $(0, 0)$ to (x, y). Find the number K of all the trees in the wood that can be seen from the origin and the number N of all the trees to compute the fraction $\frac{K}{N}$.

Hint: The Euler phi function $\varphi(n)$ is defined to be the number of integers m in the range $1 \leq m \leq n$ relatively prime to n:

$$\varphi(n) = \#\{m \mid 1 \leq m \leq n \text{ and } \gcd(m, n) = 1\} \qquad (\gcd = \textit{greatest common divisor})$$

Instead of counting (an adequate method for small n!) you could as well use the following identity:

$$\varphi(n) = n \prod_{p \in P, p|n} (1 - \frac{1}{p}), \ P = \{p \in \mathbb{N} | p \text{ prime}\}$$

Hint: Remember that $\gcd(m, n) = \gcd(m + n, n) = \gcd(m + 2n, n) = \gcd(m + in, i)$

You might be surprised that the fraction $\frac{K}{N}$ converges to $\frac{6}{\pi^2} \approx 0.607927$ for an infinitely large wood.

Input

Each scenario consists of a line with the two numbers a and b (separated by a white space), with $1 \leq a \leq 2000$ and $1 \leq b \leq 2000000$. Input is terminated by a line with two zeros.

Output

For each scenario print a line containing the fraction with a precision of 7 digits after the decimal point. Error less than 2e-7 or $2 * 10^{-7}$ will be tolerated.

Sample Input

```
3 2
0 0
```

Sample Output

```
0.7058824
```


思路

与[SDOI2008]仪仗队很像 在一个象限内
都是让求

$$\sum_{i=1}^{N}\sum_{j=1}^{M}[gcd(i,j)=1]$$

所以我们设置

$$f(n)=\sum_{i=1}^{N}\sum_{j=1}^{M}[gcd(i,j)=n],\quad f(1)=?$$

但是因为 $f(1)$ 比较难求，所以我们同时要设置一个满足 $F(n)=\sum_{n|d}f(d)$ 的 $F(n)$

$$F(n)=\sum_{i=1}^{N}\sum_{j=1}^{M}[n|gcd(i,j)],\quad F(1)=\sum_{i=1}^{N}\sum_{j=1}^{M}1$$

$$\therefore F(n)=\sum_{n|d}f(d),\quad F(1)=\sum_{d=1}^{min(N,M)}f(d)$$

$$\therefore f(n)=\sum_{n|d}\mu(\frac{d}{n})F(d),\quad f(1)=\sum_{d=1}^{min(N,M)}\mu(d)F(d)$$

$$\because 1\le d\le min(N,M)$$

$$\therefore F(d)=\left\lfloor\frac{N}{d}\right\rfloor*\left\lfloor\frac{M}{d}\right\rfloor$$

$$\therefore f(1) = \sum_{d=1}^{min(N,M)} \mu(d) * \left\lfloor \frac{N}{d} \right\rfloor * \left\lfloor \frac{M}{d} \right\rfloor$$

由于四个象限 + 四个坐标轴，所以分子为 $4 * \sum_{d=1}^{min(N,M)} \mu(d) * \lfloor \frac{n}{d} \rfloor * \lfloor \frac{m}{d} \rfloor + 4$
分母则是所有的树 $(N * 2 + 1) * (M * 2 + 1) - 1$

答案则是 $\frac{4 * \sum_{d=1}^{min(N,M)} \mu(d) * \lfloor \frac{n}{d} \rfloor * \lfloor \frac{m}{d} \rfloor + 4}{(N * 2 + 1) * (M * 2 + 1) - 1}$ 保留7位小数

🎈 程序

```
/*
          ________   _
________                              _
          /  ______| | |
|   __   |                           | |
         /  /        | |
|  |__|  |                           | |
         |  |        | |___    _   _   _   ___  _   _____
|     ___|   ______   _____   ___  _  | |
         |  |        |  __ \  |_| | | | | |  _\| | | ____|
|  |\  \    |  __  | |  _  | |  _\| | | |
         |  |        | |  \ |  _  | | | | | | \  | | \___
|  | \  \   | |_/ _| | |_| | | | \  | | |
         \  \______  | |  | | | | \ |_| / | |_/  |  ___/ |
|  |  \  \  |    /_   \__  | | |_/  | | |
Author :  \________| |_|  |_| |_|  \___/  |___/|_| |_____|
_________|__|   \__\ |______|     | | |___/|_| |_|

____| |

\_____/
*/
#include <unordered_map>
```

```
#include <algorithm>
#include <iostream>
#include <cstring>
#include <utility>
#include <string>
#include <vector>
#include <cstdio>
#include <stack>
#include <queue>
#include <cmath>
#include <map>
#include <set>

#define G 10.0
#define LNF 1e18
#define EPS 1e-6
#define PI acos(-1.0)
#define INF 0x7FFFFFFF

#define ll long long
#define ull unsigned long long

#define LOWBIT(x) ((x) & (-x))
#define LOWBD(a, x) lower_bound(a.begin(), a.end(), x) -
a.begin()
#define UPPBD(a, x) upper_bound(a.begin(), a.end(), x) -
a.begin()
#define TEST(a) cout << "---------" << a << "---------" << '\n'

#define CHIVAS_ int main()
#define _REGAL exit(0)

#define SP system("pause")
#define IOS ios::sync_with_stdio(false)
//#define map unordered_map

#define _int(a) int a; cin >> a
#define  _ll(a) ll a; cin >> a
#define _char(a) char a; cin >> a
#define _string(a) string a; cin >> a
#define _vectorInt(a, n) vector<int>a(n); cin >> a
```

```
#define _vectorLL(a, b) vector<ll>a(n); cin >> a

#define PB(x) push_back(x)
#define ALL(a) a.begin(),a.end()
#define MEM(a, b) memset(a, b, sizeof(a))
#define EACH_CASE(cass) for (cin >> cass; cass; cass--)

#define LS l, mid, rt << 1
#define RS mid + 1, r, rt << 1 | 1
#define GETMID (l + r) >> 1

using namespace std;

template<typename T> inline void Read(T &x){T f = 1; x = 0;char
s = getchar();while(s < '0' || s > '9'){if(s == '-') f = -1; s =
getchar();}while('0'<=s&&s<='9'){x=(x<<3)+(x<<1)+
(s^48);s=getchar();}x*=f;}
template<typename T> inline T MAX(T a, T b){return a > b? a :
b;}
template<typename T> inline T MIN(T a, T b){return a > b? b :
a;}
template<typename T> inline void SWAP(T &a, T &b){T tp = a; a =
b; b = tp;}
template<typename T> inline T GCD(T a, T b){return b > 0? GCD(b,
a % b) : a;}
template<typename T> inline void ADD_TO_VEC_int(T &n, vector<T>
&vec){vec.clear(); cin >> n; for(int i = 0; i < n; i ++){T x;
cin >> x, vec.PB(x);}}
template<typename T> inline pair<T, T> MaxInVector_ll(vector<T>
vec){T MaxVal = -LNF, MaxId = 0;for(int i = 0; i <
(int)vec.size(); i ++) if(MaxVal < vec[i]) MaxVal = vec[i],
MaxId = i; return {MaxVal, MaxId};}
template<typename T> inline pair<T, T> MinInVector_ll(vector<T>
vec){T MinVal = LNF, MinId = 0;for(int i = 0; i <
(int)vec.size(); i ++) if(MinVal > vec[i]) MinVal = vec[i],
MinId = i; return {MinVal, MinId};}
template<typename T> inline pair<T, T> MaxInVector_int(vector<T>
vec){T MaxVal = -INF, MaxId = 0;for(int i = 0; i <
(int)vec.size(); i ++) if(MaxVal < vec[i]) MaxVal = vec[i],
MaxId = i; return {MaxVal, MaxId};}
template<typename T> inline pair<T, T> MinInVector_int(vector<T>
```

```
vec){T MinVal = INF, MinId = 0;for(int i = 0; i <
(int)vec.size(); i ++) if(MinVal > vec[i]) MinVal = vec[i],
MinId = i; return {MinVal, MinId};}
template<typename T> inline pair<map<T, T>, vector<T> > DIV(T n)
{T nn = n;map<T, T> cnt;vector<T> div;for(ll i = 2; i * i <= nn;
i ++){while(n % i == 0){if(!cnt[i]) div.push_back(i);cnt[i] ++;n
/= i;}}if(n != 1){if(!cnt[n]) div.push_back(n);cnt[n] ++;n /=
n;}return {cnt, div};}
template<typename T>             vector<T>& operator--
(vector<T> &v){for (auto& i : v) --i;            return  v;}
template<typename T>             vector<T>& operator++
(vector<T> &v){for (auto& i : v) ++i;            return  v;}
template<typename T>             istream& operator>>(istream&
is,  vector<T> &v){for (auto& i : v) is >> i;        return is;}
template<typename T>             ostream& operator<<(ostream&
os,  vector<T>  v){for (auto& i : v) os << i << ' '; return os;}

const int maxn = 2005;

bool isprime[maxn];
ll mu[maxn];//Mobius函数表
ll n, m;
vector<ll> prime;

inline void Mobius(){//线性筛
        isprime[0] = isprime[1] = 1;
        mu[1] = 1;//特判mu[i] = 1
        for(ll i = 2; i <= maxn; i ++){
                if( !isprime[i] ) mu[i] = -1,
prime.push_back(i);//质数的质因子只有自己，所以为-1
                for(ll j = 0; j < prime.size() && i * prime[j]
<= maxn; j ++){
                        isprime[i * prime[j]] = 1;
                        if(i % prime[j] == 0) break;
                        mu[i * prime[j]] = -mu[i];//积性函数性质:
(i * prime[j])多出来一个质数因数(prime[j])，修正为 (-1) * mu[i]
                }
        }
        //剩余的没筛到的是其他情况，为0
}
```

```
inline void solve(){
        ll res = 0;
        for(ll d = 1; d <= MIN(n, m); d ++){
                res += mu[d] * (n / d) * (m / d);
        }
        res = res * 4 + 4;//四个象限 + 坐标轴的四个贡献
        ll down = (n * 2 + 1) * (m * 2 + 1) - 1;//分母，矩阵所有树
- 原点
        printf("%.7f\n", res * 1.0 / down);
}

CHIVAS_{Mobius();
        while(scanf("%lld%lld", &n, &m) == 2, n || m){
                solve();
        }
        _REGAL;
}
```