Koishi 与正方形

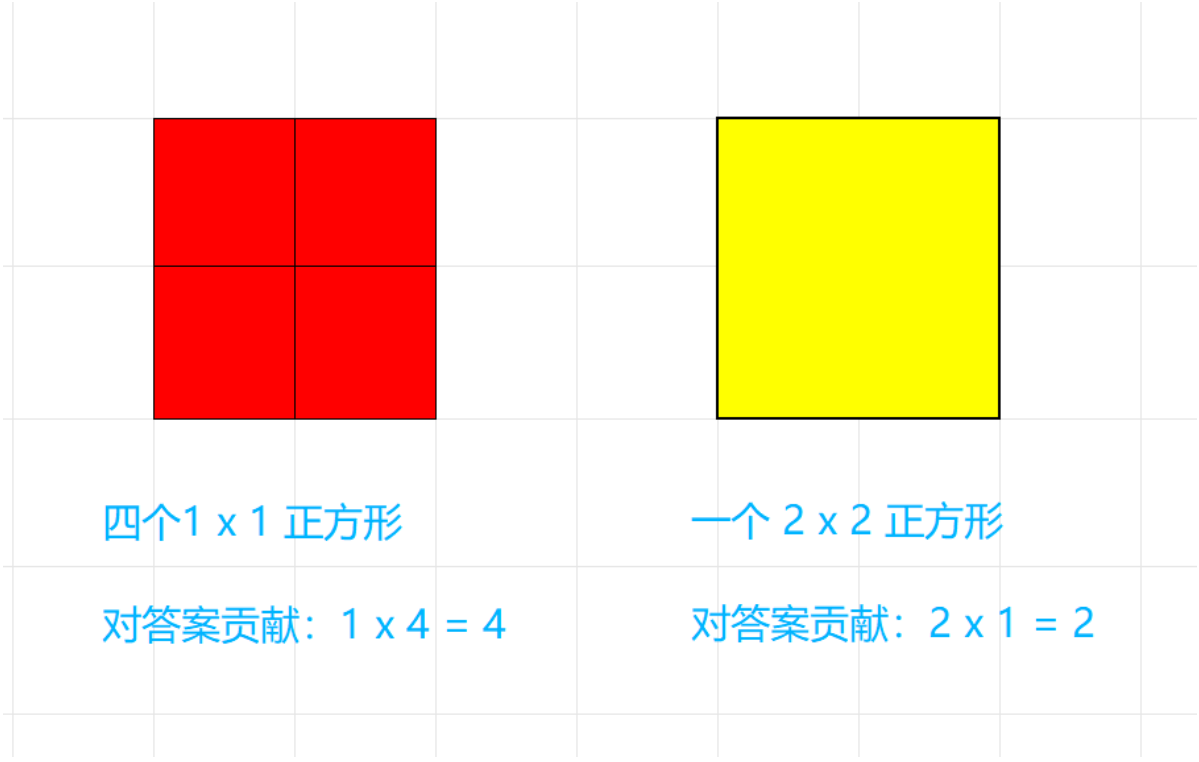
题意简述

有一张 $n \times m$ 的矩形方格纸，要求划分为诺干个正方形，使得正方形的单边长之和最小。

题目解析

想让划分出来的边长尽可能小，就要**划分尽可能大的正方形。**

举个例子。如下图，划分 2×2 的正方形对答案的贡献是 2，但是划分 4 个 1×1 的正方形对答案的贡献是 4，显然不优秀。



于是你想，怎么样让划分出来的正方形尽可能大?

你拿到的方格纸是长方形的，不妨假设边长分别为 a, b 且 $a > b$，那你肯定至少能划分出一个 $b \times b$ 大小的正方形。

巧了，这个 $b \times b$ 大小的正方形，不就是在 $a \times b$ 大小的长方形纸上能划分出的最大的正方形吗。

于是你不断去用 $b \times b$ 大小的正方形去切割，直到长边不够用了，这时候新的 a', b 大小的长方形就产生了。

由于上一轮的划分，此时应该是 $a' < b$ 的。那我们照搬上一轮的策略，用 $a' \times a'$ 大小的正方形去划分。

如此循环，直到恰好被划分完，就是正确答案。

在代码的实现上，可以采用递归来做。

具体来讲就是先划分，然后得到新的边长，然后再用新的边长进一步递归，直到划分完。

划分的过程，不要一个一个正方形边长的去扣，那样会超时。直接 $a' = a \mod b$ 即可。在过程中记得累加边长到全局的答案变量中。

另外，还是要记得开 long long。

代码

```
#include<iostream>
#include<cstdio>
#define ll long long
using namespace std;
ll ans;
void q(ll x,ll y){
   if(x==0||y==0) return; //已经被划分完了，不能再划分了
   if(x==y) {   //剩下的纸张大小是正方形，直接记入答案即可
    ans+=x;
    return;
   }
   if(y>x) {    //找到长边，用短边去划分
    ans=ans+y/x*x;       //统计答案
    q(x,y%x);        //计算新的边长，递归处理
    return;
   } else {
    ans=ans+x/y*y;  //另一边是长边的情况
    q(x%y,y);
    return;
   }
}
int main(){
    ll n,m;
    scanf("%lld%lld",&n,&m);
    q(n,m);
    printf("%lld",ans);
    return 0;
}
```

补充

这里面运用了 **贪心** 算法的思想。具体就是通过局部最优解来达成全局的最优解。

划分的过程，有点像求最大公约数的 **辗转相除法**。

感兴趣的同学可以去进一步了解。