#include<bits/stdc++.h>

#define is\_in\_range(xval, yval, xmin, ymin, xmax, ymax) (((((xval)>=xmin) and (xval)<=xmax) and ((yval)>=ymin) and ((yval)<=ymax))? 1: 0)

using namespace std;

pair<int, int> extendedEuclidean(int a, int b)

{

pair<int, int> p, q;

if(a%b==0)

{

p.first=0;

p.second=1;

return p;

}

q=extendedEuclidean(b, a%b);

p.first=q.second;

p.second=q.first+(-a/b)\*q.second;

return p;

}

pair<int, int> extendedEuclideanCover(int a, int b)

{

pair<int, int> p;

p=extendedEuclidean(abs(a), abs(b));

if(a<0) p.first=-p.first;

if(b<0) p.second=-p.second;

return p;

}

pair<int, int> find\_one\_solution\_of\_linear\_diophantine\_equation(int a, int b, int c)

{

pair<int, int> p, q;

int g=\_\_gcd(abs(a) , abs(b));

if(c%g!=0)

{

cout<<"no solution"<<endl;

return p;

}

q=extendedEuclideanCover(a,b);

p.first=q.first\*c/g;

p.second=q.second\*c/g;

return p;

}

int number\_of\_solution(int a,int b,int c,int xmin,int ymin,int xmax, int ymax)

{

if(a<0)

{

int temp=xmax;

a=-a;

xmax=-xmin;

xmin=-temp;

}

if(b<0)

{

int temp=xmax;

b=-b;

ymax=-ymin;

ymin=-temp;

}

pair<int, int> p;

p=find\_one\_solution\_of\_linear\_diophantine\_equation(a, b, c);

int g=\_\_gcd(abs(a), abs(b));

int x0=p.first;

int y0=p.second;

int ap=a/g;

int bp=b/g;

int minm1=ceil((double)(y0-ymax)/ap);

int minm2=ceil((double)(xmin-x0)/bp);

int minm=max(minm1, minm2);

int maxm1=floor((double)(y0-ymin)/ap);

int maxm2=floor((double)(xmax-x0)/bp);

int maxm=min(maxm1, maxm2);

return maxm-minm+1;

}

int main()

{

pair<int , int>p;

int a, b, c, xmin, ymin, xmax, ymax;

cin>>a>>b>>c>>xmin>>xmax>>ymin>>ymax;

cout<<number\_of\_solution(a, b, -c, xmin, ymin, xmax, ymax)<<endl;

return 0;

}