原创 欧几里德算法+扩展欧几里德算法

2019-05-03 15:23:20 _-Y-_-Y-_ 阅读数 44 更多 编辑

欧几里德算法

$$GCD(a,b)=\begin{cases} b & a=0 \\ GCD(b \bmod a,a) & \text{否则} \end{cases}=\begin{cases} a & b=0 \\ GCD(b,a \bmod b) & \text{否则} \end{cases}$$

证明

证明欧几里德算法的关键是要证明$gcd(a,b)=gcd(b\ mod\ a,a)$
$b\ mod\ a$等价于$b-\lfloor b/a \rfloor \times a$
$b-\lfloor b/a \rfloor \times a$能被$gcd(a,b)$整除
又因为a和b都能被$gcd(a,b)$整除
所以$b\ mod\ a$和a也能被$gcd(a,b)$整除

code

```
int gcd(int a,int b){
  return b==0?a:gcd(b,a%b);
}
```

扩展欧几里德算法

如果a和b都是整数，则有整数x和y 使得 $ax+by=GCD(a,b)$
推论 若a,b互素，则存在x和y使得$ax+by=1$

证明

设c是a和b的线性组合中最小整数
$ax+by=c$（x，y为整数）
令 $a=cq+r(0\le r<c)$
可得$r=a-cq=a(1-qx)-bqy$
所以 r 是 a 和 b 的线性组合
又因为 c是a和b的线性组合中最小整数
所以 r == 0
所以 c 是 a 的约数
同理 c 是 b 的约数
所以 c 是 a 和 b 的公约数
对于 a 和 b 的所有约数 d
因为 ax + by = c
所以 d 是 c 的约数$c \ge d$;
所以 c 是 最大公约数 $GCD(a,b)$

应用

因为$ax_1+by_1=GCD(a,b),ax_2+GCD(b,a\%b)y_2=GCD(b,a\%b)=GCD(a,b)$
所以$ax_1+by_1=bx_2+(a-\lfloor a/b \rfloor * b)y_2=ay_2+b(x_2-\lfloor a/b \rfloor * y_2)$
所以$x_1=y_2,y_1=x_2-\lfloor a/b \rfloor * y_2$
重复这一过程直到$b==0$此时$x=1,y=0$。

结论

$x_1 = y_2$
$y_1 = x_2 - \lfloor a/b \rfloor * y_2$
当c是gcd(a,b)的倍数时
$x = x_0 + k * \lfloor b/gcd(a,b) \rfloor$
$y = y_0 - k * \lfloor a/gcd(a,b) \rfloor$
若c不是gcd(a,b)的倍数时
则$ax + by = c$无整数解

code

```
int exgcd(int a,int b,int &x, int &y){
  if(b==0){ x=1,y=0;return a;}
  int t=exgcd(b,a%b,x,y);
  int x0=x,y0=y;
  x=y0,y=x0-(a/b)*y0;
  return t;
}
```

&是引用的符号，一变全变

例1

https://cn.vjudge.net/problem/HDU-2669

AC code

```
#include <bits/stdc++.h>
#define ll long long
using namespace std;
ll exgcd(ll a,ll b,ll &x, ll &y){
  if(b==0){ x=1,y=0;return a;}
  ll t=exgcd(b,a%b,x,y);
  ll x0=x,y0=y;
  x=y0,y=x0-(a/b)*y0;
  return t;
}
int main(){
  ll a,b,x,y;
  while(cin>>a>>b){
    if(exgcd(a,b,x,y)==1){
      int c=exgcd(a,b,x,y);
      while(x<=0){
            x=x+b/c;
        y=y-a/c;
          }
      cout<<x<<" "<<y<<endl;
    }
    else cout<<"sorry"<<endl;
  }
  return 0;
}

```

例2

https://cn.vjudge.net/problem/POJ-2142

AC code

```
#include <cstdio>
#include <iostream>
using namespace std;
int exgcd(int a,int b,int &x, int &y){
  if(b==0){ x=1,y=0;return a;}
  int t=exgcd(b,a%b,x,y);
  int x0=x,y0=y;
  x=y0,y=x0-(a/b)*y0;
  return t;
}
```

```
int main(){
  int a,b,d,x,y;
  while(~scanf("%d %d %d", &a, &b, &d)&&(a||b||d)){
    int md=exgcd(a,b,x,y);
    a/=md,b/=md,d/=md;
    int x1=x*d;
    x1=(x1%b+b)%b;
    int y1=(d-x1*a)/b;
    y1=abs(y1);
    int y2=y*d;
    y2=(y2%a+a)%a;
    int x2=(d-y2*b)/a;
    x2=abs(x2);
    if(x1+y1<x2+y2) printf("%d %d\n",x1,y1);
    else printf("%d %d\n",x2,y2);
  }
  return 0;
}

```

例3

https://cn.vjudge.net/problem/ZOJ-3593

code

```
#include <cstdio>
#include <iostream>
#define INF 0x3fffffff
using namespace std;
long long exgcd(long long a,long long b,long long &x, long long &y){
  if(b==0){ x=1,y=0;return a;}
  long long t=exgcd(b,a%b,x,y);
  long long x0=x,y0=y;
  x=y0,y=x0-(a/b)*y0;
  return t;
}
long long abs1(long long a){
    if(a<0) return -a;
    return a;
}
long long judge(long long x,long long y){
  if(abs1(x+y)!=abs1(x)+abs1(y)) return abs1(x)+abs1(y);
  return max(abs1(x),abs1(y));
}
int main(){
    long long A,B,a,b,d,x,y;
    int T;
    scanf("%d", &T);
    while(T--){
        scanf("%lld %lld %lld %lld", &A, &B, &a, &b);
        long long md=exgcd(a,b,x,y);
        d=abs1(B-A);
        if(d%md!=0){
            printf("-1\n");
            continue;
        }
        a/=md,b/=md;
    x*=d/md;
    y*=d/md;
        long long mid = (y - x) / (a + b);//当x和y最接近的时候，|x|+|y|最小
    long long ans = (long long)INF * (long long)INF;
    for(long long t=mid-2;t<=mid+2;t++){
        ans=min(ans,judge(x+b*t,y-a*t));
    }
    printf("%lld\n", ans);
    }
}

```

有 0 个人打赏

文章最后发布于: 2019-05-03 15:23:20