<http://paste.ubuntu.com/24375097/>

*/\**  
  
*Solves x in a^x=b (mod m) when a, b*  
*and m are known in O( sqrt(m)\*lg(sqrt(m))*  
  
*Let x= q\*(sqrt(m)) + r*  
  
*Thus (a^(q\*sqrt(m))) \* (a^r) = b(mod m)*  
  
*=> a^r = (b/a^(q\*sqrt(m))) (mod m)*  
  
*Step 1: Find all mods of a^r (from 0 to sqrt(m)-1)*  
  
*Step 2: For q=1, .... , n, such that q\*n < m;*  
*Find out if (b/a^(q\*sqrt(m))) (mod m) matches ANY VALUE of a^r*  
  
*The minimum q for which a match is found gives x=q\*sqrt(m)+r*  
*[r here being the value with which it matched]*  
  
*If there is no match, there is no solution :'(*  
  
*\*/*  
  
map <**long** **long**, **int**> valr;  
  
**long** **long** fast\_expo(**long** **long** a, **long** **long** power, **long** **long** mod){ *//returns (a^power) % mod*  
 **if**(power==0) **return** 1LL;  
 **if**(power%2) **return** (a\*fast\_expo(a, power-1, mod))%mod;  
 **long** **long** save=fast\_expo(a, power/2, mod)%mod;  
 **return** (save\*save)%mod;  
}  
  
**void** loop\_over(**long** **long** a, **long** **long** m){ *//Stores all values of a^r % m*  
 **long** **long** rt=sqrt(m);  
 **long** **long** cur=1LL;  
 a%=m;  
 valr[cur]=1;  
 **for**(**long** **long** i=1; i<rt; i++){  
 cur=(cur\*a)%m;  
 **if**(valr[cur]==0)valr[cur]=i+1;  
 **else** **break**;  
 }  
}  
  
**long** **long** find\_match(**long** **long** a, **long** **long** b, **long** **long** m){ *//Assumes m is a prime*  
 **long** **long** rt=sqrt(m), val, mul;  
 b%=m;  
 **for**(**long** **long** i=0; (i\*rt)<m; i++){  
 mul=fast\_expo(a, i\*rt, m);  
 val=(b\*fast\_expo(mul, m-2, m))%m;  
 **if**(valr[val]!=0){  
 **return** (i\*rt)+valr[val]-1;  
 }  
 }  
 **return** -1; *//There is no solution*  
}  
  
**long** **long** discrete\_log(**long** **long** a, **long** **long** b, **long** **long** m){  
 loop\_over(a, m);  
 **return** find\_match(a, b, m);  
}