model pollard_rho

标准

```
#include<bits/stdc++.h>
using namespace std;

namespace MR{
    const int times=50;
    // 乘法防止溢出， 如果p * p不爆LL的话可以直接乘； O(1)乘法或者转化成二进制加法
    long long qmul(long long x,long long y,long long mod){ return (x*y-(long long)(x/(long double)mod*y+1e-3)*mod+mod)%mod; }
    long long qpow(long long a,long long b,long long p){a%=p; long long ret=1;for(;b;b>>=1,a=qmul(a,a,p)) if(b&1) ret=qmul(ret,a,p); return ret; }
    bool Miller_Rabin(long long n){
        if(n<3) return n==2;
        long long u=0,v=n-1;
        while(v%2==0) u++,v>>=1;
        for(int i=0;i<times;i++){
            long long w=2+rand()%(n-2),x=qpow(w,v,n);
            if(x==1||x==n-1) continue;
            int j;
            for(j=0;j<u;j++){
                x=qmul(x,x,n);
                if(x==n-1) break;
            }
            if(j>=u) return 0;
        }
        return 1;
    }
    long long f(long long x,long long c,long long mod){ return ((__int128)x*x+c)%mod; }
    long long find_factor(long long p){
        long long x,y,z,c=0,g; int i,j;
        while(1){
            y=x=rand()%p;
            z=1; c++;
            i=0,j=1;
            while(++i){
                x=f(x,c,p);
                z=(__int128)z*abs(y-x)%p;
                if(x==y||!z) break;
                if(!(i%127)||i==j){
                    g=__gcd(z,p);
                    if(g>1) return g;
                    if(i==j) y=x,j<<=1;
                }
            }
        }
    }
    void Pollard_Rho(vector<long long> &cnt,long long n){
        while(!(n&1)) cnt.push_back(2),n>>=1;
        if(n==1) return;
        if(Miller_Rabin(n)) return cnt.push_back(n),void();
        long long p=find_factor(n);
        Pollard_Rho(cnt,n/p),Pollard_Rho(cnt,p);
    }
}
void solve(){
    srand((unsigned)time(NULL));
    long long n; cin>>n;
    vector<long long> res;

    MR::Pollard_Rho(res,n);

    map<long long,int> cnt;
    for(auto v:res) cnt[v]++;
    for(auto v:cnt)
        if(v.second>1){
            cout<<"yes\n";
            return;
        }
    puts("no");
}
int main(){
    int T; cin>>T;
    while(T--) solve();
}
```

玄学

```
#include<bits/stdc++.h>
using namespace std;

namespace MR{
    // 18位素数：154590409516822759
    // 19位素数：2305843009213693951 (梅森素数)
    // 19位素数：4384957924686954497
    long long  prime[11] = {2,3,5,7,233,331,11,13,17,19,23};
    long long  mi;
    // 乘法防止溢出， 如果p * p不爆LL的话可以直接乘； O(1)乘法或者转化成二进制加法
    long long qmul(long long x,long long y,long long mod){ return (x*y-(long long)(x/(long double)mod*y+1e-3)*mod+mod)%mod; }
    long long qpow(long long a,long long b,long long mod){ long long ret=1; for(;b;a=qmul(a,a,mod),b>>=1) if(b&1) ret=qmul(ret,a,mod); return ret; }
    bool M_R(long long p){//传入值，返回0即为合数，犯为1即为质数，范围可测到ll范围
        if(p==2) return 1;
        if(p<2||!(p&1)) return 0;
        long long s = p - 1;
        while(!(s&1)) s>>=1;
        for(int i=0;i<11;++i) {
            if(p==prime[i]) return 1;
            long long t=s,m=qpow(prime[i],s,p);
            while(t!=p-1&&m!=1&&m!=p-1){
                m=qmul(m,m,p);
                t<<=1;
            }
            if(m!=p-1&&!(t&1)) return 0;
        }
        return 1;
    }
    long long f(long long x,long long mod,int a){ return ((__int128)x*x+a)%mod; }
    long long find_factorplus(long long N,long long seed){
        long long a=rand(),b=a,p;
        do{
            a = f(a,N,seed);
            b = f(f(b,N,seed),N,seed);
            p = __gcd( abs( b - a ) , N);
            if( p > 1&&p<N) return p;
        }while(b!=a);
        return 0;
    }
    void p_r(vector<long long> &cnt,long long x){
        while((x&1)==0) cnt.push_back(2),x>>=1;
        if(x==1) return;
        if(M_R(x)) return cnt.push_back(x),void();
        long long p=0;
        while(p==0){
            long long seed=1+rand()%(x-1);
            p=find_factorplus(x,seed);
        }
        p_r(cnt,p),p_r(cnt,x/p);
    }
}
void solve(){
    srand((unsigned)time(NULL));
    long long n; cin>>n;
    vector<long long> res;

    MR::p_r(res,n);

    map<long long,int> cnt;
    for(auto v:res) cnt[v]++;
    for(auto v:cnt)
        if(v.second>1){
            puts("yes");
            return;
        }
    puts("no");
}
int main(){
    int T; cin>>T;
    while(T--) solve();
}
```