Pollard Rho Prime Factorization (Python recipe) by Mukesh Tiwari

ActiveState Code (http://code.activestate.com/recipes/577037/)

This code is implementation of Pollard Rho prime factorization. As i am a bit new in python so further improvement is appreciated.Also added Brent variant.

Python, 108 lines

```python
# To change this template, choose Tools | Templates
# and open the template in the editor.

__author__="Mukesh Tiwari"
__date__ ="$Feb 10, 2010 1:35:26 AM$"

import random
from Queue import Queue
def gcd(a,b):
    while b:
        a,b=b,a%b
    return a

def rabin_miller(p):
        if(p<2):
                return False
        if(p!=2 and p%2==0):
                return False
        s=p-1
        while(s%2==0):
                s>>=1
        for i in xrange(10):
                a=random.randrange(p-1)+1
                temp=s
                mod=pow(a,temp,p)
                while(temp!=p-1 and mod!=1 and mod!=p-1):
                        mod=(mod*mod)%p
                        temp=temp*2
                if(mod!=p-1 and temp%2==0):
                        return False
        return True
def brent(n):
    if(n%2==0):
        return 2;
    x,c,m=random.randrange(0,n),random.randrange(1,n),random.randrange(1,n)
    y,r,q=x,1,1
    g,ys=0,0
    while(True):
        x=y
        for i in range(r):
            y,k=(y*y+c)%n,0
        while(True):
            ys=y
            for i in range(min(m,r-k)):
                y,q=(y*y+c)%n,q*abs(x-y)%n
            g,k=gcd(q,n),k+m
            if(k>= r or g>1):break
        r=2*r
        if(g>1):break
    if(g==n):
        while(True):
            ys,g=(x*x+c)%n,gcd(abs(x-ys),n)
            if(g>1):break
    return g

def pollard(n):
        if(n%2==0):
            return 2;
        x=random.randrange(2,1000000)
        c=random.randrange(2,1000000)
        y=x
        d=1
        while(d==1):
            x=(x*x+c)%n
            y=(y*y+c)%n
            y=(y*y+c)%n
            d=gcd(x-y,n)
            if(d==n):
                break;
        return d;
def factor(n):
    #if(rabin_miller(n)):
     #   print n
      #  return
    #d=pollard(n)
    #if(d!=n):
     #   factor(d)
      #  factor(n/d)
    #else:
     #   factor(n)

    Q_1=Queue()
    Q_2=[]
    Q_1.put(n)
    while(not Q_1.empty()):
        l=Q_1.get()
        if(rabin_miller(l)):
            Q_2.append(l)
            continue
        d=pollard(l)
        if(d==l):Q_1.put(l)
        else:
            Q_1.put(d)
            Q_1.put(l/d)
    return Q_2



if __name__ == "__main__":
    while(True):
        n=input();
        L=factor(n)
        L.sort()
        i=0
        while(i<len(L)):
            cnt=L.count(L[i])
            print L[i],'^',cnt
            i+=cnt
```

Tags: algorithm, algorithms