***NUMWORLD MODULE***

This module is executed on python .it has several constants and functions.

Devoleped by satish

***constants***

pi=3.141592653589793238462643383279

e=2.718281828459045235460

mp=1.672621777\*(10\*\*-27)

mn=1.674927351\*(10\*\*-27)

me=9.10938291\*(10\*\*-31)

h=6.62606957\*(10\*\*-34)

H=h/(2\*pi)

R=8.3144621

F=96485.3365

Na=6.02214129\*(10\*\*23)

Bk=1.3806488\*(10\*\*-23)

Eo=8.854187817\*(10\*\*-12)

Uo=1.256637061\*(10\*\*-6)

g=9.80665

Go=7.748091735\*(10\*\*-5)

G=6.67384\*(10\*\*-11)

atm=101325

C=299792458

prime=2\*\*82589933-1

phi=1.61803398874989484820458683

rad=180/pi

deg=pi/180

***documentation on functions***

* """
* fib(n) : returns highest fibonacci number below n
* isprime(n) : checks the number prime or not
* iscomposite(n): checks the number composite or not
* isperfect(n) : checks the number perfect or not
* issquare(n) : checks the number perfect square or not
* rfact(n) : reverse factorial u gives 120 it returns 5.0
* comfact(\*n) : it returns the all common factors to a tuple of integers or list of integers
* isnsquare(n) : return true when number is not square
* nprimes(n) : it returns list of n primes from starting prime (2)
* limprimes(n1,n2) : it takes two numbers as arguements and return all primes between them
* tables(tables=1,multiple=10,step=1) : it shows a table for children with any table and multiple means upto which number we want to multiply
  + - * + step means like indexing in list
* maxpow(fact,num) : it takes factorial in form of "n!" and return how many multiples of num present in its elongated form
* primefact(n) : return all prime factors of number n
* """

***Functions:***

def fib(n):

a=0

b=1

c=0

while(c<=n):

c=a+b

a=b

b=c

if c<=n:

temp=c

return temp

def isprime(n):

c=0

for i in range(1,n+1):

if n%i==0:

c+=1

if(c==2):

return True

else:

return False

def iscomposite(n):

c=0

for i in range(1,n+1):

if n%i==0:

c+=1

if(c==2):

return False

else:

return True

def issquare(n):

c=n\*\*0.5

if c==int(c):

return True

else:

return False

def isnsquare(n):

c=n\*\*0.5

if c==int(c):

return False

else:

return True

def isperfect(n):

p=1

for i in range(1,n):

if n%i==0:

p=p\*i

if n==p:

return True

else:

return False

def rfact(n):

if n==1 or n==0:

return 1

else:

i=1

while(1):

if n%i==0:

n=n/i

i=i+1

if n==i:

return n

else:

pass

else:

return None

def comfact(\*n):

l=[]

for i in range(1,(min(n)+1)):

c=0

for j in range(len(n)):

if n[j]%i==0:

c=c+1

if c==len(n):

l.append(i)

else:

pass

return l

def nprimes(n):

l=[]

x=2

while(len(l)<n):

c=0

for i in range(1,x+1):

if x%i==0:

c=c+1

if c==2:

l.append(i)

x=x+1

return l

def limprimes(n1,n2):

l=[]

for i in range(n1,n2+1):

c=0

for j in range(1,i+1):

if i%j==0:

c=c+1

if(c==2):

l.append(i)

else:

pass

return l

def tables(t=1,n=10,step=1):

i=0

while(i<=n):

print(" | "+str(t)+" X "+str(i)+" = "+str(t\*i)+" |")

i=i+step

def maxpow(fact,num):

s=0

while(num<=fact):

s=s+(fact//num)

num=num\*num

return s

def primefact(n):

l=[]

for i in range(2,n+1):

if n%i==0:

c=0

for j in range(1,i+1):

if i%j==0:

c=c+1

if c==2:

l.append(i)

else:

pass

return l