```
"""
Registration: xxxx;
Description: Code for Integration by Simpson's 1/3 Rule
Author: AKB
"""

import numpy as np
from scipy.integrate import simps

def f(x):
    return np.exp(-x**2)*np.sin(x)     # 0,1
    #return np.sqrt(x)*np.exp(x)       # 0,1
    #return np.sqrt(1-x**2)*np.cos(x)  # 0,0.25

# Main

# Enter the integration limits
a,b = input('Enter the lower and upper multiplier of pi: ')
a *= np.pi
b *= np.pi
tol = 1E-10  # Tolerance

# Compute I0
h  = (b-a)*0.5
s1 = f(a) + f(b)
s4 = f(a+h)
I0 = 1E-16
I1 = (s1+4.0*s4)*h/3.0

# Do the integral
i=2; s2=0.0;
while(abs((I1-I0)/I1)>tol):
    x   = a+h*0.5
    s2 += s4
    s4  = 0.0
    for j in range(i):
        s4 += f(x)
        x += h
        #print 'x=',x,'i=',i,'h=',h,'s2=',s2,'s4='s4 #check what's happening
    h *= 0.5
    i *= 2
    I0 = I1
    I1 = (s1+2.0*s2+4.0*s4)*h/3.0

# Print the solution
print 'Number of iterations : ', i/2
print 'Integral_',a,'^',b,' sqrt(1-x^2)*cos(x) dx = ', I1

# Using Numpy/Scipy
x = np.linspace(a, b, 128)
print 'Integral_',a,'^',b,' sqrt(1-x^2)*cos(x) dx = ', simps(f(x),x)

"""
Results:
integral_0^pi      e^(-x^2)sin(x) dx =  0.424437737132, # of iterations = 32
integral_0^pi       sqrt(x)exp(x) dx = 32.8303077022,    # of iterations = 32
integral_0^pi/4 sqrt(1-x^2)cos(x) dx =  0.633009142817, # of iterations = 4
"""
```