Numerical Integration Solutions

1.

```
import scipy
from scipy.integrate import quad
from math import *

#part A
def f(x):
    return x * log(x)
print quad(f, 1, 2)[0]

#part B
def g(x):
    return 1.0 / (x ** 6 + 1)
print quad(g, -pi, pi)[0]

#part C
def h(x):
    return x ** x
print quad(h, 1, 2)[0]

#part D
def k(x):                   #we will 'fill' the discontinuity of the function at 0 with its limit at 0
    if x == 0: return 1
    else: return sin(x) / x

def trap(k, n, a, b):     #define a function to do integration of k(x) btw. a and b
    h = (b - a) / float(n)
    intgr = 0.5 * h * (k(a) + k(b))
    for i in range(1, int(n)):
        intgr = intgr + h * k(a + i * h)
    return intgr

a = 0
b = 1e6
n = 1e6

while(abs(trap(k, n, a, b) - trap(k, n * 4, a, b * 2)) > 1e-6):
    n *= 2
    b *= 2
print trap(k, n, a, b)
```

2.

```
from scipy import *
from scipy.integrate import odeint
from pylab import *

def equations_a(u,t):
    x, x_prime = u
    return (x_prime, x_prime - sin(t)) #the vector (x', x'')

def equations_b(u,t):
    x, x_prime = u
    return (x_prime, x_prime - sin(x)) #the vector (x', x'')

def equations_c(u,t):
    x, x_prime, x_prime_prime = u
    return (x_prime, x_prime_prime, t * x - x_prime_prime) #the vector (x', x'', x''')

t = arange(-1, 1.001,0.001)
u0 = array([1, -1])   #initial values of x, x' for parts A and B
v0 = array([1, -1, 2]) #initial values of x, x', x'' for part C

part_a = odeint(equations_a, u0, t)
part_b = odeint(equations_b, u0, t)
part_c = odeint(equations_c, v0, t)
subplot(3, 1, 1)
plot(t, part_a[:, 0], 'r-')
title('Part A')
subplot(3, 1, 2)
plot(t, part_b[:, 0], 'r-')
title('Part B')
subplot(3, 1, 3)
plot(t, part_c[:, 0], 'r-')
title('Part C')
show()
```

3.

```
from scipy import *
from scipy.integrate import odeint
from pylab import *

def equations(u,t):
    x, y = u
    return (sqrt(y), x * sqrt(y)) #the vector (x', y')

t = arange(-1, 1.001, 0.001)
u0 = array([0, 1])   #initial values of x and y

u = odeint(equations, u0, t)
figure(1)
plot(t, u[:, 0], 'r-')
title('X')
figure(2)
plot(t, u[:, 1], 'r-')
title('Y')
show()
```