```
 1: from pylab import *
 2: from numpy import *
 3: from scipy import integrate
 4:
 5:
 6: def trapazoidal(func, a, b, h=0.001):
 7:     return h * ((0.5 * (func(a) + func(b))) + sum(
 8:         [func(a + k * h) for k in range(1, int((b - a) / h))]))
 9:
10:
11: def simpson(func, a, b, h=0.001):
12:     n = int(abs(b - a) / h)
13:     n -= 1 if n % 2 == 1 else 0
14:     return (h / 3.0) * (
15:         func(a) + func(b) +
16:         (4.0 * sum([func(a + (k * h)) for k in range(1, n, 2)])) +
17:         (2.0 * sum([func(a + (k * h)) for k in range(2, n - 1, 2)])))
18:
19:
20: def adaptive(input_func, a, b, h=0.001, delta=10e-4):
21:     if a == inf and b == inf:
22:         func = lambda x: (input_func(-x/(1-x))+input_func(x/(1-x))) / pow(1-x,2)
23:         a = 0
24:         b = 1
25:     elif a != inf and b == inf:
26:         func = lambda x: input_func((x / (1 - x)) + a) / pow(1 - x, 2)
27:         a = 0
28:         b = 1
29:     else:
30:         func = input_func
31:     try:
32:         func(a)
33:     except ZeroDivisionError:
34:         a += 0.00000000000001
35:     try:
36:         func(b)
37:     except ZeroDivisionError:
38:         b -= 0.00000000000001
39:     n = int(abs(b - a) / h)
40:     i0 = h * ((0.5 * (func(a) + func(b))) + sum(
41:         [func(a + k * h) for k in range(1, int((b - a) / h))]))
42:     epsilon = delta + 10
43:     while epsilon > delta:
44:         h /= 2
45:         n *= 2
46:         i1 = (0.5 * i0) + (h * sum([func(a + k * h) for k in range(1, n, 2)]))
47:         epsilon = abs(i1 - i0) / 3
48:         i0 = i1
49:     return i1, epsilon
50:
51: real = 0.13525725794999465456801359978220103186955208405595013898 2
52: a = adaptive(lambda x: exp(-(x**2)), 1, 2, 0.001)
53: print(a)
54: # b = adaptive(lambda x: exp(-((x / (1 - x))**2)) / pow(1 - x, 2), 0,
55: #               1 - 0.0000001, 0.001, 1e-12)
56: # c = adaptive_a_to_inf(lambda x: exp(-x**2), 0, 0.001, 1e-12)
57: d = adaptive(lambda x: exp(-x**2), 0, inf, 0.001, 1e-12)
58: print(d)
```