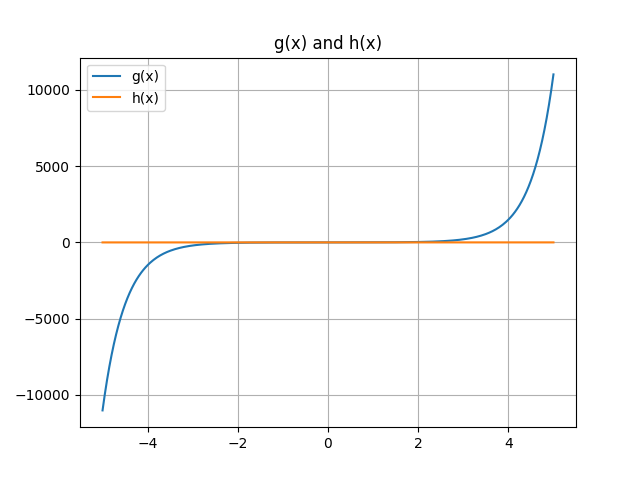
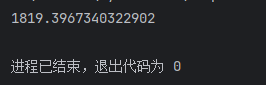
## 1

import numpy as np   
import matplotlib.pyplot as plt   
   
def func\_g(x):   
 return 0.5 \* (np.exp(2 \* x) - np.exp(-2 \* x))   
   
def func\_h(x):   
 ex2 = np.exp(2 \* x)   
 e\_minus\_2x = np.exp(-2 \* x)   
 return (ex2 - e\_minus\_2x) / (ex2 + e\_minus\_2x)   
   
x\_vals = np.linspace(-5, 5, 400)   
plt.plot(x\_vals, func\_g(x\_vals), label='g(x)')   
plt.plot(x\_vals, func\_h(x\_vals), label='h(x)')   
plt.legend()   
plt.title('g(x) and h(x)')   
plt.grid()   
plt.show()



## 2

def future\_value(pv, rate, periods):   
 if pv < 0 or rate < 0 or periods < 0:   
 raise ValueError("Inputs must be non-negative")   
 return pv \* (1 + rate) \*\* periods   
   
fv = future\_value(1000, 0.005, 120)   
print(fv)



## 3

import numpy as np   
import matplotlib.pyplot as plt   
   
def free\_fall(y0, t):   
 if np.any(t < 0):   
 raise ValueError("Time must be non-negative")   
 g = -9.81   
 v = g \* t   
 y = y0 + 0.5 \* g \* t\*\*2   
 return v, y   
   
t = np.arange(0, 10.1, 0.1)   
v, y = free\_fall(20, t)   
   
plt.plot(t, v, label='Velocity')   
plt.plot(t, y, label='Distance')   
plt.legend()   
plt.title('Free Fall')   
plt.xlabel('Time (s)')   
plt.grid()   
plt.show()

