```python
import numpy as np
import matplotlib.pyplot as plt
import statistics as stats

# target function
x = np.linspace(-1, 1, 10**5+1)
y = x**2
plt.figure()
plt.plot(x, y)
a = []
b = []
e = []

g_square = []
a = []
b = []
x_test = []
y_test = []
x_square_mean = stats.mean(x**3)
x_cubic_mean = stats.mean(x**2)
x_linear_mean = stats.mean(x)
x_quad_mean = stats.mean(x**4)
Ex_mean = []
for i in range(10000):
    x1 = np.random.randint(0, 10**5+1)
    x2 = np.random.randint(0, 10**5+1)
    x3 = np.random.randint(0, 10**5+1)
    y1 = x[x1] ** 2
    y2 = x[x2] ** 2
    y3 = x[x3] ** 2
    # g = ax + b
    ai = x[x1] + x[x2]
    bi = -x[x1] * x[x2]
    #Ex = x_quad_mean - 2 * ai * x_cubic_mean + (ai ** 2 - 2 * bi) * x_cubic_mean + bi ** 2
    Ex = (ai**2 - 2 * bi) / 3 + bi**2 + 1/5
    Ex_mean.append(Ex)
```

```
    a.append(ai)
    b.append(bi)
a_mean = stats.mean(a)
b_mean = stats.mean(b)
print('mean of g(x): {} * x + {}'.format(a_mean, b_mean))
y_pred = a_mean * x + b_mean
plt.plot(x, y_pred)
plt.show()

# Ed_Eout
Ex = x_quad_mean - 2 * np.transpose(a) * x_cubic_mean + (np.transpose(a)**2 - 2 * np.transpose(b)) *
x_cubic_mean + np.transpose(b)**2
print('Ed(Eout): {}'.format(stats.mean(Ex)))

# bias
Ex_bias = (a_mean * x + b_mean + x**2)**2
bias = stats.mean(Ex_bias)
print('bias: {}'.format(bias))

# variance
#for i in range(len(a)):
Ex_variance = (np.transpose(a) - a_mean)**2 * x_cubic_mean + 2 * (np.transpose(a) - a_mean) * (np.transpose(b) -
a_mean) * x_linear_mean + (np.transpose(b) - b_mean)**2
variance = stats.mean(Ex_variance)
print('variance: {}'.format(variance))
```