```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Mar 15 16:11:40 2022

@author: siobhanie
"""

import numpy as np


#Del 1

def loss_gradient(x, f_of_x, theta):
    res = theta[0] + theta[1] * x - f_of_x
    return 2*(res.mean()), 2*((res * x).mean())

def gradient_descent(
    gradient, x, f_of_x, start, gamma, m, tol
):

    vector_nu = start
    for i in range(m): # ta m steg
        diff = -gamma * np.array(gradient(x, f_of_x, vector_nu)) # beräkna steget
        if np.all(np.abs(diff) <= tol): # sluta om steget är tillräckligt litet
            break
        vector_nu += diff # uppdatera
        print(vector_nu)
    return vector_nu


x = np.array([1, 2, 3, 4, 5, 6])
f = lambda x: 2 + 3*x
f_of_x = f(x)

coeffs = (gradient_descent(
  loss_gradient, x, f_of_x, start=[0.5, 0.5], gamma=0.001,
  m=30000, tol=1.e-6
))

#print(coeffs)

approx_func = lambda x: coeffs[0] + coeffs[1]*x

val = (approx_func(x) - f(x))**2
tran_fel = val.mean()
print(tran_fel)
```

```
x2 = np.random.rand(10,1)

val = (approx_func(x2) - f(x2))**2
gen_fel = val.mean()
print(gen_fel)




#Del 2

def loss_gradient(x, f_of_x, theta):
    res = theta[0] + theta[1] * x - f_of_x
    return 2*(res.mean()), 2*((res * x).mean())



def sgd(
    gradient, x, f_of_x, start, gamma, batch_size, m,
    tol
):

    N_obs = x.shape[0]
    x_f_of_x = np.c_[x.reshape(N_obs, -1), f_of_x.reshape(N_obs, 1)]
    print(x_f_of_x)

    # Initialisering
    seed = None
    rng = np.random.default_rng(seed=seed)
    theta_nu = start

    for i in range(m):

        rng.shuffle(x_f_of_x) # blanda

        first=0
        last = first + batch_size
        x_batch, y_batch = x_f_of_x[first:last, :-1], x_f_of_x[first:last, -1:]
        # punkter vi använder i beräkningen av gradienten

        # beräkna gradienten och steget
        grad = np.array(gradient(x_batch, y_batch, theta_nu))
        diff = -gamma * grad

        # sluta om steget är tillräckligt litet
        if np.all(np.abs(diff) <= tol):
            break
        # uppdatera approximationen till theta
        theta_nu += diff
        #print(theta_nu)

    return theta_nu

x = np.array([1, 2, 3, 4, 5, 6])
f = lambda x: 2 + 3*x
f_of_x = f(x)

coeffs2 = sgd(
loss_gradient, x, f_of_x, start=np.array([0.5, 0.5]), gamma=0.001,
batch_size=3, m=10000, tol=1.e-6
)

#print(coeffs2)




approx_func2 = lambda x: coeffs2[0] + coeffs2[1]*x
x2 = np.random.rand(10,1)

val = (approx_func2(x) - f(x))**2
tran_fel = val.mean()
print(tran_fel)


val = (approx_func2(x2) - f(x2))**2
gen_fel = val.mean()
print(gen_fel)
```