Ввод [1]:
```
import numpy as np
```

Задача 1

Ввод [ ]:
```
Даны значения величины заработной платы заемщиков банка (zp) и значения их поведенческого кредитного с
zp = [35, 45, 190, 200, 40, 70, 54, 150, 120, 110],
ks = [401, 574, 874, 919, 459, 739, 653, 902, 746, 832].
Используя математические операции, посчитать коэффициенты линейной регрессии,
приняв за X заработную плату (то есть, zp - признак),
а за y - значения скорингового балла (то есть, ks - целевая переменная).
Произвести расчет как с использованием intercept, так и без.
```

Ввод [2]:
```
zp = [35, 45, 190, 200, 40, 70, 54, 150, 120, 110]
ks = [401, 574, 874, 919, 459, 739, 653, 902, 746, 832]
n = len(zp)
```

Ввод [3]:
```
# Расчет с использованием intercept
Y = np.array(ks).reshape(-1, 1)
X = np.array(zp).reshape(-1, 1)
X = np.hstack([np.ones((n,1)), X])
#
B = np.dot(np.linalg.inv(np.dot(X.T, X)),np.dot(X.T, Y))
print(B)
```

```
[[444.17735732]
 [  2.62053888]]
```

Ввод [4]:
```
# Расчет без использования intercept
Y = np.array(ks).reshape(-1, 1)
X = np.array(zp).reshape(-1, 1)
#
B = np.dot(np.linalg.inv(np.dot(X.T, X)),np.dot(X.T, Y))
print(B)
```

```
[[5.88982042]]
```

$$b = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}$$

Ввод [5]:
```
b = sum([x*y for x,y in zip(zp,ks)])/sum([x**2 for x in zp])
print(b)
```

```
5.889820420132689
```

Задача 2

Ввод [ ]:
```
Посчитать коэффициент линейной регрессии при заработной плате (zp), используя градиентный спуск (без i
```

Ввод [6]:
```
X = np.array(zp)
Y = np.array(ks)
```

Ввод [7]:
```
def mse_func(Y, X, b):
    n = len(X)
    return np.sum((b*X - Y) ** 2) / n

def print_func(m, b, mse):
    print(f'Количество итераций: {m}\nb = {b}')
    print(f'mse = {mse}')
    print('-'*25)
```

Ввод [8]:
```
alpha = 0.00001
b = 5
m = 0

while m < 100:
    m += 1
    delta_b = alpha * (2 / n) * np.sum((b*X - Y) * X)
    b -= delta_b

print_func(m, b, mse_func(Y, X, b))

while m < 200:
    m += 1
    delta_b = alpha * (2 / n) * np.sum((b*X - Y) * X)
```

```
    b -= delta_b

print_func(m, b, mse_func(Y, X, b))
```

```
Количество итераций: 100
b = 5.889820420132679
mse = 56516.85841571941
-------------------------
Количество итераций: 200
b = 5.889820420132688
mse = 56516.8584157194
-------------------------
```

Задача 3

Ввод [ ]:
```
Произвести вычисления как в пункте 2, но с вычислением intercept.
Учесть, что изменение коэффициентов должно производитьсяна каждом шаге одновременно
(то есть изменение одного коэффициента не должно влиять на изменение другого во время одной итерации).
```

Ввод [9]:
```
X = np.array(zp)
Y = np.array(ks)
```

Ввод [10]:
```
def mse_func(Y, X, b1, b0):
    n = len(X)
    return np.sum((b1*X + b0 - Y) ** 2) / n

def print_func(m, b0, b1, mse):
    print(f'Количество итераций: {m}\nb0 = {b0}\nb1 = {b1}')
    print(f'mse = {mse}')
    print('-'*25)
```

Ввод [11]:
```
alpha_b1 = 0.00001
alpha_b0 = 0.4
b1 = 5
b0 = 400
m = 0
while m < 500:
    m += 1
    delta_b1 = alpha_b1 * (2 / n) * np.sum((b1*X + b0 + - Y) * X)
```

```
    delta_b0 = alpha_b0 * (2 / n) * np.sum((b1*X + b0 + - Y))
    b1 -= delta_b1
    b0 -= delta_b0

print_func(m, b0, b1, mse_func(Y, X, b1, b0))


while m < 1000:
    m += 1
    delta_b1 = alpha_b1 * (2 / n) * np.sum((b1*X + b0 + - Y) * X)
    delta_b0 = alpha_b0 * (2 / n) * np.sum((b1*X + b0 + - Y))
    b1 -= delta_b1
    b0 -= delta_b0

print_func(m, b0, b1, mse_func(Y, X, b1, b0))
```

```
Количество итераций: 500
b0 = 444.1773573242368
b1 = 2.6205388824038933
mse = 6470.414201176655
-------------------------
Количество итераций: 1000
b0 = 444.17735732435926
b1 = 2.6205388824027684
mse = 6470.414201176657
-------------------------
```

Ввод [ ]: