In [1]:

```
import numpy as np

X = np.array([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1],   # для умножения на interce
pt
              [1, 1, 2, 1, 3, 0, 5, 10, 1, 2]]) # стаж

# средний балл ЕГЭ (целевая переменная)
y = [45, 55, 50, 59, 65, 35, 75, 80, 50, 60]
```

In [2]:

```
def calc_mse(y, y_pred):
    err = np.mean((y - y_pred)**2)
    return err

def calc_mae(y, y_pred):
    err = np.mean(np.abs(y - y_pred))
    return err
```

1. Подберите скорость обучения (alpha) и количество итераций:

In [3]:

```
def func(n, w, alpha, n_iter):
    for i in range(n_iter):
        y_pred = np.dot(w, X)
        err = calc_mse(y, y_pred)
        for j in range(w.shape[0]):
            w[j] -= alpha * (1/n * 2 * np.sum(X[j] * (y_pred - y)))
        if i % 100 == 0:
            print(i, w, err)

n = X.shape[1]
w = np.array([1, 0.5])

i = 1
for alpha in [1e-1, 1e-2, 1e-3]:
    for n_iter in [500, 800, 1000, 1200]:
        print(f'\nПопытка #{i}: alpha={alpha}, n_iter={n_iter}')
        func(X.shape[1], np.array([1, 0.5]), alpha, n_iter)
        i += 1
```

```
Попытка #1: alpha=0.1, n_iter=500
0 [12.02 34.5 ] 3173.15
100 [1.21092169e+31 6.55765787e+31] 1.6499598531554106e+64
200 [3.40912000e+61 1.84618401e+62] 1.3077533388173627e+125
300 [9.59772979e+91 5.19758039e+92] 1.0365214595478304e+186
400 [2.70205851e+122 1.46328003e+123] 8.215438678021257e+246

Попытка #2: alpha=0.1, n_iter=800
0 [12.02 34.5 ] 3173.15
100 [1.21092169e+31 6.55765787e+31] 1.6499598531554106e+64
200 [3.40912000e+61 1.84618401e+62] 1.3077533388173627e+125
300 [9.59772979e+91 5.19758039e+92] 1.0365214595478304e+186
```

```
400 [2.70205851e+122 1.46328003e+123] 8.215438678021257e+246
500 [7.60713247e+152 4.11958697e+153] inf
600 [2.14164365e+183 1.15979146e+184] inf
700 [6.02939088e+213 3.26517254e+214] inf

Попытка #3: alpha=0.1, n_iter=1000
0 [12.02 34.5 ] 3173.15
100 [1.21092169e+31 6.55765787e+31] 1.6499598531554106e+64
200 [3.40912000e+61 1.84618401e+62] 1.3077533388173627e+125
300 [9.59772979e+91 5.19758039e+92] 1.0365214595478304e+186
400 [2.70205851e+122 1.46328003e+123] 8.215438678021257e+246
500 [7.60713247e+152 4.11958697e+153] inf
600 [2.14164365e+183 1.15979146e+184] inf
700ipykernel_launcher:2: RuntimeWarning: overflow encountered
in square
C:\Users\konst\AppData\Local\Programs\Python\Python37\lib\sit
e-packages\numpy\core\_methods.py:151: RuntimeWarning: overfl
ow encountered in reduce
  ret = umr_sum(arr, axis, dtype, out, keepdims)
C:\Users\konst\AppData\Local\Programs\Python\Python37\lib\sit
e-packages\numpy\core\fromnumeric.py:90: RuntimeWarning: over
flow encountered in reduce
  return ufunc.reduce(obj, axis, dtype, out, **passkwargs)
 [6.02939088e+213 3.26517254e+214] inf
800 [1.69746047e+244 9.19247307e+244] inf
900 [4.77887752e+274 2.58796618e+275] inf

Попытка #4: alpha=0.1, n_iter=1200
0 [12.02 34.5 ] 3173.15
100 [1.21092169e+31 6.55765787e+31] 1.6499598531554106e+64
200 [3.40912000e+61 1.84618401e+62] 1.3077533388173627e+125
300 [9.59772979e+91 5.19758039e+92] 1.0365214595478304e+186
400 [2.70205851e+122 1.46328003e+123] 8.215438678021257e+246
500 [7.60713247e+152 4.11958697e+153] inf
600 [2.14164365e+183 1.15979146e+184] inf
700 [6.02939088e+213 3.26517254e+214] inf
800 [1.69746047e+244 9.19247307e+244] inf
900 [4.77887752e+274 2.58796618e+275] inf
1000 [1.34540219e+305 7.28592718e+305] inf
1100 [nan nan] nan

Попытка #5: alpha=0.01, n_iter=500
0 [2.102 3.9  ] 3173.15
100 [31.88770806  6.74418155] 175.19445858001853
200 [41.83683774  4.90699865] 61.9177717428135
300 [45.33508261  4.26102097] 47.913169919666785
400 [46.56511152  4.03388672] 46.181755648107604

Попытка #6: alpha=0.01, n_iter=800
0 [2.102 3.9  ] 3173.15
100 [31.88770806  6.74418155] 175.19445858001853
200 [41.83683774  4.90699865] 61.9177717428135
300 [45.33508261  4.26102097] 47.913169919666785
400 [46.56511152  4.03388672] 46.181755648107604
500 [46.99760587  3.95402334] 45.96769776787538
600 [47.14967657  3.92594232] 45.941233404700036
700 [47.20314662  3.91606866] 45.93796156758051

Попытка #7: alpha=0.01, n_iter=1000
0 [2.102 3.9  ] 3173.15
```

```
0 [2.102 3.9  ] 3173.15
100 [31.88770806  6.74418155] 175.19445858001853
200 [41.83683774  4.90699865] 61.9177717428135
300 [45.33508261  4.26102097] 47.913169919666785
400 [46.56511152  4.03388672] 46.181755648107604
500 [46.99760587  3.95402334] 45.96769776787538
600 [47.14967657  3.92594232] 45.941233404700036
700 [47.20314662  3.91606866] 45.93796156758051
800 [47.2219474   3.91259695] 45.93755706443538
900 [47.228558    3.91137626] 45.937507054979434

Попытка #8: alpha=0.01, n_iter=1200
0 [2.102 3.9  ] 3173.15
100 [31.88770806  6.74418155] 175.19445858001853
200 [41.83683774  4.90699865] 61.9177717428135
300 [45.33508261  4.26102097] 47.913169919666785
400 [46.56511152  4.03388672] 46.181755648107604
500 [46.99760587  3.95402334] 45.96769776787538
600 [47.14967657  3.92594232] 45.941233404700036
700 [47.20314662  3.91606866] 45.93796156758051
800 [47.2219474   3.91259695] 45.93755706443538
900 [47.228558    3.91137626] 45.937507054979434
1000 [47.23088237  3.91094704] 45.937500872219864
1100 [47.23169965  3.91079613] 45.93750010783411

Попытка #9: alpha=0.001, n_iter=500
0 [1.1102 0.84  ] 3173.15
100 [ 7.43515582 10.71754582] 899.5990444359252
200 [11.44993589 10.49282195] 735.5540619653672
300 [14.98921624  9.8634316 ] 606.0048772266812
400 [18.17511588  9.27626113] 500.7989288428639

Попытка #10: alpha=0.001, n_iter=800
0 [1.1102 0.84  ] 3173.15
100 [ 7.43515582 10.71754582] 899.5990444359252
200 [11.44993589 10.49282195] 735.5540619653672
300 [14.98921624  9.8634316 ] 606.0048772266812
400 [18.17511588  9.27626113] 500.7989288428639
500 [21.04606457  8.74617136] 415.355423398327
600 [23.63334499  8.26841272] 345.9620646005472
700 [25.96499229  7.83785632] 289.6039104863001

Попытка #11: alpha=0.001, n_iter=1000
0 [1.1102 0.84  ] 3173.15
100 [ 7.43515582 10.71754582] 899.5990444359252
200 [11.44993589 10.49282195] 735.5540619653672
300 [14.98921624  9.8634316 ] 606.0048772266812
400 [18.17511588  9.27626113] 500.7989288428639
500 [21.04606457  8.74617136] 415.355423398327
600 [23.63334499  8.26841272] 345.9620646005472
700 [25.96499229  7.83785632] 289.6039104863001
800 [28.06626432  7.44984037] 243.83236130353217
900 [29.95992301  7.10016181] 206.65877484533297

Попытка #12: alpha=0.001, n_iter=1200
0 [1.1102 0.84  ] 3173.15
100 [ 7.43515582 10.71754582] 899.5990444359252
200 [11.44993589 10.49282195] 735.5540619653672
300 [14.98921624  9.8634316 ] 606.0048772266812
400 [18.17511588  9.27626113] 500.7989288428639
500 [21.04606457  8.74617136] 415.355423398327
```

```
500 [21.04606457  8.74617156] 415.555425596527
600 [23.63334499  8.26841272] 345.9620646005472
700 [25.96499229  7.83785632] 289.6039104863001
800 [28.06626432  7.44984037] 243.83236130353217
900 [29.95992301  7.10016181] 206.65877484533297
1000 [31.66648131  6.78503276] 176.46806566379882
1100 [33.20442544  6.50103962] 151.94853425111427
```

Оптимальна попытка #3: alpha=0.01, n_iter=1000

In [4]:

```
n = X.shape[1]
alpha = 1e-2
w = np.array([1, 0.5])

for i in range(1000):
    y_pred = np.dot(w, X)
    err = calc_mse(y, y_pred)
    '''for j in range(W.shape[0]):
        W[j] -= alpha * (1/n * 2 * np.sum(X[j] * (y_pred - y)))'''
    # w -= (alpha * (1/n * 2 * np.sum(X * (y_pred - y))))
    w -= (alpha * (1/n * 2 * np.dot(X, (np.dot(w, X) - y))))
    if i % 100 == 0:
        print(i, w, err)
```

```
0 [2.102 3.9  ] 3173.15
100 [31.88770806  6.74418155] 175.19445858001842
200 [41.83683774  4.90699865] 61.9177717428135
300 [45.33508261  4.26102097] 47.913169919666785
400 [46.56511152  4.03388672] 46.181755648107604
500 [46.99760587  3.95402334] 45.96769776787538
600 [47.14967657  3.92594232] 45.941233404700036
700 [47.20314662  3.91606866] 45.93796156758051
800 [47.2219474   3.91259695] 45.93755706443538
900 [47.228558    3.91137626] 45.937507054979434
```

In [5]:

```
n = X.shape[1]
alpha = 1e-2
w = np.array([1, 0.5])
threshold = 1e-10

i = 0
err_prev = 0
while True:
    y_pred = np.dot(w, X)
    err = calc_mse(y, y_pred)
    w -= (alpha * (1/n * 2 * np.dot(X, (np.dot(w, X) - y))))
    if i % 100 == 0:
        print(i, w, err)

    if abs(err_prev - err) < threshold or i > 10000:
        break
    i += 1
    err_prev = err
```

```
0 [2.102 3.9  ] 3173.15
100 [31.88770806  6.74418155] 175.19445858001842
200 [41.83683774  4.90699865] 61.9177717428135
300 [45.33508261  4.26102097] 47.913169919666785
400 [46.56511152  4.03388672] 46.181755648107604
500 [46.99760587  3.95402334] 45.96769776787538
600 [47.14967657  3.92594232] 45.941233404700036
700 [47.20314662  3.91606866] 45.93796156758051
800 [47.2219474   3.91259695] 45.93755706443538
900 [47.228558    3.91137626] 45.937507054979434
1000 [47.23088237  3.91094704] 45.937500872219864
1100 [47.23169965  3.91079613] 45.93750010783411
1200 [47.23198702  3.91074306] 45.93750001333172
```