1) Gradient Descent is an optimisation algorithm for finding a local minimum of a differentiable function. Gradient descent is used in Neural Networks to find the values of the weights that minimise a cost function.

We are going to find the minimum point of the objective function $f(x) = (x+5)^2$ using gradient descent. Derivative of $f(x)$ is $f'(x) = 2(x+5)$. Watch this 5 min video to see what we are doing: https://www.youtube.com/watch?v=Gbz8RljxIHo

Step 1 - initialise the following variables with values.

Variables	Explanations
x_start	This is the starting point for the function. Could be a random value.
learning_rate	Step size. Should be between 0 and 1.
n_iter	Maximum number of iterations to run.
tol	Tolerance. Stop algorithm if values between two successive iterations are smaller than this value. Usually a small number like 0.000001.

Step 2 - A python function called gradient_descent is in Week8.py. It takes in the above parameters, the function and the derivative of $f(x)$. It returns the minimum point. Try it!

2) Use the gradient descent function to find the minimum point of $f(x) = x^4 - 5x^2 - 3x$. Its derivative is $f'(x) = 4x^3 - 10x - 3$.

a) Initialise using x_start = 2.4, learning_rate = 0.005.
b) Initialise using x_start = 0, learning_rate = 0.2
c) Initialise using x_start = 0, learning_rate = 0.1
d) Initialise using x_start = -2, learning_rate = 0.1
e) Initialise using x_start = -2, learning_rate = 0.15

Notice that (d) and (e) gets trapped in a local minimum. To avoid this, add a momentum term to the gradient descent function with the following steps.

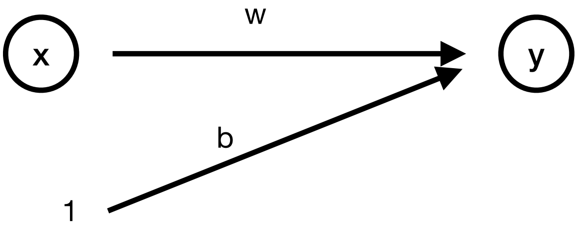
- Add an input variable in the function called decay_rate (a value between 0 and 1)
- Set diff = 0 before the while loop
- Update the line for diff to diff = decay_rate x diff - learning_rate x f'(min_point)

- Similar code is found here: https://realpython.com/gradient-descent-algorithm-python/#momentum-in-stochastic-gradient-descent

Try again using x_start = -2, learning_rate = 0.1, decay_rate = 0.8

- What is momentum? https://datascience.stackexchange.com/questions/84167/what-is-momentum-in-neural-network

3) A simplest Neural Network is linear regression with 1 input: x, 2 parameters: (w, b) and 1 output: y. We have y = wx+b, where w can be seen as a 1x1 matrix and b is the bias.



Given data for x and y, we want to find values for w and b which minimises the Mean squared error function, $C = \sum_{i}^{n} \frac{1}{2n}(y_i - b - wx_i)^2$. Here there are 2 parameters so we have to take derivative with respect to w and b separately.

$$\frac{dC}{dw} = \sum_{i}^{n} \frac{1}{n}(b + wx_i - y_i)x_i$$

$$\frac{dC}{db} = \sum_{i}^{n} \frac{1}{n}(b + wx_i - y_i)$$

Implement the #Todo for gradient_descent_reg() function in the file Week8.py and find values for the w and b.

Neural networks are simply doing this with multiple layers, higher dimensions for x, output classes for y, activation functions and a different cost function. With multiple layers, chain rule is used to compute the derivative for gradient descent.