**Introduction:**

This exercise is an implementation of learning a perceptron to perform a binary NAND function on inputs x1 and x2.

**How it works:**

The algorithm has weights vector, which is initiated in some arbitrary small number.

Then the algorithm is inputted with a training set.

The training set is basically a set of example which consists of input and the desired result.

On each iteration on an example the weights vector is updated with some new and more accurate values.

In that manner, the algorithm learns to perform the operation it should perform, by getting its weights vector to be more updated

The algorithm is done learning when the weights vector did not change since the two last iterations; and by now, it should know how to predict the desired operation output on new inputs.

**How I chose the value of the algorithm properties:**

threshold = 0.5:

learning\_rate = 0.1;

*I chose 0.1 because we want accurate predictions without having long calculation time.*

w = np.array([0, 0, 0]);

*I chose [0,0,0] because this is the smallest values.*

training\_set = np.array([(np.array([1, 0, 0]), 1), (np.array([1, 0, 1]), 1), (np.array([1, 1, 0]), 1), (np.array([1, 1, 1]), 0)]);

*I chose this training set because it covers all the input possibilities of x1 and x2: 00,01,10,11*