Week 2 Quiz - Neural Network Basics

1. What does a neuron compute?
    - [ ] A neuron computes an activation function followed by a linear function (z = Wx + b)
    - [x] A neuron computes a linear function (z = Wx + b) followed by an activation function
    - [ ] A neuron computes a function g that scales the input x linearly (Wx + b)
    - [ ] A neuron computes the mean of all features before applying the output to an activation function

    Note: we generally say that the output of a neuron is a = g(Wx + b) where g is the activation function (sigmoid, tanh, ReLU, ...).

2. Which of these is the "Logistic Loss"?
    - Check here.

    Note: this is the logistic loss you've seen in lecture!

3. Suppose img is a (32,32,3) array, representing a 32x32 image with 3 color channels red, green and blue. How do you reshape this into a column vector?
    - `x = img.reshape((32 * 32 * 3, 1))`

4. Consider the two following random arrays "a" and "b":

    ```
    a = np.random.randn(2, 3) # a.shape = (2, 3)
    b = np.random.randn(2, 1) # b.shape = (2, 1)
    c = a + b
    ```

    What will be the shape of "c"?

    b (column vector) is copied 3 times so that it can be summed to each column of a. Therefore, `c.shape = (2, 3)`.

5. Consider the two following random arrays "a" and "b":

    ```
    a = np.random.randn(4, 3) # a.shape = (4, 3)
    b = np.random.randn(3, 2) # b.shape = (3, 2)
    c = a * b
    ```

    What will be the shape of "c"?

    "*" operator indicates element-wise multiplication. Element-wise multiplication requires same dimension between two matrices. It's going to be an error.

6. Suppose you have n_x input features per example. Recall that X=[x^(1), x^(2)...x^(m)]. What is the dimension of X?

    `(n_x, m)`

7. Recall that `np.dot(a,b)` performs a matrix multiplication on a and b, whereas `a*b` performs an element-wise multiplication.

    Consider the two following random arrays "a" and "b":

    ```
    a = np.random.randn(12288, 150) # a.shape = (12288, 150)
    b = np.random.randn(150, 45) # b.shape = (150, 45)
    c = np.dot(a, b)
    ```

    What is the shape of c?

    `c.shape = (12288, 45)`, this is a simple matrix multiplication example.

8. Consider the following code snippet:

    ```
    # a.shape = (3,4)
    # b.shape = (4,1)
    for i in range(3):
      for j in range(4):
        c[i][j] = a[i][j] + b[j]
    ```

    How do you vectorize this?

    `c = a + b.T`

9. Consider the following code:

    ```
    a = np.random.randn(3, 3)
    b = np.random.randn(3, 1)
    c = a * b
    ```

    What will be c?

This will invoke broadcasting, so b is copied three times to become (3,3), and $*$ is an element-wise product so `c.shape = (3, 3)`.

10. Consider the following computation graph.

```
J = u + v - w
  = a * b + a * c - (b + c)
  = a * (b + c) - (b + c)
  = (a - 1) * (b + c)
```

Answer: `(a - 1) * (b + c)`