$b = np.random.randn(2, 1)$ # $b.shape = (2, 1)$

$c = a + b$

What will be the shape of c?

- The computation cannot happen because the sizes don't match. It's going to be "Error"!
- c.shape = (2, 1)
- c.shape = (3, 2)
- c.shape = (2, 3) (selected)

Expand

Correct
Yes! This is broadcasting. b (column vector) is copied 3 times so that it can be summed to each column of a.

5. Consider the two following random arrays a and b: **1 / 1 point**

$a = np.random.randn(1, 3)$ # $a.shape = (1, 3)$

$b = np.random.randn(3, 3)$ # $b.shape = (3, 3)$

$c = a * b$

What will be the shape of c?

- The computation cannot happen because the sizes don't match.
- c.shape = (3, 3) (selected)
- c.shape = (1, 3)
- The computation cannot happen because it is not possible to broadcast more than one dimension.

Expand

Correct
Yes. Broadcasting allows row a to be multiplied element-wise with each row of b to from c.

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? **1 / 1 point**

- (m, n_x)
- $(m, 1)$
- $(1, m)$
- (n_x, m) (selected)

Expand

Correct

7. Consider the following array: **1 / 1 point**

$a = np.array([[2, 1], [1, 3]])$

What is the result of $a * a$?

- $\begin{pmatrix} 4 & 2 \end{pmatrix}$

Expand

Incorrect

9. Consider the following code: **1 / 1 point**

$a = np.random.randn(3, 3)$

$b = np.random.randn(3, 1)$

$c = a * b$

What will be c? (If you're not sure, feel free to run this in python to find out).

- It will lead to an error since you cannot use "*" to operate on these two matrices. You need

to instead use np.dot(a,b)

- This will invoke broadcasting, so b is copied three times to become (3, 3), and $*$ invokes a matrix multiplication operation of two 3x3 matrices so c.shape will be (3, 3)
- This will multiply a 3x3 matrix a with a 3x1 vector, thus resulting in a 3x1 vector. That is, c.shape = (3,1).
- This will invoke broadcasting, so b is copied three times to become (3,3), and $*$ is an element-wise product so c.shape will be (3, 3)

Expand

Correct

Expand

Correct
Yes. Broadcasting allows row a to be multiplied element-wise with each row of b to from c.

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? **1 / 1 point**

- (m, n_x)
- $(m, 1)$
- $(1, m)$
- (n_x, m)

Expand

Correct

7. Consider the following array: **1 / 1 point**

$a = np.array([[2, 1], [1, 3]])$

What is the result of $a * a$?

- $\begin{pmatrix} 4 & 2 \\ 2 & 6 \end{pmatrix}$
- $\begin{pmatrix} 5 & 5 \\ 5 & 10 \end{pmatrix}$
- $\begin{pmatrix} 4 & 1 \\ 1 & 9 \end{pmatrix}$
- The computation cannot happen because the sizes don't match. It's going to be an

Expand

Correct
Yes, recall that * indicates element-wise multiplication.

8. Consider the following code snippet: **0 / 1 point**

$a.shape = (3, 4)$

$b.shape = (4, 1)$

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
- c = a + b
- c = a.T + b
- c = a.T + b.T

Expand

Incorrect

9. Consider the following code: **1 / 1 point**

$a = np.random.randn(3, 3)$

$b = np.random.randn(3, 1)$

$c = a * b$

What will be c? (If you're not sure, feel free to run this in python to find out).

- It will lead to an error since you cannot use "*" to operate on these two matrices. You need to instead use np.dot(a,b)
- This will invoke broadcasting, so b is copied three times to become (3, 3), and $*$ invokes a matrix multiplication operation of two 3x3 matrices so c.shape will be (3, 3)
- This will multiply a 3x3 matrix a with a 3x1 vector, thus resulting in a 3x1 vector. That is, c.shape = (3,1).
- This will invoke broadcasting, so b is copied three times to become (3,3), and $*$ is an element-wise product so c.shape will be (3, 3)

Expand

Correct

- $\begin{pmatrix} 2 & 6 \end{pmatrix}$
- $\begin{pmatrix} 5 & 5 \\ 5 & 10 \end{pmatrix}$
- $\begin{pmatrix} 4 & 1 \\ 1 & 9 \end{pmatrix}$
- The computation cannot happen because the sizes don't match. It's going to be an

Expand

Correct
Yes, recall that * indicates element-wise multiplication.

8. Consider the following code snippet: **0 / 1 point**

$a.shape = (3, 4)$

$b.shape = (4, 1)$

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
- c = a + b
- c = a.T + b
- c = a.T + b.T