1 point 1. Suppose I first execute the following Octave/Matlab commands:

Which of the following are then valid commands? Check all that apply. (Hint: A' denotes the transpose of A.)

Answer: A and B

1 point 2. $Let A = \begin{bmatrix} 16 & 2 & 3 & 13 \\ 5 & 11 & 10 & 8 \\ 9 & 7 & 6 & 12 \\ 4 & 14 & 15 & 1 \end{bmatrix}.$

Which of the following indexing expressions gives $B = \begin{bmatrix} 16 & 2 \\ 5 & 11 \\ 9 & 7 \\ 4 & 14 \end{bmatrix}$? Check all that apply.

Answer: A and B

1 point 3. Let A be a 10x10 matrix and x be a 10-element vector. Your friend wants to compute the product Ax and writes the following code:

```
1 v = zeros(10, 1);

2 for i = 1:10

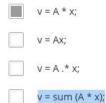
3 for j = 1:10

4 v(i) = v(i) + A(i, j) * x(j);

5 end

6 end
```

How would you vectorize this code to run without any FOR loops? Check all that apply.



Answer: A

1 point 4. Say you have two column vectors v and w, each with 7 elements (i.e., they have dimensions 7x1). Consider the following code:

```
1 z = 0;

2 for i = 1:7

3 z = z + v(i) * w(i)

4 end
```

Which of the following vectorizations correctly compute z? Check all that apply.

```
z = sum (v .* w);

z = v' * w;

z = v * w';

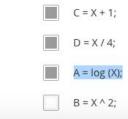
z = v .* w;
```

Answer: A and B

1 point 5. In Octave/Matlab, many functions work on single numbers, vectors, and matrices. For example, the sin function when applied to a matrix will return a new matrix with the sin of each element. But you have to be careful, as certain functions have different behavior. Suppose you have an 7x7 matrix *X*. You want to compute the log of every element, the square of every element, add 1 to every element, and divide every element by 4. You will store the results in four matrices, *A*, *B*, *C*, *D*. One way to do so is the following code:

```
for i = 1:7
for j = 1:7
A(i, j) = log(X(i, j));
B(i, j) = X(i, j) ^ 2;
C(i, j) = X(i, j) + 1;
D(i, j) = X(i, j) / 4;
end
end
```

Which of the following correctly compute A, B, C, or D? Check all that apply.



Answer: A, B and C