B.Tech (VII)- IT Workshop/MATLAB

Unit 3 (Part 1)

Lab Assignment/Practise Question set 04

NOTE:

- Explore MATLAB Desktop environment before starting these experiments.
- All the variables provided are case sensitive.

1. If a variable has the dimensions 3 × 4, could it be considered to be (check all that apply):
   a matrix
   a row vector
   a column vector
   a scalar
2. If a variable has the dimensions 1 × 5, could it be considered to be (check all that apply):
   a matrix
   a row vector
   a column vector
   a scalar
3. If a variable has the dimensions 5 × 1, could it be considered to be (check all that apply):
   a matrix
   a row vector
   a column vector
   a scalar
4. If a variable has the dimensions 1 × 1, could it be considered to be (check all that apply):
   a matrix
   a row vector
   a column vector
   a scalar
5. Using the colon operator, create the following row vectors

```
2     3     4     5     6     7
1.1000    1.3000    1.5000    1.7000
8     6     4     2
```

6. Using a built-in function, create a vector *vec* which consists of 20 equally spaced points in the range of – pi to +pi.
7. Write an expression using linspace that will result in the same as 2: 0.2: 3

8. Using the colon operator and also the **linspace** function, create the following row vectors:

```
-5  -4  -3  -2  -1
 5   7   9
 8   6   4
```

9. How many elements would be in the vectors created by the following expressions?

```
linspace(3,2000)
logspace(3,2000)
```

10. Create a variable *myend* which stores a random integer in the inclusive range of 5 to 9. Using the colon operator, create a vector that iterates from 1 to *myend* in steps of 3.
11. Using the colon operator and the transpose operator, create a column vector *myvec* that has the values −1 to 1 in steps of 0.5.
12. Write an expression that refers to only the elements that have odd-numbered subscripts in a vector, regardless of the length of the vector. Test your expression on vectors that have both an odd and even number of elements.
13. Generate a 2 × 4 matrix variable *mat*. Replace the first row with 1:4. Replace the third column (you decide with which values).
14. Generate a 2 × 4 matrix variable *mat*. Verify that the number of elements is the product of the number of rows and columns.
15. Which would you normally use for a matrix: **length** or **size**? Why?
16. When would you use **length** vs. **size** for a vector?
17. Generate a 2 × 3 matrix of random
    real numbers, each in the range (0, 1)
    real numbers, each in the range (0, 10)
    integers, each in the inclusive range from 5 to 20
18. Create a variable *rows* that is a random integer in the inclusive range from 1 to 5. Create a variable *cols* that is a random integer in the inclusive range from 1 to 5. Create a matrix of all zeros with the dimensions given by the values of *rows* and *cols*.
19. Create a matrix variable *mat*. Find as many expressions as you can that would refer to the last element in the matrix, without assuming that you know how many elements or rows or columns it has (i.e., make your expressions general).
20. Create a vector variable *vec*. Find as many expressions as you can that would refer to the last element in the vector, without assuming that you know how many elements it has (i.e., make your expressions general).
21. Create a 2 × 3 matrix variable *mat*. Pass this matrix variable to each of the following functions and make sure you understand the result: **flip**, **fliplr**, **flipud**, and **rot90**. In how many different ways can you **reshape** it?

22. What is the difference between `fliplr(mat)` and `mat = fliplr(mat)`?
23. Use **reshape** to reshape the row vector 1:4 into a 2x2 matrix; store this in a variable named mat. Next, make 2×3 copies of mat using both **repelem** and **repmat**.