ENGR 15100: Software Tools for Engineers
Spring 2015

Computer Assignment #4

Due: Tuesday, February 17, 2015, 9am CST

Departments of Engineering
School of Engineering, Mathematics, & Sciences
Purdue University Calumet

PURDUE
UNIVERSITY
CALUMET

1. OBJECTIVE

Become familiar with generating random numbers and obtaining Console Window user input. Additionally, continue practicing with element-by-element array operations, analyzing arrays with built-in functions, and solving systems of linear equations.

2. PROCEDURE

Task I

Create a MATLAB script file having the name `LASTNAME_LAB4.m` and perform the following sequence of steps in the file. ***Unless otherwise specified, do not suppress the output to the MATLAB Command Window.***

(a) **[1 point]** Clear the MATLAB Workspace and clear the contents of the MATLAB Command Window.

(b) **[0.5 points]** Activate a *diary* in a file named `LASTNAME_LAB4_DIARY.txt`.

(c) **[1 point]** Create a *variable* named `full_name` and assign to it a string indicating your full first and last name separated by a blank space. Suppress the output to the MATLAB Command Window. Then, using variable `full_name` in combination with the **`disp()`** function, display your full name in the MATLAB Command Window.

(d) **[1 point]** Create a variable named `lab_part` and assign to it the string `'Lab #4: Part #1'`. Suppress the output to the MATLAB Command Window. Then, display the contents stored in variable `lab_part` to the MATLAB Command Window using built-in function **`disp()`**.

(e) **[9 points]** Using built-in `input()` function, prompt the user to enter *scalar values* for variables named `smallest`, `largest`, and `number_of_elements`. After completing this step, you should see an output in the MATLAB Command Window similar to the sample output shown below. Suppress the output to the MATLAB Command Window.

```
Enter an integer for variable smallest: <entered value>
Enter an integer for variable largest: <entered value>
Enter an integer for the length of a row vector t: <entered value>
```

When testing your script, enter the scalar values -5, 5, and 10 for `smallest`, `largest`, and `number_of_elements`, respectively.

(f) **[6 points]** Create a row vector named `t` and assign to it a `1 x number_of_elements` row vector whose elements are each randomly generated real numbers chosen uniformly from the open interval (`smallest`, `largest`). Use built-in function **`rand()`**.

(g) **[12 points]** Create a *variable* named `f` and assign to it a row vector whose elements are values obtained from evaluating the expression below for corresponding elements of row vector `t`.

$$f = 2te^{-2t}\cos\left(\frac{20}{t} + \pi\right)$$

(h) **[1 point]** Assign to `lab_part` the string `'Lab #4: Part #2'`. Suppress the output to the MATLAB Command Window. Then, display the contents stored in variable `lab_part` to the MATLAB Command Window using built-in function **`disp()`**.

(i) **[10 points]** Create a variable named **A** and assign to it a 3×3 *coefficient matrix* obtained by prompting the user for three, 3-element row vectors corresponding to the 3 rows of *coefficient matrix* **A**. Prompt the user three times, once for each row vector.

After completing this step, you should see an output in the MATLAB Command Window similar to the sample output shown below. Suppress the output to the MATLAB Command Window.

```
Enter row 1 [a11 a12 a13] of coefficient matrix A: <enter vector here>
Enter row 2 [a21 a22 a23] of coefficient matrix A: <enter vector here>
Enter row 3 [a31 a32 a33] of coefficient matrix A: <enter vector here>
```

When testing your script, enter *row vectors* $[3 \quad -2 \quad 5]$, $[-4.5 \quad 2 \quad 3]$, and $[5 \quad 1 \quad -2.5]$ for rows 1 through 3 of *coefficient matrix* **A**, respectively.

(j) **[4 points]** Create a variable named **b** and assign to it a *constants column vector* obtained by prompting the user for such. Prompt the user once to enter 3 elements for the column vector

After completing this step, you should see an output in the MATLAB Command Window similar to the sample output shown below. Suppress the output to the MATLAB Command Window.

```
Enter constants column vector b [b1 b2 b3]: <enter vector here>
```

When testing your script, enter *constants column vector* $[7.5, 5.5, 4.5]^T$ for column vector **b**.

(k) **[4 points]** Create a variable named **x** and assign to it a *solutions column vector* obtained by solving the linear system of equations represented by $\boldsymbol{Ax} = \boldsymbol{b}$. Use the MATLAB left division operator \. Suppress the output to the MATLAB Command Window.

(l) **[6 points]** Report to the user the solution of the system of linear equations by displaying the column vector **x** to the MATLAB Command Window. Use the built-in function **`disp()`** twice.

After completing this step, you should see an output in the MATLAB Command Window similar to the sample output shown below. Suppress the output to the MATLAB Command Window.

```
The solution x = [x1 x2 x3] to the linear system Ax = b is:
   <value_of_column_vector_x>
```

(m) **[1 point]** Assign to `lab_part` the string `'Lab #4: Part #3'`. Suppress the output to the MATLAB Command Window. Then, display the contents stored in variable `lab_part` to the MATLAB Command Window using built-in function **`disp()`**.

(n) **[5 points]** Using built-in function **`input()`**, prompt the user to enter *scalar values* for variables named `number_of_rows` and `number_of_columns`. These two variables will be the dimensions of a matrix stored in a variable named `R` created as part of step (p) below. Prompt the user twice, once for the value of each variable.

After completing this step, you should see an output in the MATLAB Command Window similar to the sample output shown below. Suppress the output to the MATLAB Command Window.

```
Enter the number of rows of matrix R: <enter value here>
Enter the number of columns of matrix R: <enter value here>
```

When testing your script, enter scalar values 4 and 5 for `number_of_rows` and `number_of_columns`, respectively.

(o) **[3 points]** Using built-in function **`input()`**, prompt the user to enter *scalar integer values* for variables named `min_integer` and `max_integer`. Prompt the user twice.

After completing this step, you should see an output in the MATLAB Command Window similar to the sample output shown below. Suppress the output to the MATLAB Command Window.

```
Enter the minimum integer value: <enter value here>
Enter the maximum integer value: <enter value here>
```

When testing your script, enter scalar integer values −200 and +400 for `min_integer` and `max_integer`, respectively.

(p) **[5 points]** Create a *variable* named `R` and assign to it a `number_of_rows` x `number_of_columns` matrix whose elements are each randomly generated integers chosen uniformly from the closed interval `[min_integer, max_integer]`. Use the built-in function **`randi()`**.

(q) **[5 points]** Using one line of code, create a *variable* named `R_mean_rows` and assign to it a *column vector* whose elements are the mean (average) values of *each row* of matrix `R`. Use built-in function **`mean()`**.

(r) **[5 points]** Using one line of code, create a *variable* named `R_mean_all` and assign to it a scalar obtained by determining the mean (average) value of *all the elements* of matrix `R`. *Do not use the variable* `R_mean_rows` *as part of your code for this step.* Use built-in function **`mean()`**. Suppress the output to the MATLAB Command Window.

(s) **[5 points]** Using the built-in **`fprintf()`** function, display the scalar `R_mean_all` as a fixed-point (real number) showing a maximum of 3 digits after the decimal place.

After completing this step, you should see an output in the MATLAB Command Window similar to the sample output shown below. *Note, due to the random numbers, your mean value may be different.*

```
The mean of all the elements of matrix R is 70.200.
```

(t) **[4 points]** Using one line of code, create *a variable* named `R_sorted_by_columns` and assign to it a matrix obtained by sorting the elements of *each column* of matrix `R` in *ascending* order. Use built-in function **`sort()`**.

(u) **[5 points]** Using one line of code, create two variables named `R_min` and `R_min_row_index`. Assign `R_min` the *one element* of matrix `R` whose value is the minimum of the entire matrix. Assign to `R_min_row_index` the *row index (position)* of `R_min`. *Suppress the output to the MATLAB Command Window*.

(v) **[6 points]** Using the built-in **`fprintf()`** function, display the scalar `R_min` followed by the scalar `R_min_row_index` using a *single format string*.

- Format scalar `R_min` as an integer and display its value according to the default MATLAB settings.
- Format scalar `R_min_row_index` as an integer and display its value within a 2-digit field width showing a minimum of 2 digits.

After completing this step, you should see an output in the MATLAB Command Window similar to the example output shown below. *Note, due to the random numbers, your values may be different.*

```
The minimum value in matrix R is -198 and it resides in row #02.
```

(w) **[0.5 points]** Deactivate the *diary* in the file named `LASTNAME_LAB4_DIARY.txt`.

Task II

Upload the following files onto Blackboard Learn.

(a) Script file `LASTNAME_LAB4.m`.

(b) Diary text file `LASTNAME_LAB4_DIARY.txt`.