Programming Assignments

Introduction to Programming with MATLAB

Lesson 3

- Unless otherwise indicated, you may assume that each function will be given the correct number of inputs and that those inputs have the correct dimensions. For example, if the input is stated to be three row vectors of four elements each, your function is not required to determine whether the input consists of three two-dimensional arrays, each with one row and four columns.
- Unless otherwise indicated, your function should not print anything to the Command Window, but your function will not be counted incorrect if it does.
- Note that you are not required to use the suggested names of input variables and output variables, but you must use the specified function names.
- Also, read the instructions on the web page on how to test your functions with the auto-grader program provided, and what to submit to Coursera to get credit.
- Note that we have not covered if-statements or loops, so they are neither needed nor allowed!

1. Write a function called **`circle`** that takes a scalar input **`r`**. It needs to return an output called **`area`** that is the area of a circle with radius **`r`** and a second output, **`cf`** that is the circumference of the same circle. You are allowed to use the built-in function **`pi`**. In fact, you need to use it to get the value of π as accurately as possible.

2. Write a function called **`even_index`** that takes a matrix, **`M`**, as input argument and returns a matrix that contains only those elements of **`M`** that are in even rows and columns. In other words, it would return the elements of **`M`** at indices (2,2), (2,4), (2,6), …, (4,2), (4,4), (4,6), …, etc. Note that both the row and the column of an element must be even to be included in the output. The following would not be returned: (1,1), (2,1), (1,2) because either the row or the column or both are odd. As an example, if **`M`** were a 5-by-8 matrix, then the output must be 2-by-4 because the function omits rows 1, 3 and 5 of **`M`** and it also omits columns 1, 3, 5, and 7 of **`M`**.

3. Write a function called **`flip_it`** that has one input argument, a row vector **`v`**, and one output argument, a row vector **`w`** that is of the same length as **`v`**. The vector **`w`** contains all the elements of **`v`**, but in the exact opposite order. For example, is **`v`** is equal to **`[1 2 3]`** then **`w`** must be equal to **`[3 2 1]`**. You are not allowed to use the built-in function **`flip`**.

4. Write a function called **`top_right`** that takes two inputs: a matrix **`N`** and a scalar non-negative integer **`n`**, in that order, where each dimension of **`N`** is greater than or equal to **`n`**. The function returns the **`n`**-by-**`n`** square array at the top right corner of **`N`**.

5. Write a function called **`peri_sum`** that computes the sum of the elements of an input matrix **A** that are on the "perimeter" of **A**. In other words, it adds together the elements that are in the first and last rows and columns. Note that the smallest dimension of **A** is at least 2, but you do not need to check this.

6. Write a function called **`light_speed`** that takes as input a row vector of distances in kilometers and returns two row vectors of the same length. Each element of the first output argument is the time in minutes that light would take to travel the distance specified by the corresponding element of the input vector. To check your math, it takes a little more than 8 minutes for sunlight to reach Earth which is 150 million kilometers away. The second output contains the input distances converted to miles. Assume that the speed of light is 300,000 km/s and that one mile equals 1.609 km.

7. Write a function that is called like this: **`amag = accelerate(F1,F2,m)`**. **`F1`** and **`F2`** are three-element column vectors that represent two forces applied to a single object. The argument **`m`** equals the mass of the object in units of kilograms. The three elements of each force equal the x, y, and z components of the force in Newtons. The output variable **`amag`** is a scalar that is equal to the magnitude of the object's acceleration. The function calculates the object's acceleration vector $\mathbf{a}$ by using Newton's law: $\mathbf{F} = m\mathbf{a}$, where $\mathbf{F}$ is the sum of **`F1`** and **`F2`**. Then it returns the magnitude of $\mathbf{a}$.

8. Write a function called **`income`** that takes two row vectors of the same length as input arguments. The first vector, **`rate`** contains the number of various products a company manufactures per hour simultaneously. The second vector, **`price`** includes the corresponding per item price they sell the given product for. The function must return the overall income the company generates in a week assuming a 6-day work week and two 8-hour long shifts per day.