ENG1060: COMPUTING FOR ENGINEERS

PASS 6 – Week 8

2020 OCT NOV

TASK 1

A vector is given by V = [4, 9, -3, 12, 0, -9, 15, 17, 25, -6, 4, 10, -2, 15]

Write an m-file that:

A. Prints the length of the original vector using fprintf.
B. Prints a new vector that
   - doubles the elements that are positive and divisible by 3 or 5
   - raises to the power of 3 the elements that are negative but greater than -5
   - excludes all other elements

   Conditions are checked in the order listed above.
C. Prints the new vector (use disp) and its length using fprintf

New vector has a length of X and contains:
  Y Y Y Y Y

TASK 2

a) Create a function file that produces the x, y coordinates of a polar plot. The function file should take inputs of a function handle and the number of revolutions over which the polar function (f) is to be plotted. Use the function declaration as shown below:

   `function [x,y] = fun_polar(fhandle, N)`

   Note that 1 revolution is defined as θ from 0 to 2π. Use a resolution of 0.01 radians. The required equations to project the polar function into Cartesian coordinates are

   $$x = f\cos(\theta); \quad Y = f\sin(\theta).$$

b) Create function handles for the 4 equations below and store them in a cell array using `fset = {f1;f2;f3;f4}`. Then use a for loop to plot the equations on Cartesian axes in a 2x2 subplot arrangement using $N = 10$ in these cases and set the axes limits to be 'equal'. The titles can be generated in the for loop using `title(char(fset{<index>})`.

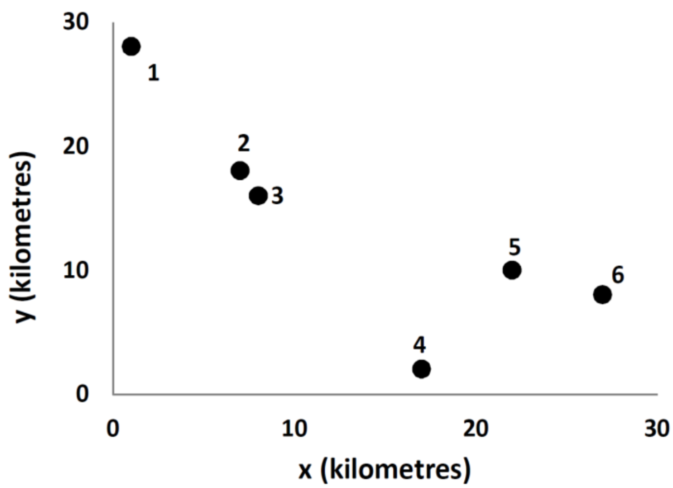
   i) $f_1 = \exp(0.01\theta)\sin(5\theta)$
   ii) $f_2 = \sin(10\theta) + 0.15$
   iii) $f_3 = \sin(0.5\theta)\,(0.85 + \sin(\theta))$
   iv) $f_4 = 10 + \sin(2\pi\theta)$

TASK 3

Engineers in industry must continually look for ways to make their designs and operations more efficient. One way of doing so is optimisation, which uses a mathematical description of the design or operation to select the best values of certain variables. Often these problems have a limited number of possible variables and can use MATLAB loop structures to search for the optimum solution.

A company wants to locate a distribution centre that will serve six of its major customers in a 30 x 30 km square area. The locations of the customers relative to the southwest corner of the area are given in the following table in terms of (x, y) coordinates (the x-direction is east; the y-direction is North) (see the figure below). The volume in tons per week that must be delivered from the distribution centre to each customer is also given (see the table below).

The weekly delivery cost c_i for customer **i** depends on the volume V_i and the distance d_i from the distribution centre. For simplicity, we will assume that this distance is the straight line distance (This assumes that the road network is dense).



Customer	x location (km)	y location (km)	Volume (tons/week)
1	1	28	5
2	7	18	11
3	8	16	1
4	17	2	9
5	22	10	7
6	27	8	6

The weekly cost is given by $\mathbf{c_i} = 0.5*\mathbf{d_i}*\mathbf{V_i}$ (where i = 1, 2, … 6).

A. Write an m-file that finds the location of the distribution centre (to the nearest km) that has the lowest total weekly cost to service all six customers.

B. Reproduce the figure above and in addition, mark the location of the distribution centre that has the lowest total weekly cost with a red coloured solid diamond. Fill in all symbols with the 'MarkerFaceColor' plot option. You do not need to label the data points.

Hint: You may need to use two min functions (i.e. min(min(X))) to find the minimum value within a matrix.

TASK 4

The rounded-square-root (RSR) of a positive integer *n* is defined as the square root of *n* rounded to the nearest integer. Adapting Heron's method to integer arithmetic allows the rounded-square-root of *n* to be calculated as follows. Let *d* be the number of digits of number *n*.

$$\text{If } d \text{ is odd, then } x_0 = 2 \times 10^{\frac{d-1}{2}}$$
$$\text{If } d \text{ is even, then } x_0 = 7 \times 10^{\frac{d-2}{2}}$$

where x_0 is the starting guess for the rounded-square-root of *n*. The proceeding guess is calculated using the current guess as:

$$x_1 = \left\lfloor \frac{x_0 + \left\lceil \frac{n}{x_0} \right\rceil}{2} \right\rfloor$$

where $\lfloor x \rfloor$ and $\lceil x \rceil$ (notice the ticks at the top and bottom of the brackets) represent the floor and ceiling of *x*, respectively. This pattern repeats such that,

$$x_2 = \left\lfloor \frac{x_1 + \left\lceil \frac{n}{x_1} \right\rceil}{2} \right\rfloor \qquad x_3 = \left\lfloor \frac{x_2 + \left\lceil \frac{n}{x_2} \right\rceil}{2} \right\rfloor \qquad x_4 = \left\lfloor \frac{x_3 + \left\lceil \frac{n}{x_3} \right\rceil}{2} \right\rfloor$$

Hence, the general equation is:

$$x_{i+1} = \left\lfloor \frac{x_i + \left\lceil \frac{n}{x_i} \right\rceil}{2} \right\rfloor$$

Naming x_i as *xi* and x_{i+1} as *xi1*, the equation is given as *xi1*=floor((*xi*+ceil(*n*/*xi*))/2). For the next iteration, x_i becomes x_{i+1} and x_{i+1} is recalculated using this new x_i. **Repeat this process until $x_{i+1} = x_i$.**

1. Write a function named *numdigs* that takes *n* as an input, and outputs the number of digits *d*. Hint: convert the number to a string and take the length of the string.
2. Write a script that calculates the RSR value for a positive integer *n*. It is strongly recommended that you perform several iterations of this process by hand to obtain a deeper understanding of the changes in variables required. As a check, below is a table of *n,* corresponding RSR and number of iterations taken.

n	RSR	Iterations taken
1234	35	4
1337	37	3
5678	75	3

Modify your script such that it calculates the RSR values for all 4-digit numbers (i.e. *n*=1000:9999). You may want to make a copy of your code in step 2 as a backup.