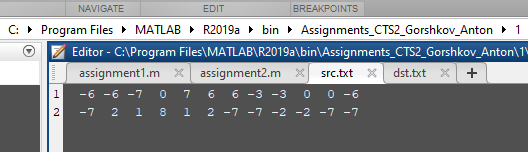
**CTS2 LINA MATLAB Gorshkov Anton**

**Assignment 1**

Here is a content of **src.txt**:

-6 -6 -7 0 7 6 6 -3 -3 0 0 -6

-7 2 1 8 1 2 -7 -7 -2 -2 -7 -7



Here is a code of the function:

% Write a MATLAB-function, that rotates an object O in the x-y-coordinate system of R2 by an angle alpha and stretches or shrinks it by a factor lambda. Both alpha and lambda

% are parameters of your function. The function you write needs to read the rectangular coordinates of the object O from a data file. The object is assumed to have only a finite

% number of vertices. Any two vertices are joined by a stright line. If the object, for example, is a triangle it can be described by three vertices, having coordinates Vi = (xi; yi),

% i = 1; 2; 3. Theese coordinates are organised as columns in a data file.

% V1 V2 V3

% x1 x2 x3

% y1 y2 y3

% Object O needs to be displayed graphically before and after its transformation. Your function also needs to be able to save the coordinates of the transformed object O1 in an

% external data file in the described manner on demand. Test your generated MATLAB file with the following coordinates of an object O, having the following twelf vertices.

% -6 -6 -7 0 7 6 6 -3 -3 0 0 -6

% -7 2 1 8 1 2 -7 -7 -2 -2 -7 -7

% This object O needs to be rotated counter clockwise by alpha = 45 degrees and stretched by lambda = 2.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [] = assignment1(varargin)

srcFileName = 'src.txt'; % src file path initialization

dstFileName = 'dst.txt'; % dst file path initialization

%%%%% Setting default values in case when no parameters are passed %%%%%%%%%%%%%%%%%%%%%

if nargin < 2

transfMagn = 2; % set default transformation magnitude initialization

transfAngleDeg = 45; % set default transformation angle in degree initialization

else

transfMagn = varargin{1}; % set src file path from the function parameters

transfAngleDeg = varargin{2}; % set src file path from the function parameters

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

srcFileID = fopen(srcFileName,'r'); % Open the src file for reading

xSrcStr = fgetl(srcFileID); % Read the X-coordinates from the src file as a string

ySrcStr = fgetl(srcFileID); % Read the Y-coordinates from the src file as a string

fclose(srcFileID); % Close the src file

xSrcNum = str2num(xSrcStr); % Convert X-coordinates into numbers

ySrcNum = str2num(ySrcStr); % Convert Y-coordinates into numbers

complexSrc = complex(xSrcNum,ySrcNum); % Create a src-complex array for the next transformation

plot(complexSrc); % Draw the src-object

disp('Press any key for transformation') % Displaying 'Press any key for transformation'

pause; % Wait for the pressing any key. Meanwhile we can see to the untransformed object

transfAngleRad = deg2rad(transfAngleDeg); % Transformation angle in degree into radians convertion

transfComlpex = transfMagn\*exp(transfAngleRad\*1i); % Transformation comlpex initialization

complexDst = complexSrc \* transfComlpex; % Applying transformation into src-object using src-complex and transformation complex.

xDstNum = real(complexDst); % Get an array of real parts of the transformed object

yDstNum = imag(complexDst); % Get an array of imeginary parts of the transformed object

hold on; % retain plots in the current axes so that new plots added to the axes do not delete existing plots.

plot(complexDst); % Draw the dst-object

axis equal; % use equal data unit lengths along each axis.

dstFileID = fopen(dstFileName, 'w'); % Open the src file for reading

len = length(xSrcNum); % Find the number of vertices

vertices = 1:len; % Initialize vertices names

dstData = [vertices; xDstNum; yDstNum]; % Create dst-coordinates matrix

fprintf(dstFileID, '%9.dV', dstData(1,:)); % Print the vertices numbers of the transformed object

fprintf(dstFileID, '\n');

fprintf(dstFileID, '%10.3f', dstData(2,:)); % Print the x-values of the transformed object

fprintf(dstFileID, '\n');

fprintf(dstFileID, '%10.3f', dstData(3,:)); % Print the y-values of the transformed object

fprintf(dstFileID, '\n');

fclose(dstFileID); % Close dst-file

hold off; % set the hold state to off so that new plots added to the axes clear existing plots and reset all axes properties.

end

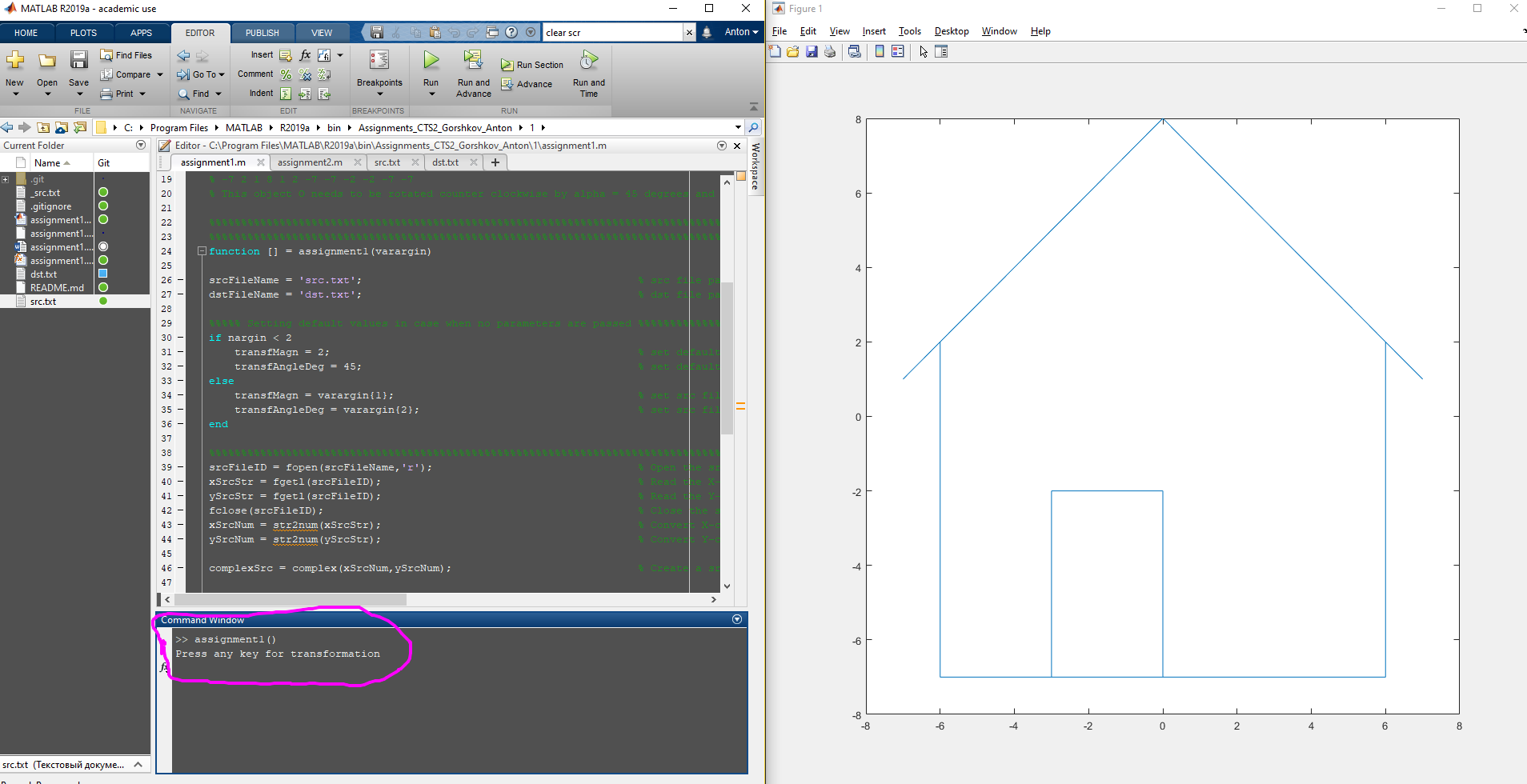
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

It is possible to launch function in several modes:

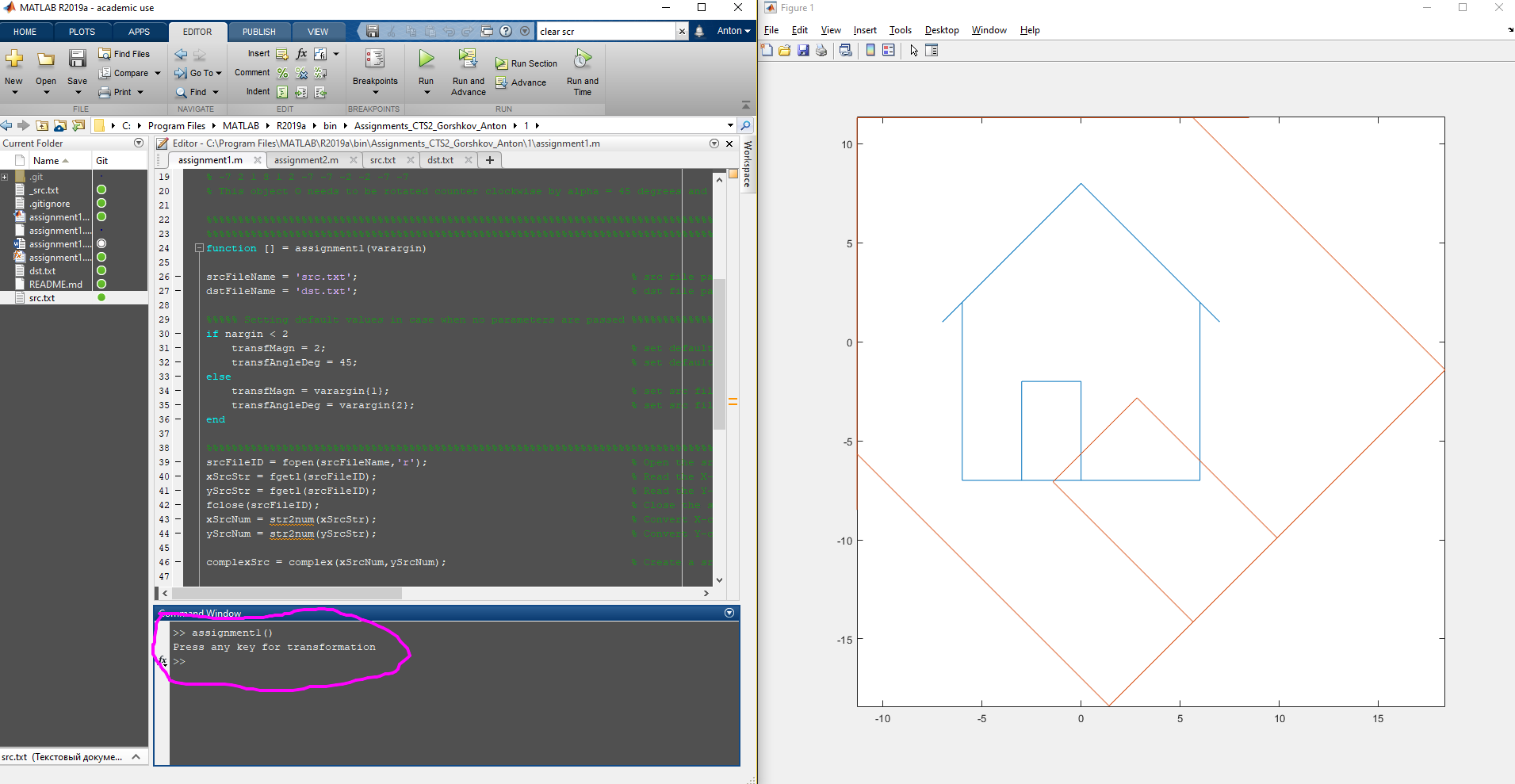
* with particular transformation magnitude and angle.
* with default transformation magnitude and angle (2 and 45 degrees respectively).

Let us try to execute the function without any parameters (with default values).

When the function executes, the original object appears with an invitation to transform it.



Then, customer can press any key and the transformation is executed:



After the program execution, the content of **dst.txt** is:

1V 2V 3V 4V 5V 6V 7V 8V 9V 10V 11V 12V

1.414 -11.314 -11.314 -11.314 8.485 5.657 18.385 5.657 -1.414 2.828 9.899 1.414

-18.385 -5.657 -8.485 11.314 11.314 11.314 -1.414 -14.142 -7.071 -2.828 -9.899 -18.385

