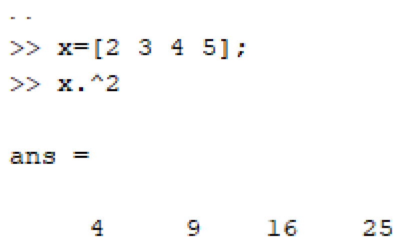
Lab EXE 2

1. Some Matlab basic coding

1.1 Matlab square operation on an array:

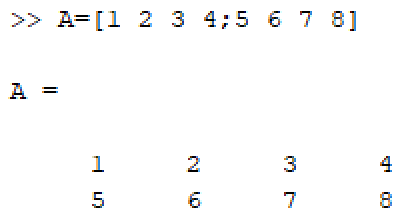
Taking square on each element of an array;



```
>> x=[2 3 4 5];
>> x.^2

ans =

     4     9    16    25
```

1.2 A 2D array

A=[1 2 3 4;5 6 7 8];



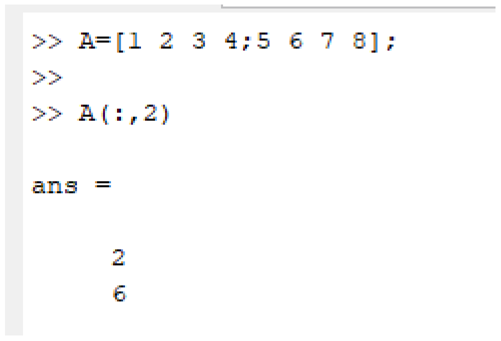
```
>> A=[1 2 3 4;5 6 7 8]

A =

     1     2     3     4
     5     6     7     8
```

Now each column of the 2D array is a column vector.

Get a column from a 2D array

A(:,2)

We get



```
>> A=[1 2 3 4;5 6 7 8];
>>
>> A(:,2)

ans =

     2
     6
```

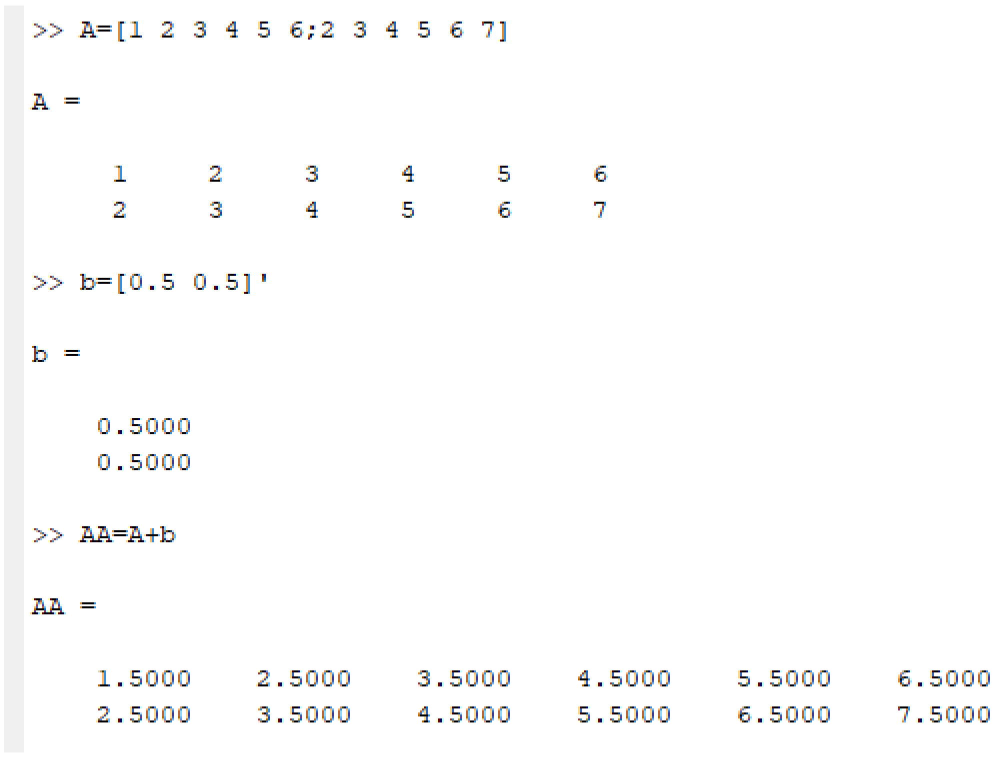
1.3 Adding each column vector of the 2D array $\boldsymbol{A}$ with a vector $\boldsymbol{b}$

(Of course, the numbers of rows in $\boldsymbol{A}$ and $\boldsymbol{b}$ should be equal)

In Matlab, you can use

$$\boldsymbol{AA} = \boldsymbol{A} + \boldsymbol{b}$$

Example code:



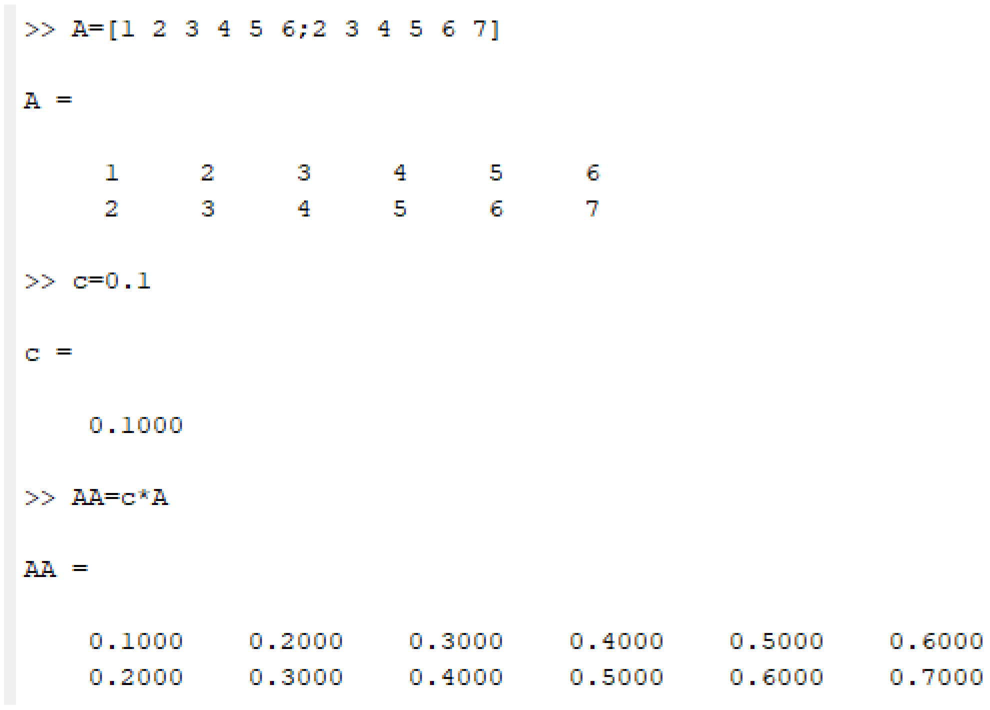
```
>> A=[1 2 3 4 5 6;2 3 4 5 6 7]

A =

     1     2     3     4     5     6
     2     3     4     5     6     7

>> b=[0.5 0.5]'

b =

    0.5000
    0.5000

>> AA=A+b

AA =

    1.5000    2.5000    3.5000    4.5000    5.5000    6.5000
    2.5000    3.5000    4.5000    5.5000    6.5000    7.5000
```

1.3 Multiply each element of a 2D array $\boldsymbol{A}$ with a constant scalar c

In Matlab, you can use

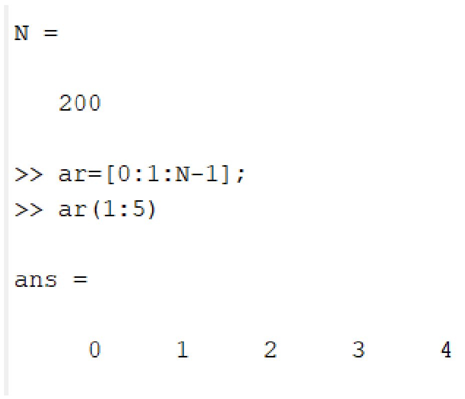
$$\boldsymbol{AA} = c * \boldsymbol{A}$$

Example code:



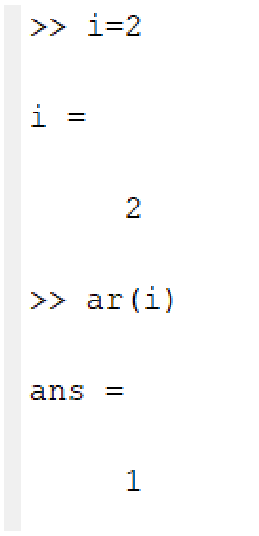
```
>> A=[1 2 3 4 5 6;2 3 4 5 6 7]

A =

     1     2     3     4     5     6
     2     3     4     5     6     7

>> c=0.1

c =

    0.1000

>> AA=c*A

AA =

    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000
    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000
```

1.4 Define array for with fixed integer increments

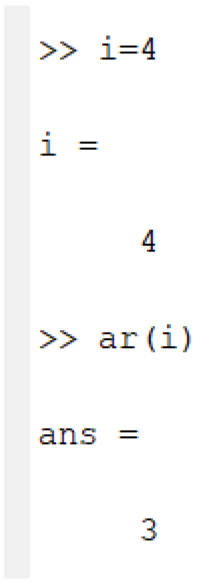


```
N =

   200

>> ar=[0:1:N-1];
>> ar(1:5)

ans =

     0     1     2     3     4
```

Get the i-th element of ar

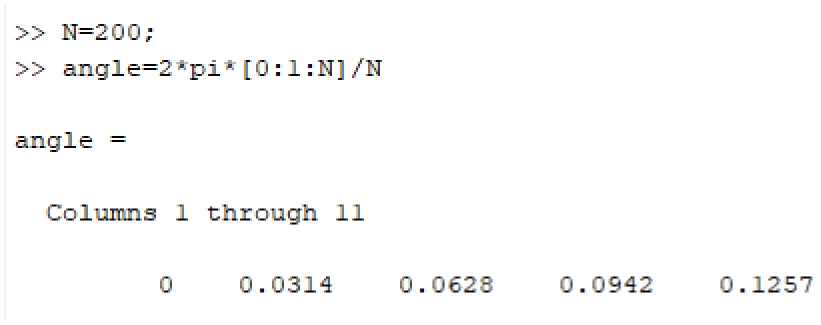


```
>> i=2

i =

     2

>> ar(i)

ans =

     1
```



```
>> i=4

i =

     4

>> ar(i)

ans =

     3
```

1.5 Define an array for angles from 0 to 2π



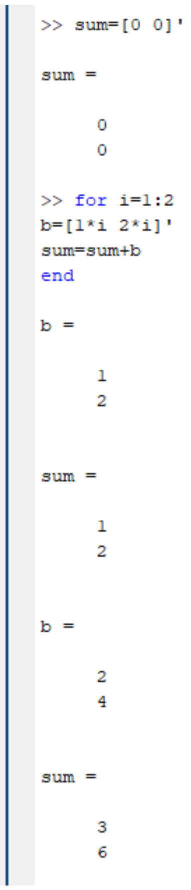
```
>> N=200;
>> angle=2*pi*[0:1:N]/N

angle =

  Columns 1 through 11

         0    0.0314    0.0628    0.0942    0.1257
```

1.6 Matlab for loop. `for` loop to repeat specified number of times

Syntax

```
for index = values
    statements
end
```

Example



```
>> sum=[0 0]'

sum =

     0
     0

>> for i=1:2
b=[1*i 2*i]'
sum=sum+b
end

b =

     1
     2


sum =

     1
     2


b =

     2
     4


sum =

     3
     6
```

2. Introduction

The aim of this laboratory is to let you to understand the basic concept of using vectors in computer graphics. First at all, you should review your notes about vector displacement and scalar vector product. We will use a simple matlab code to visualize our result.

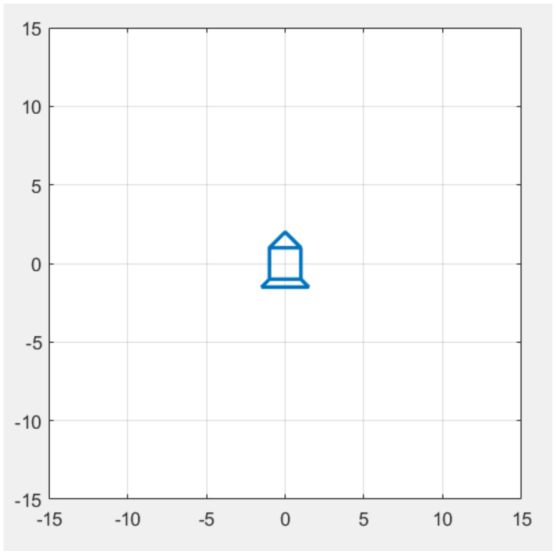


Figure 1 Display an object in the MatLab figure window.

In computer graphics, there are two concepts to represent images. They are raster graphics and vector graphics. The raster graphics concept directly stores the pixel values of the image as a 2D array. For example, if the image resolution is 256×256, the raster graphics concept stores all the 65,536 pixels, as a 2D array. That is a table with 256 columns and 256 rows.

The vector graphics concept does not directly store an image. It uses a number of control points to represent an object. For example, the object, shown in Figure 1, can be represented by a 2×11 matrix (11 2D control points), given by

data matrix =
```
[1    1  -1   -1  1  0 -1 -1 -1.5   1.5  1;
 1   -1  -1    1  1  2  1 -1 -1.5  -1.5 -1];
```

Each column vector of the matrix defines a control point. To draw the object, the machine first gets the initial position by picking up the first control point. It then draws a line from the first control point to the second point, a line from the second control point to the third control point, and so on.

Question 1
Discuss main differences between raster graphics and vector graphics. (around than 100 -150 words) Hint: use google to study but use your own words.

Table 1 shows a basic MatLab program to draw an object in the Matlab figure window. The program has sufficient comments to describe the way to program.

Table 1. Sample code

```
% define control points of the object
theta=[2*pi*[0:40]/40];
 x=[cos(theta);sin(theta)];

% you can define some operations to modify the object, such as
% scaling an object and translating an object (modifying
% rather using xx=x.
% The for-loop is to run the statements 200 times.
% If your operations on "xx= .... " do not change with variable i, then
% the object does not change with time.
% On the other hand, if your operations depend on i, then you will
% find that you can get an animation.

N=200;
for i=1:N
%
xx=x;
% draw the object
% Plot is to draw the object with line width equal to 2
% axis is to define the display range
% daspect is to define aspect ratio to 1:1
% grid on is to grid lines in the display window
% draw now is to tell draw the object now and delay the a while
plot(xx(1,:),xx(2,:),'Linewidth',1);
axis([-20, 20, -20, 20]);
daspect([1 1 1])
grid on
drawnow
end
```

3. Part 1 experiment:

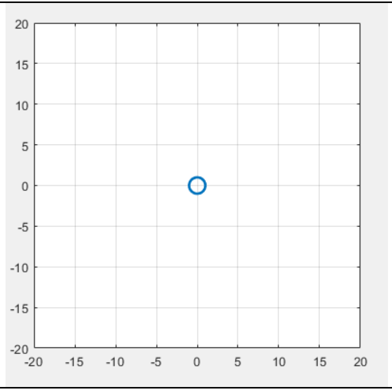


Figure 2. The output of Step1.

1. Run the program, you can get Figure 2.
2. Now change
   `x=xx` to `x=xx+[10 15]'`
   Question 2: What is the output? Explain.
   Question 3: Write a matlab program to produce an animation (see video1 in canvas).
   Hints: Inside the for-loop, you need to design a displacement vector whose elements depend on i. Thinking the initial position and the final position of the center of the object.
3. Now change
   `xx=x` to `xx=10*x`
   Question 4: What is the output? Explain.
   Question 5: Write a matlab program to produce an animation (see video2 in canvas).
   Hints: Inside the for-loop, you need to design a scalar whose elements depend on i. Thinking the initial size and the final size of the object.

4. Part 2 experiment:

According to gravitation force, after we launch a rock at angle α from ground with an initial velocity v_o, the rock goes through a path and eventually hits the ground, as shown in Fig The path that the rock goes through is the trajectory of the rock.

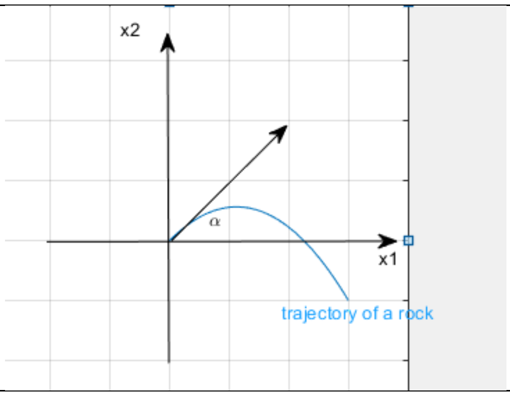


Figure 3. Trajectory of the rock.

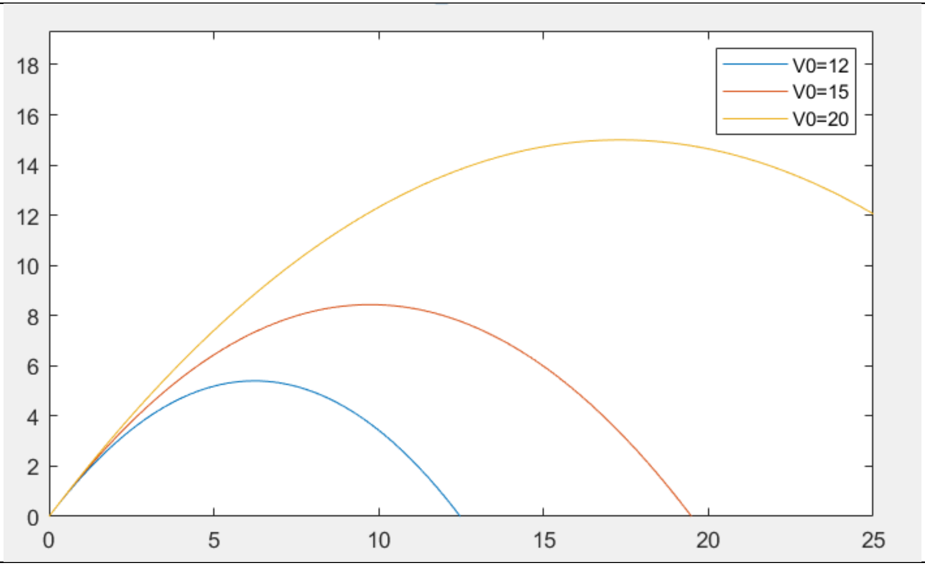


Figure 4. Trajectories of the rock with different initial velocity.

The trajectory can be described by the following equation:

$$x_2 = x_1 \tan(\alpha) - \frac{g x_1^2}{2 v_o^2 \cos^2(\alpha)} \tag{1}$$

where

- x_1 is the horizontal coordinate.
- x_2 is the vertical coordinate
- g is the gravitation constant (here we simplify it as 10)
- v_o is the initial velocity of the rock
- α is the launch angle.

In Table 2, we have an incomplete program for simulating the trajectory of a rock.
In the program, the statements:

```
angle=pi/3;
g=10;
vo=20;
dx1=[0:0.05:25];
dx2=dx1*tan(angle)- g*(dx1.^2)/(2*vo^2*(cos(angle)^2));
dx=[dx1;dx2];
```

are used to set $\alpha = \frac{\pi}{3}$, $v_o = 20$, and $g = 10$.
The program will stop when the ground is approximately hit.

Step 1
Question 6: Based on Table 2, write a matlab program to produce an animation (see video3 in canvas).

Step 2
Change the values of α and v_o such that the rock approximately hit the point= $\begin{pmatrix}20\\0\end{pmatrix}$.

Question 7: List two sets of α and v_o such that the rock approximately hits the point= $\begin{pmatrix}20\\0\end{pmatrix}$. You should provide some evidences to show that with your answer, the rock approximately hits the point= $\begin{pmatrix}20\\0\end{pmatrix}$.

Table 2. incomplete code for simulating a trajectory of a rock.

```
% define control points of the object
theta=[2*pi*[0:40]/40];
 x=[cos(theta);sin(theta)]

% define the trajectory
angle=pi/3;
g=10;
vo=20;
dx1=[0:0.05:25];
dx2=dx1*tan(angle)- g*(dx1.^2)/(2*vo^2*(cos(angle)^2));
dx=[dx1;dx2];

for i=1:length(dx1)
xx=x+????;
if ??? <0,
break
end
hold off
plot(xx(1,:),xx(2,:),'Linewidth',2);
hold on
plot(dx1,dx2,'Linewidth',1);
axis([0, 30, 0, 30]);
daspect([1 1 1])
grid on
drawnow
end
```

3. Report

1. Answer all the questions. You should use your own words rather than directly copying the text from Internet.
2. Perform all the steps in Section 2 and Section 3. Present and explain the algebra operations.
3. Include the Matlab Code and useful important figures.