Loops in MATLAB - 2

Lecture 6

Dr. Görkem Saygılı

Department of Biomedical Engineering
Ankara University

Introduction to MATLAB, 2017-2018 Spring

Outline:

In this lecture, we will learn:

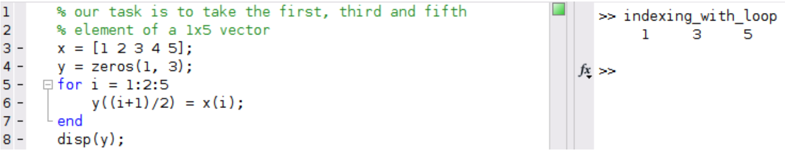
- for loop and while loop,
- break and continue statements,
- making loops more efficient in MATLAB,
- logical indexing.

Concept of Implicit Loop:

Implicit Loop: In MATLAB, we do not always have to use for or while loops explicity to do array operations. Since there is no loop index or loop body, programs with implicit looops are easy to code, and less prone to errors.

Logical indexing is a way of using implicit loops that we will see in the next slides.

Indexing Example with For Loop:

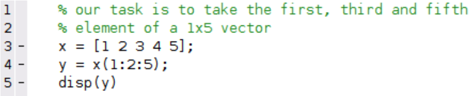


```
% our task is to take the first, third and fifth
% element of a 1x5 vector
x = [1 2 3 4 5];
y = zeros(1, 3);
for i = 1:2:5
    y((i+1)/2) = x(i);
end
disp(y);
```

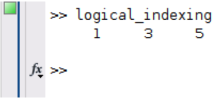
```
>> indexing_with_loop
     1     3     5

>>
```

Different Approach:

Another approach for selecting elements from a vector:



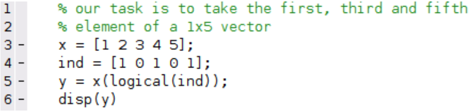
```
% our task is to take the first, third and fifth
% element of a 1x5 vector
x = [1 2 3 4 5];
y = x(1:2:5);
disp(y)
```



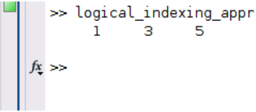
```
>> logical_indexing
     1     3     5

>>
```

Logical Indexing:

Here is another solution with logical indexing:

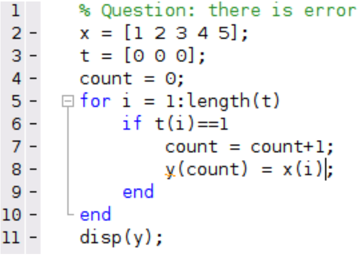


```
% our task is to take the first, third and fifth
% element of a 1x5 vector
x = [1 2 3 4 5];
ind = [1 0 1 0 1];
y = x(logical(ind));
disp(y)
```



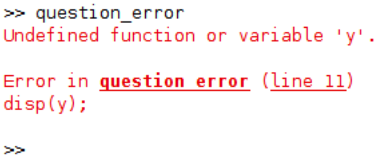
```
>> logical_indexing_appr
     1     3     5

>>
```

Question:



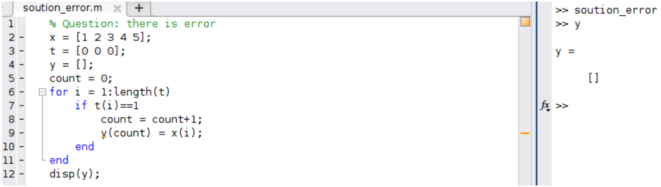
```
% Question: there is error
x = [1 2 3 4 5];
t = [0 0 0];
count = 0;
for i = 1:length(t)
    if t(i)==1
        count = count+1;
        y(count) = x(i);
    end
end
disp(y);
```

Error Message:



```
>> question_error
Undefined function or variable 'y'.

Error in question_error (line 11)
disp(y);

>>
```

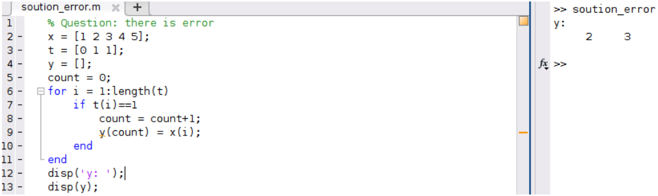
Solution:



```
% Question: there is error
x = [1 2 3 4 5];
t = [0 0 0];
y = [];
count = 0;
for i = 1:length(t)
    if t(i)==1
        count = count+1;
        y(count) = x(i);
    end
end
disp(y);
```

```
>> soution_error
>> y

y =

     []

>>
```

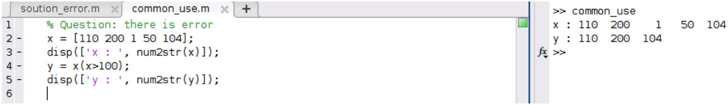
When y is not empty:



```
% Question: there is error
x = [1 2 3 4 5];
t = [0 1 1];
y = [];
count = 0;
for i = 1:length(t)
    if t(i)==1
        count = count+1;
        y(count) = x(i);
    end
end
disp('y: ');
disp(y);
```

```
>> soution_error
y: 
     2     3

>> 
```

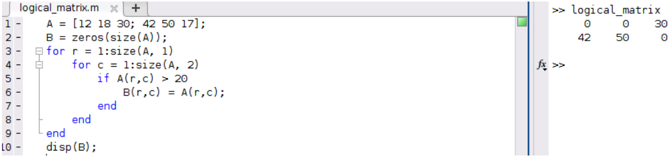
Common Use:



```
% Question: there is error
x = [110 200 1 50 104];
disp(['x : ', num2str(x)]);
y = x(x>100);
disp(['y : ', num2str(y)]);
```

```
>> common_use
x : 110   200     1    50   104
y : 110   200   104
>>
```

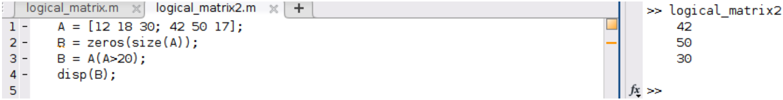
Matrices:



```
A = [12 18 30; 42 50 17];
B = zeros(size(A));
for r = 1:size(A, 1)
    for c = 1:size(A, 2)
        if A(r,c) > 20
            B(r,c) = A(r,c);
        end
    end
end
disp(B);
```

```
>> logical_matrix
     0     0    30
    42    50     0

>>
```

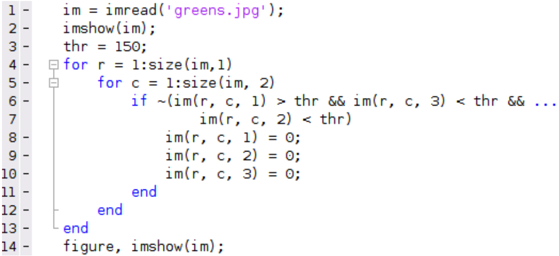
Indexing with Matrices:



```
A = [12 18 30; 42 50 17];
B = zeros(size(A));
B = A(A>20);
disp(B);
```

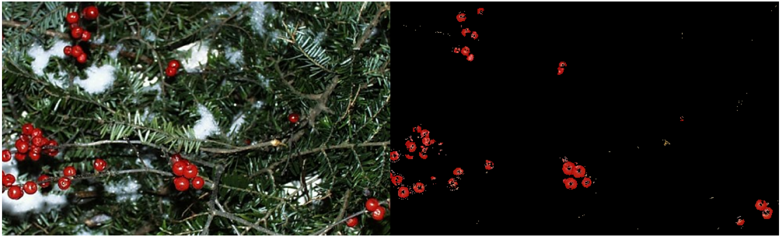
```
>> logical_matrix2
    42
    50
    30

>>
```

repmat Function - 1:

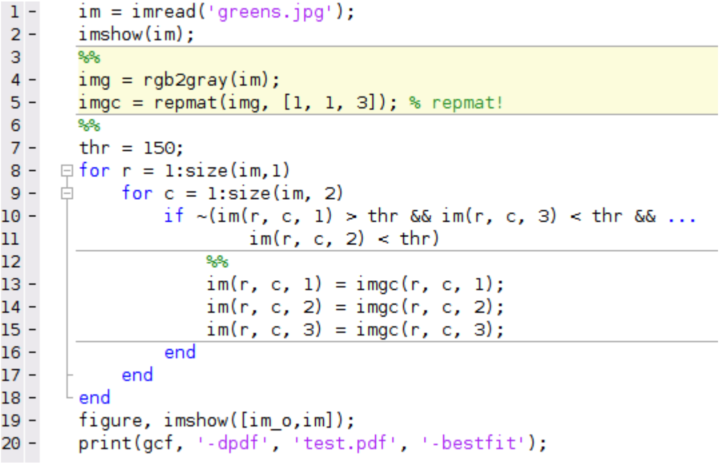


```
im = imread('greens.jpg');
imshow(im);
thr = 150;
for r = 1:size(im,1)
    for c = 1:size(im, 2)
        if ~(im(r, c, 1) > thr && im(r, c, 3) < thr && ...
                im(r, c, 2) < thr)
            im(r, c, 1) = 0;
            im(r, c, 2) = 0;
            im(r, c, 3) = 0;
        end
    end
end
figure, imshow(im);
```

Segmented Fruits:

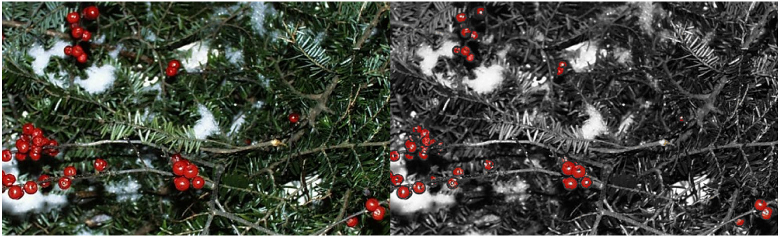


Overlaying the Segmeted Region on Gray Scale:

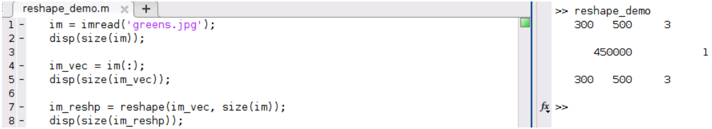


```
im = imread('greens.jpg');
imshow(im);
%%
img = rgb2gray(im);
imgc = repmat(img, [1, 1, 3]); % repmat!
%%
thr = 150;
for r = 1:size(im,1)
    for c = 1:size(im, 2)
        if ~(im(r, c, 1) > thr && im(r, c, 3) < thr && ...
                im(r, c, 2) < thr)
            %%
            im(r, c, 1) = imgc(r, c, 1);
            im(r, c, 2) = imgc(r, c, 2);
            im(r, c, 3) = imgc(r, c, 3);
        end
    end
end
figure, imshow([im_o,im]);
print(gcf, '-dpdf', 'test.pdf', '-bestfit');
```

Overlated Result:



reshape Function:



```
im = imread('greens.jpg');
disp(size(im));

im_vec = im(:);
disp(size(im_vec));

im_reshp = reshape(im_vec, size(im));
disp(size(im_reshp));
```

```
>> reshape_demo
   300   500     3

      450000           1

   300   500     3

>>
```