For Loops 2

ENGR 1181
MATLAB 9
For Loops and Looped Programming in Real Life

As first introduced last lecture, looping within programs has long been a useful tool for completing mundane tasks over and over again. The focus of today’s lesson is fixed number looping although later you will learn about indefinite looping (while looping). Internal combustion engines use a combination of these loops for controlling the spark plugs and valves during a four stage cycle. The engine completes these four stages (for loop) while the car is engine is running indefinitely (while loop, to be learned later).
Today's Learning Objectives

- After today’s class, students will be able to:
  - Use more complex ways of setting the loop index.
  - Construct nested loops in the following situations:
    - For use with two dimensional arrays
    - For repeating a calculation numerous times
  - Use loops to repeat a code with conditional statements.
Review: For Loops

```matlab
for k=1:3:10
    x=k^2
end
```

- Every for must have an end
- Loops can be used in the command window or in script files
- Loops can be nested (loops inside of other loops!)
- if statements can be used inside of loops... and loops can be used inside if statements
Summing with For Loops

- If you use a for loop to calculate a sum, the sum variable must be initialized to zero before the loop.

- MATLAB needs to be reminded to start counting at 0 when it calculates a sum.

- Remember to avoid using MATLAB function names as variable names!

```matlab
sum1=0;
for k=2:2:10
    sum1=sum1+k;
end
disp(sum1)
```

```
sum1= 
    30
```
For Loops With If Statements

- Conditional statements can be used within for loops
- Just like nesting for loops, if statements and for loops can also be nested
- Useful in many applications such as sorting data or displaying statements to the screen
Example

- Given the vector $v=[2 \ 4 \ 6 \ 8]$, use a loop to calculate the square of each element.

- If the square is less than 40, display a message saying: “The square is ..., which is less than 40.”

- If the square is more than 40, display a message saying: “The square is..., which is greater than 40.”
v=[2 4 6 8];
for k=1:4
    vs=v(k)^2;
    if vs<40
        fprintf(‘\nThe square is %i, which is less than 40.’, vs)
    elseif vs>40
        fprintf(‘\nThe square is %i, which is greater than 40.’, vs)
    end
end
Example Program Output

The square is 4, which is less than 40.
The square is 16, which is less than 40.
The square is 36, which is less than 40.
The square is 64, which is greater than 40.
The Break Command

- The ‘break’ command will terminate a current loop.
- MATLAB will jump to the end and carry on with the program.
- Often used with if statements: If a certain condition is satisfied, the loop can be terminated.
- See prep material for an example with ‘break’.
- Type ‘help break’ in command window for more information
The Continue Command

- The ‘continue’ command will stop the remainder of the current loop pass.
- The program continues with the next iteration of the loop.
- Often used with if statements: If a certain condition is satisfied, the pass can be stopped.
- See prep material for an example of ‘continue’.
- Type ‘help continue’ in command window for more information.
Important Takeaways

- For loops and if statements can be used in combination by nesting
- The ‘break’ command has MATLAB jump to the ‘end’ of the loop (breaks the loop)
- The ‘continue’ command has MATLAB jump to the end of the iteration (doesn’t break the loop)
Preview of Next Class

- Program Design
  - What is an algorithm, how are they useful?
  - Flowcharts
  - Top Down Examples
What’s Next?

- Review today’s Quiz #09
- Open the in-class activity from the EEIC website and we will go through it together.
- Then, start working on MAT-09 homework.
- Prepare for next class by reading about While Loops