Topics: User-defined function

Reading (ML): Sec 5.2, 5.4, 5.5, 5.7 (exclude private function)
Optional Reading: Sec 5.3 (optional arguments)

User-Defined Function

- Can easily "reuse" code
- Functions can be independently tested
- Upon invocation, each function has its own memory space inaccessible by other functions or the command window space—variables in a function can be "seen" only inside the function
- Values stored in variables are not preserved between function calls .
- Arguments are "passed by value"

Be sure you understand the example on p. 164 in Chapman.

Example: find prime numbers (again!)

Script file **savePrime.m**:

```
% Save prime numbers in [2,n] to vector prime

n = input('Enter number: ');
prime = 2;  % vector to store prime #s
i = 3;      % next number to be checked
while (i<=n)
  % check number i, save if prime
    prime = [prime isPrime(i)];
  % go to next number
    i = i+1;
end
prime
```

Function file **isPrime.m**:

```
function out = isPrime(n)
% Determine if n is prime, n>=2
% out <-- n if n is prime
% out <-- [] if n is composite

divisor = 2;
while ( mod(n,divisor) ~= 0 )
  divisor = divisor + 1;
end
if ( divisor==n )
  out = n;
else
  out = [];
end
```

Global Memory

- Global memory can be accessed from any workspace
- Global variable must be declared to be global before it is used for the first time in a function.

  global *variable1 variable2 ...*

Persistent Memory

Persistent memory can be accessed from within the function only *and is preserved unchanged between calls to the function.*

persistent *variable1 variable2 ...*

Aside: creating vectors through concatenation

```
% Add vectors a and b
n = length(a);
c = zeros(1,n); % unnecessary statement but improves efficiency
for i = 1:n
   c(i) = a(i) + b(i);
end
```