Topics: User-defined function

Reading (KU): Sec 3.4, 7.1–7.3, 7.5, 7.6

General form of user-defined function

function *[outarg1, outarg2, ...]* = **fname**(*inarg1, inarg2, ...*)

```
% H1 comment line
% Other comment lines
```

executable code

Example: find prime numbers (again!)

Script file **listPrime.m**:

```
% List prime numbers in (2..n)

n= input('Enter number: ');
fprintf('Prime numbers in (2..%d):\n', n);

for num= 2:n
    % Display n if n is prime









end
```

Function file **isPrime.m**. How to write the *function header*?

```
% = n is prime, n>=2
% out <-- 1 if n is prime
% out <-- 0 if n is composite

divisor= 2;
while ( mod(n,divisor) ~= 0 )
  divisor= divisor + 1;
end
if ( divisor==n )


else


end
```

User-Defined Function

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

Subfunctions

- more than one function in an M-file
- top function is normal
- remaining functions are *subfunctions*, accessible only by top function

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 ...*

Note: In Project 2, we used several global variables for convenience in our programs. This indiscriminate use of global variables is *not* good! In general, we want a function to have *local* variables that are visible only within the function—protected from other functions. We used many global variables in Project 2 only because we have not taught data structure yet. In the future we will use better designs!

Persistent Memory

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

persistent *variable1 variable2 ...*