CSS 455 Winter 2012 C. Jackels
Activity No. 2 (Answer Key) January 11, 2012

1. **Variable scope exercise (main function)**
   a. Boot Matlab on your machine
   b. Download the m-file *demo1.m* from the set1 examples on the course web site.
   c. Place *demo1.m* in your working directory and examine it.
   d. Enter the following commands at the command line:

```
>> clc
>> clear
>> x = 5;
>> y = 10;
>> z = 15;
>> disp ([‘x, y, z in main’])
>> [x y z]
>> demo1 (z)
>> [x y z]
```

   e. Considering your output, answer the following questions:
      i. Are arguments in Matlab passed by reference or by value? How can you tell? They appear to be passed by value. The argument of *demo1* is passed as *z* from the main program, and it is incremented by 2 in *demo1*. Upon return, *z* still has the value 15 instead of 17. If called by reference, *z* would have had the value 17 after return.
      ii. Are variables in the main program (command line) available to the function? Do they appear to be local or global?They appear not to be global. The function *demo1* tries to reference a variable *y* and an error results. If the main program variables had been global, *y* would have displayed its value from the main program (10) instead of an error.

Function and variable scope exercise A

   f. Download the m-file *demo2.m* from the set1 examples on the course web site.
   g. Place *demo2.m* in your working directory and examine it. *Demo2.m* contains a subfunction *demo3.m.*
   h. Enter the following commands at the command line:

```
>> clc
>> clear
>> x = 5;
>> y = 10;
>> z = 15;
>> disp ([‘x, y, z in main’])
>> [x y z]
>> demo2 (z)
>> [x y z]
>> demo3(z)
```

   i. Considering your output, answer the following questions:
      i. Are arguments in Matlab passed by reference or by value between functions and their subfunctions? How can you tell? Again, passed by value – same as above. *Demo3* was called from *demo2* with argument *x*

See reverse side

that had been incremented to 17. After incrementing it by 3, the local variable in *demo3* had a value 20. Upon return to *demo2*, the value of *x* was still 17, however. To see this, you had to comment out the attempt to reference the nonlocal variable *x* within *demo3*. That error caused *demo2* to abort as well along with *demo3*. At that point, your conclusion is that variables are passed between functions by value as well.

ii. To which calling programs are subfunctions visible?
Apparently, *demo3* is only visible to *demo2* and not to command line or main program. Subfunctions can only be found by functions within that same file. Matlab is interpreted - there is no compile or link-edit step, so the functions within a file do not appear in an entry point table. The only way the function is found from an external file is by the m-file file name. That is the reason they usually bear the name of the primary function.

iii. Are local variables in the primary function visible to the subfunction?
No. Attempt to access *x* in subfunction *demo3* throws an error.

2. Function and variable scope exercise B

a. Download the m-file *demo4.m* from the set1 examples on the course web site.
b. Place *demo4.m* in your working directory and examine it. *Demo4.m* contains a nested function *demo3.m.* Otherwise it is the same as *demo2.m.*
c. Enter the following commands at the command line:

```
>> clc
>> clear
>> x = 5;
>> y = 10;
>> z = 15;
>> disp ([‘x, y, z in main’])
>> [x y z]
>> demo4 (z)
>> [x y z]
>>
```

d. Considering your output, answer the following questions:
   i. Are arguments in Matlab passed by reference or by value between functions and their nested functions? How can you tell?
   By value. The value of *x* in demo4 after return from *demo3* was still 17 even though the calling argument had been increased to 20 by *demo3. This appears to be a universal convention in Matlab. See the course bulletin board for further discussion.*
   ii. Are the local variables in the primary function visible to the nested function? Yes. We see that the nonlocal variable *x* within *demo3* has the value set in the primary function *demo4.* We also note that the value of *w* set in *demo3* is available to the primary function *demo4.* This is the primary difference between nested and sub functions. The primary function and nested functions share some variable space. See the help pages for more details about sharing space between nested functions at different levels of nesting.

See reverse side