Final Exam

Computer Programming for Engineering and Science
Held on 7th of August, 2016 (CSCE 155N, Summer 2016)

Name :

Save : **YES** **NO**

Course No : **CSCE155N Matlab**

Instructions:

1. This is open book, open note, but not open computer or open neighbor. Please do not use email, texting, Morse code, sign language, spy cameras, etc. during the exam.
2. If you have a question about the meaning of an exercise, ask! Getting things wrong because of misunderstandings can be aggravating for me as well as you.
3. If you wish to have the exam saved so that you can retrieve it later, please indicate this. Exams so marked will be saved until at least the second week of classes in the fall.

1. (10 points) Consider the following snippet of code. How will Matlab respond to each line? Explain the significance of *eval* and of the *input* functions inside the square brackets.

```
>>big = 'bigger';
>>bigger = 'biggest';
>>biggest= 'ultrabiggest';

>>eval('bigger')

>>eval(bigger)

>>eval(eval('bigger'))

>>eval(eval(eval('big')))

>> eval([input('number: ','s') input('operator: ','s') input('number: ','s')])
number: 5*2
operator: +
number: 100
```

2. (10 points) Consider the following functions. What would be the response to the command line **x = nifty(2,4) + nifty(1,2)**?

```
function a = nifty(b,c)
t = c + neato(bummer(c), bummer(b));
a = t + bummer(neato(t,c))
end

function x = neato(y,z)
x = y + z;
end

function x = bummer(y)
x = y * 2;
end
```

3. (10 points)

   (a) Rewrite the following code using *while* instead of *for*.

   (b) What is the final value of *count*?

```
count = 0;
for x = 10:2:20
   disp(x)
   for y = x:-1:0
      count = count + 1;
      disp(y)
   end
end
```

4. (10 points) Consider the following functions. What would be the response to the call **foo(5)**? Note that there is both a returned value and side effects.

```
function x = foo(n)
disp(n)
if n > 0
   x = 3 * bar(n-1);
else
   x = 1;
end

function x = bar(n)
disp(n)
if n > 0
   x = - 2 * foo(n-1);
else
   x = -1;
end
```

5. (10 points) Consider the following function.

   (a) What would be the response to the call **fun** with no arguments?

   (b) Rewrite it (the function and default x) so that it evaluates infix notation expressions that are formed as nested cell arrays.

```
function t = fun(x)
if nargin == 0
   x = {{1,2,'+'} {{3,4,'*'},5,'+'} '*'};
end

if iscell(x)
   switch x{3}
      case '+'
         t = fun(x{1}) + fun(x{2});
      case '*'
         t = fun(x{1}) * fun(x{2});
   end
else
   t = x;
end
```

6. (10 points) Consider the following code that checks to see if an array is sorted. Create anonymous functions for each of the following cases that responds with **true** if the first argument is less than the second. Finally show how to invoke **isSorted** to answer the requests that follow:

   (a) Compare two fractions (each a struct with numerator and denominator) to see if the first is less than the second. The array of fractions to verify is called *fractions*.

   (b) Compare two times according to their occurrence in the day. A time is a struct with three numeric fields *hour*, *min* and *sec*. The array of times to verify is called *times*.

```
function s = isSorted(a, cFun)
   s = true;
   for ii = 2:length(a)
      if ~cFun(a(ii-1), a(ii))
         s = false;
      end
   end
```

7. (10 points) Consider the following code. What is the final value of *count*?

```
count = 0;
for a = 0:2
   for b = 2:-1:a
      for c = a:b
         count = count + 1;
      end
   end
end
```

8. (10 points) Consider the following code. What is printed by the code that follows the function definitions?

```
function q = enqueue(v, q)
   q = [v q]; fprintf('enqueing %d into queue\n', v);
end

function [v q] = dequeue(q)
   s = s(end); fprintf('dequeing %d from queue\n', v);
   q = q(1:end-1);
end

function s = push(v, s)
   s = [v s]; fprintf('pushing %d onto stack\n', v);
end

function [v s] = pop(s)
   v = s(1); fprintf('popping %d from stack\n', v);
   s = s(2:end);
end

s1 = [];
s2 = [5,6,7];
q  = [];

[v s2] = pop(s2);
q = enqueue(v, q)
[v s2] = pop(s2);
q = enqueue(v, q)
[v s2] = pop(s2);
s1 = push(v, s1)
[v q] = dequeue(q);
s1 = push(v, s1)

disp(s1);
disp(s2);
disp(q);
```

9. (10 points) Consider the following code. What will the final contents of each of the files be?

```
f1 = fopen('able','w');
f2 = fopen('baker','w');
for ii = 1:5
   fprintf(f1, '%d\n', ii*2)
   fprintf(f2, '%d\n', ii*2+1)
end
fclose(f1)
fclose(f2)
f1 = fopen('able');
f2 = fopen('baker','a');
f3 = fopen('charlie','w');
for ii = 1:5
   x(ii) = {fgetl(f1)}
   fprintf(f2, '%s\n', x{ii})
end
for ii = 5:-1:1
   fprintf(f3, '%s\n', x{ii})
end
fclose(f1)
fclose(f2)
fclose(f3)
```

10. (10 points) Please describe how you expect to be using programming in future classes and your career. (If you don't have any idea, I would like to know that also!)

Question	Points	Score
1	10	
2	10	
3	10	
4	10	
5	10	
6	10	
7	10	
8	10	
9	10	
10	10	
Total:	100	