Final Exam

Computer Programming for Engineering and Science
(CSCE 150, Fall 2008)

Held on 17/19th December, 2008 *17/19 December, 2008*

Name :
Course No : **CSCE150**

Instructions:

1. There are two parts to this exam. The first part pertains to a program development of the game of Sudoku. The final version is functional but not perfect. You are very welcome to have copies for your entertainment and further practice in programming! The second part is in a traditional exam format.
2. All answers should go on blank paper that will be provided. Very carefully number and draw lines separating each solution!
3. This is open book, open note, open computer, but not open neighbor. There should be lab space available.
4. 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.
5. This exam is offered on two days. You are on your honor not to reveal anything to members coming on Friday or accept anything from members coming on Wednesday.

The first set of exercises refer to the code of the Sudoku program series. Complete at least half of the questions on each version of the program (ceil(n/2) where n is the number of questions).

1. (Sudoku 1) What would a typical run of this version display on the screen?

2. (Sudoku 1) What kind of a variable is *another*?

3. (Sudoku 1) Is the game really set up or played? Explain.

4. (Sudoku 1) Some of the functions are incomplete ’stubs’. What purpose is server by having stubs?

5. (Sudoku 1) What would likely happen if the nearly top line *another = true;* were missing? Why?

6. (Sudoku 2) In the function *setUpGame()* an array is preassigned. What array is this and what is a common reason for doing this?

7. (Sudoku 2) In the eventual production version of Sudoku, the ’Default Board’ would likely not be an option. Why is it an option in this early version?

8. (Sudoku 3) There is no error checking or provision of correcting user input while getting a Custom Board. Why not?

9. (Sudoku 3) What is happening in the lines *mask = board == ’.’* and *board(mask) = ’ ’*?

10. (Sudoku 3) Why are the expected semicolons omitted from the lines referred to in the previous exercise?

11. (Sudoku 3) In the production version one might expect another way of getting custom Sudoku games loaded into the program. What would that method be?

12. (Sudoku 4) Displaying a menu, accepting a choice from the user, and using a switch statement to direct control to the appropriate module is very common. How could this be adapted to allowing either upper or lower case letters as (equivalent) valid responses?

13. (Sudoku 5) Sketch the appearance of a typical board as displayed by the function *showBoard().*

14. (Sudoku 6) There are three stub functions added along with *solved()*. What would you expect each of them to do when they are eventually fleshed out? (What do you suppose I mean by *quad*?)

15. (Sudoku 7) (I really did this is three stages, checking each before going on.) What does the variable *n* ultimately look like for *allColumnsDone()* and *allRowsDone()*?

16. (Sudoku 7) The number 4293 represents the sum of the ASCII values of the digits in a completed Sudoku board. Is it possible there is a mistake here, that what we really want is a total of 81? Explain.

17. (Sudoku 7) Why is each row and column sorted? Hint: What is the effect of $c = a == n$?

18. (Sudoku 7) In the function *allQuadsDone()* how is it legal to refer to *q(1:9)* when q is an array with 3 rows and 3 columns?

19. (Sudoku 8) Why is *int2str(guess(3))* needed in the function *makeGuess()*?

20. (Sudoku 8) if the board with the new play is not valid, what happens?

21. (Sudoku 9) It seems that checking for a valid board is not a trivial exercise since this is taking a couple interations! What would be the advantage of the line in function *validBoard()* that sets *quad* to a ceiling of a strange looking expression? (What would *quad* really be?)

22. (Sudoku 10) Wow, another iteration to determine a valid board! In plain English, what does determine a valid row, column, and quad?

23. (Sudoku 10) What do you suppose is really the intent of the stub function *valid9()*?

24. (Sudoku 10) What is the meaning of the gobbledygook setting a row and col in the function *validQuad()*?

25. (Sudoku 11) What are valid characters in the varible *validChars* of function *valid9()*?

26. (Sudoku 11) Note that the input string r of function *valid9()* is sorted. How does the line computing *dups* correctly detect the existance of duplicate characters in r?

27. (Sudoku 11) In plain English, what does it mean for the (up to) 9 digits (of a row, col, or quad) to be valid? Note that this is what is calculated in the last line of the function *valid9()*!

28. (Sudoku 12) In this iteration we are preparing to put in some real reasoning power! Why do I have nested while loops instead of nested for loops in the function *getHint()*?

29. (Sudoku 13) Note that function *goodSpot()* does very little work and the real reasoning happens inside function *checkRowColQuad()*. So what is the big advantage of having two functions here instead of just one?

30. (Sudoku 13) What is the function *checkRowColQuad()* really checking for? Hint: What is the effect in the assignment to *hint* of checking the lengths of q, r, or c to be 8?

31. (Sudoku 14) Even more brains are put into the program! We now have two kinds of hints, one from the previous interation, and a new one in the form of *hint2*. What is the meaning of this new hint?

This part of the exam may closely resemble the midterm and quizzes. If you have your old midterm and have studied from it, you may certainly use it and have no problem with this part of the exam. But note - there may be slight changes, so read carefully!

32. What is output by the following fragment of code?

```
a = [1:5; 6:10; 11:15];
for ii = a
    fprintf(’%d %d %d\n’, ii);
end
```

33. What is output by the following fragment of code? Would growth be linear or quadratic?

```
for ii = 1:5
   for jj = 1:ii
      fprintf(’up ’);
   end
   for jj = ii:5
      fprintf(’down’);
   end
   fprintf(’\n’);
end
```

34. What is output by the following fragment of code? Would growth be linear or quadratic?

```
t = 9;
for ii = 0:t
  for jj = 0:t
    fprintf(’%4d ’, ii*jj);
  end
  fprintf(’\n’);
end
```

35. Matlab is an interpreted language rather than compiled, although there is a jit (just-in-time) facility to provide one of the big advantages of compiled langauges. Check all the statements that are consequences of this (warning - don’t just check the true statements!):

    (a) Matlab tends to run slower.

    (b) Each line is translated on-the-fly into something the machine can run.

    (c) Matlab can be used like an interactive calculator.

    (d) Matlab works with matrices.

36. There are three types of files associated with Matlab. Mark as correct the following statements or reword (don’t simply cross them off!) to make them correct.

    (a) .m files are used to hold the programs themselves. They can be created and edited using either the Matlab’s edit window or any wordprocessor.

(b) Data files are used to hold the data generated by or to be used by a program. The commands in Matlab used to access these files are ‘input’ and ‘fprintf’.

(c) Diary files are used to keep a comprehensive log of activity while Matlab is running. Once they are created, they can be used to automatically rerun a sequence of steps.

37. Data items in Matlab are held in the computer’s memory which is organized as a linear series of cells. Answer the following questions about memory assuming a has been assigned the value $[15.5, 29.2, 51.7; 61.3, 22.1, 86.9]$:

(a) Assuming each piece of data in an array consumes one unit of memory (not really true), and the 15.5 is stored at location 500, then where in memory would the 29.2 be stored?

(b) Data are assigned into an array (as in $arr = [3, 5, 2; 5, 2, 3]$) in (column or row) major form and stored in memory in (column or row) major form.

(c) Complex real numbers are typically stored in four parts rather than just one. What would be the four parts? Would each likely be one byte in size?

(d) Strings of characters (eg. $'howdy'$) could be stored either directly (as in $str =' howdy'$) or as an array (as in $str = ['h','o','w','d','y']$). True or false? Are these treated the same in Matlab?

38. Check all of the following that are true about short circuits, and correct those that are not true:

(a) Short circuiting an AND expression means that the right hand side is not evaluated if the left hand side evaluates to false.

(b) A short circuit occurs when lines from either side of a voltage source touch each other.

(c) Short circuiting an OR expression means that the right hand side is not evaluated if the left hand side evaluates to false.

(d) Short circuiting in Matlab can only be done with scalar values.

39. Manipulating matrices is a forte of Matlab. How can one do the following using a single assignment statement?

(a) Make an array consisting of the odd numbers from 1 through 1000.

(b) Update only the first column of an array to all ones.

(c) Update only the lowest rightmost corner cell to one.

(d) Zero out all but the diagonal (starting at cell 1,1) of a 10 by 10 matrix.

(e) Zero out all but the diagonal (starting at cell 1,10) of a 10 by 10 matrix.

(f) Zero out only the diagonal (starting at cell 1,1) of a 10 by 10 matrix.

(g) Do a matrix multiply of arrays A and B.

(h) Assign the (integer) quotient and remainder of a/b to a two-element array x.