UCR Department of Computer Science & Engineering

CS 30 Take-Home Final Examination -- June 2009

Due: 11:59pm Monday June 8, 2009

NOTE: All questions are NOT of equal difficulty, and I don't expect all students to be able to answer every question perfectly. This is intentional, so that I have some basis to distinguish between A's, B's and C's.

1. Briefly explain what "lazy evaluation" means in the context of logical expressions, and thus the significance of having two similar operators, such as "&" versus "&&". Give an example where switching between these two operators gives a different result.

2. One day, one of your existing, well-tested Matlab scripts suddenly starts producing completely unexpected and nonsensical results. Therefore, you suspect that some Matlab predefined value (such as "pi") has been accidentally changed. How would you go about finding whether this is really the cause of your problems? If so, how would you fix it?

3. Suppose you wish to store the data gathered from 10 replications of laboratory experiment in Matlab. For each replication, the data includes a numeric value (the max temperature reading, say), a character string (the color of the reaction products) and a time series (the digitized sound of test tubes shattering when you try cooling them too quickly).

    a. One way to store the data is to create three separate arrays called `temperature`, `color`, `sound`.
        i. Show the Matlab statement to set the `temperature` for replication 4 to 350.
        ii. Briefly explain how Matlab represents strings as an array of characters. How do you then create an array of strings? For example, suppose the color for replication 7 was ′purple′; how do you add that value to the array `color`?
        iii. What problem occurs if some of the colors or time series have different lengths? How can it be solved?
    b. Alternatively, all the data could be combined into one structure array, with one numerically-indexed element per experiment, and three named fields per element.
        i. Show the Matlab statement to set the `temperature` for replication 4 to 350.
        ii. Show the Matlab statement to set `color` for replication 7 to ′purple′.
        iii. Briefly explain why this method of organizing the data does NOT have the same problem with variable-length strings or time series.

4. The following sample data about MS degrees awarded by the CS Department from 1986 to 2007 is available at http://www.cs.ucr.edu/~mart/30/CS_MS.txt

   17 13 27 25 22 13 16 15 7 12 11 12 9 10 7 6 10 6 11 24 9 20

   a. Give the sequence of Matlab commands for plotting this data in as a two-dimensional graph where the x-axis is 1 through 22 (representing years since the start of the MS program) and the y-axis is the degrees awarded in that year. Be sure to label the axes and provide a title for the figure. Save the result as figure3a.fig and submit it with your exam.

   b. Now suppose we wish to find the "best" polynomial approximation to this data set in the sense that it **minimizes the maximum error**, when calculated in the following way:

      i. Let **`PN`** be the degree-N polynomial approximation to this data set produced by the built-in Matlab function `polyfit` (described in section 20.8).
      ii. Let **`CS`** be the cubic spline approximation to the same data set produced by the built-in Matlab function `spline` (described in section 21.2).
      iii. Let **`test_X = 1 : 0.1 : 22`** be a set of equally spaced test points over the spaces between the original data samples.

      Evaluate both **`PN`**, using the built-in Matlab function `polyval`, and **`CS`**, using the built-in Matlab function `ppval`, at every test point and remember the largest difference for this polynomial degree N. Repeat for all degrees between 1 and 22, and plot the maximum error as a function of N in figure3b.fig and submit it with your exam.

   c. If your solution to part (b) used a loop to search the two arrays of y-values for the pair with the largest difference, explain how to find the largest difference with an array operation and no loops. Also show how this array operation method could easily be modified to find the average (not max) of the absolute value of difference.

5. This question builds on the lecture material about how to write an m-file function **swap(a,b)** that interchanges the values of the two variables given as its input parameters.
   a. Briefly explain the difference between a *script mfile* (described in Chapter 4 of the textbook) and an *mfile function* (described in Chapter 12) with respect to the use of parameters and access to workspace variables.

   b. More specifically, suppose you want to use the Matlab statement:

      ```
      numVars = size(whos,1)
      ```

to set the variable numVars to the total number of variables currently available in this workspace. How does the effect of executing that Matlab statement change if you:

i. type it directly into the Matlab command window,
ii. place it inside a *script mfile* that you call from the command window, or
iii. place it inside an *mfile function* that you call from the command window?

c. Write an *mfile function* called **`countargs.m`** that returns the number of input parameters in the environment of its caller. For example, if you type the following

```
silly_func(1, 2, 13)
```

in the Matlab command window and the statement

```
x=countargs
```

appears inside the mfile function called **`silly_func.m`**, then the value of `x` should be set to 3.

d. Write an *mfile function* called **`showargs.m`** that returns a structure array giving the name, size and class for all the input parameters in the environment of its caller. For example, if you type the following

```
x=1; y=2:4; z='happy'; silly_func(x, y, z)
```

in the Matlab command window and the statement

```
x=showargs
```

appears inside the mfile function called **`silly_func.m`**, then the value of `x` should be a 3-element structure array containing the following data:

x(1).name: 'x'	x(1).size: [1 1]	x(1).class: 'double'
x(2).name: 'y'	x(2).size: [1 2]	x(2).class: 'double'
x(3).name: 'z'	x(3).size: [1 5]	x(3).class: 'char'