UCR Department of Computer Science & Engineering

CS 30 Final Examination -- June 2010

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. State whether each of the following is true or false. (2 points for each correct answer, -1 point for each incorrect answer; 0 for blank)

   a. In the Matlab command `plot(x, y, c)`, the parameters `x` and `y` can be one-dimensional vectors of numbers with the same number of elements, and `c` can be a string with information about how to format the resulting line.

   b. The Matlab function `polyfit(x,y,n)` generates a smooth curve that always pass through every point in the data set defined by `x` and `y` unless `n` is negative.

   c. The Matlab `spline` function generates a smooth curve that passes through every data point in a sequence.

   d. If `A` and `B` are two-dimensional matrices, then matrix multiplication `A*B` is not allowed unless `A` and `B` have exactly the same number of rows and columns.

   e. If `data='This is a test'` then the Matlab command sequence `b=findstr(data,' '); data(b(2):b(3))` will produce the answer ‘is’.

   f. When plotting graphs, the command `hold on` means that the next command will add to, rather than replace, the most recently generated graph.

   g. In Matlab, you can use `pi` as one of your own variable names, but not `if`.

   h. The Matlab command `M(2,:)=[]` deletes the second row from a two dimensional matrix.

   i. If `A=[1 2 3; 4 5 6; 7 8 9]` is a two-dimensional matrix, then `A(8)` and `A(2,3)` both refer to the same element, which has value 6.

   j. In Matlab, the operators `=` and `==` are related in the same way as `&` and `&&`. In other words, `A==B` means assign the value of `B` to `A` if they are different, and do nothing otherwise.
2. In this question, you will be completing the following Matlab function

```matlab
function FM = Flat_Spot (M, r, c)
```

with three input parameters: two dimensional array `M`, row `r` and column `c`, and one output parameter `FM` that is identical to `M` except the four elements `M(r,c)`, `M(r+1,c)`, `M(r,c+1)` and `M(r+1,c+1)` have been replaced by their average.

a. Briefly explain the meaning of “call by value” parameter passing in Matlab. If your function makes changes to an input parameter (say `M`), will those changes have any effect on its value in the workspace of the caller?

b. Write some Matlab code that can be placed inside your function `Flat_Spot` to validate its input parameters. There must be exactly 3 input parameters. `M` must be an two-dimensional array with at least two rows and two columns, `r` and `c` must be positive integers which are less than the number of rows and columns, respectively.

c. Write the remainder of the function `Flat_Spot`, which copies `M` to `FM`, calculates the average of the four elements, and assigns it to those elements.
3. Question 5 on the 2009 CS 30 Final Exam was about writing an *mfile function* called `countargs` 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

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

in the Matlab command window and the statement

```matlab
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:

<table>
<thead>
<tr>
<th><code>x(1).name:</code></th>
<th><code>x(1).size:</code></th>
<th><code>x(1).class:</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>'x'</td>
<td>[1 1]</td>
<td>'double'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><code>x(2).name:</code></th>
<th><code>x(2).size:</code></th>
<th><code>x(2).class:</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>'y'</td>
<td>[1 2]</td>
<td>'double'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><code>x(3).name:</code></th>
<th><code>x(3).size:</code></th>
<th><code>x(3).class:</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>'z'</td>
<td>[1 5]</td>
<td>'char'</td>
</tr>
</tbody>
</table>

Most of the difficulty in the *previous exam question* was about how to access the environment of the caller from inside the `countargs` function. Therefore, in *this exam question*, I want you to write a *script mfile* called `countargs.m` which does the same thing. Since a *script mfile* cannot have any input or output parameters, your code should simply store its answer in the variable called `argstuff`. Continuing with the previous example, if the statement

```matlab
showargs
```

(without an assignment) appears inside the mfile function `silly_func.m`, then the value of `argstuff` should be a 3-element structure array containing the data shown above. [**HINT:** Note that a *script mfile* runs in the caller’s environment, so your script can use `x=0`, rather than `assignin('caller','x=0')`, to assign zero to a variable `x` from the workspace of `silly_func`.]