

Homework 1

Due: 2:30 PM Feb. 14, 2008

To submit your code electronically, make sure that in your `home` directory you have a `course` directory and that in that there is a `cs004` directory. You should save all your programs into `.m` files. For this assignment, you will place all your code into the directory

```
/u/<login>/course/cs004/hw1
```

(*Change <login> to be your login name.*) After you have your code saved in the proper directory you can run the handin script. Open a terminal, then type

```
cs004_handin hw1
```

For this homework, put all of your `Matlab` code into one `m`-file, with each problem separated by comments. Name your file `hw1`.

The following commands will be useful for completing the homework. You can learn more about them by entering `help x` into the `MATLAB` command window, where `x` is the command.

```
zeros, ones, eye, max, reshape
```

Problem 1.1

Here are some more useful exercises to make working with `Matlab` easier:

DO these things, and then state that you did them in a comment in your `.m` file. Stating that you did them gets you full credit. (Stating it falsely should give you a bad feeling, and would be stupid; these things are here to help you.)

Task 1: Open up one of your old programs and highlight a block of text, then press `Ctrl+R` to comment out a whole portion of your code. `Ctrl+T` will uncomment it.

Task 2: Next, highlight all your code and type `Ctrl+I` to smart indent it. This can be very useful to keep your code organized, especially in later programs with multiple loops - keeping your indenting logical can actually

help you debug your code, by helping you visually keep track of where certain loops and functions begin and end.

Task 3: Run your code, either by clicking on the Run icon (a sheet of paper with a downward pointing arrow next to it), or by pressing F5.

Task 4: Click on one of the the little dashes that appear to the right of the line numbers. This will turn it into a red dot, which will be used as a *breakpoint*. When execution reaches a breakpoint, it stops and **Matlab** allows you to examine the variables and output of your program. Run your program, then examine the values of some of your variables by typing the names of the variables into the command window. When you want to program to finish evaluation (or continue onto the next breakpoint) click the **continue** icon (where the run icon is while editing) or press F5.

Task 5: Go to some point near the middle of your code, and on a new line type `asdf`; Run your program, and you should get an error saying ‘‘Undefined function or variable ‘asdf’.’’ Click on the line number that shows up underneath the error message. When you run code with a bug in it (this will happen often!), **Matlab** will give you an error and a line number. Clicking on the line number in the command window will automatically take you to corresponding line in the editor and place the cursor at the beginning so you can fix the bug.

Task 6: If you hit **Tab** after typing the first few letters of a function name, **Matlab** will attempt to complete the rest of the name automatically. It will display a list of all the functions that begin with those letters and allow you to scroll through them to select the one you want. Try it out for yourself. Type `accum` into the command window and hit **Tab**. Notice how **Matlab** replaces it with `accumarray`. Next, type `imag` into the command window and hit **Tab**. Use the dropdown menu that appears to select `image`.

Task 7: Read through the **Matlab** reference guide. It has a bunch of useful tips and descriptions of the most commonly used **Matlab** commands. The reference guide can be found online at the course website in the **handouts** sections.

Problem 1.2

Create three vectors:

$$a = [2 \quad -1 \quad 0 \quad 6]$$
$$b = [-5 \quad 20 \quad 12 \quad -3]$$

$$c = [10 \ 7 \ -2 \ 1]$$

- Use the three vectors in a MATLAB command to create a 3 x 4 matrix in which the rows are the vectors a , b and c .
- Use the three vectors in a MATLAB command to create a 4 x 3 matrix in which the columns are the vectors a , b and c .

Problem 1.3

Create the following matrix A :

$$A = \begin{bmatrix} 0.1 & 0.2 & 0.3 & 0.4 & 0.5 & 0.6 & 0.7 \\ 14 & 12 & 10 & 8 & 6 & 4 & 2 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 \end{bmatrix}$$

- Create a 3 x 4 matrix B from the 1st, 2nd, and 3rd rows, and the 1st through the 4th columns of the matrix A .
- Create a 2 x 7 matrix C from the 2nd, and 3rd rows, and all the columns of the matrix A .

Problem 1.4

Create a vector $v = [1, 2, 3, \dots, 35]$ with 35 elements. Then, use the `reshape` function to create a 5 x 7 matrix in which the first row is the numbers 1 2 3 4 5 6 7, the second row is the numbers 8 9 10 11 12 13 14, the third row is the numbers 15 through 21, and so on.

Problem 1.5

Create a 3 x 3 matrix A in which all the elements are 1. Then use A to create a new matrix B , such that B will become:

$$B = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Problem 1.6

Task 1 Read in the file `bushels.csv` and save the resulting matrix in `bushels` by typing

```
bushels = csvread('/course/cs004/pub/hw1/bushels.csv')
```

(assuming you are in the Sun Lab - if you want to work from home, make sure to download `bushels.csv` from the website and replace `'/course/cs004/pub/hw1/bushels.csv'` with the local path to the file). MATLAB will read in the file, which is essentially a table giving production (in bushels) of ten different vegetables in the years 1999-2004.

Recall that the columns of `bushels` represent the years 1999-2004 and the rows represent the following vegetables:

Cabbage
Tomatoes
Watermelon
Carrots
Corn
Lettuce
Cucumbers
Broccoli
Cauliflower
Peppers

Recall the way in which particular elements of a matrix can be extracted in MATLAB by reviewing your lecture notes and/or reading the MATLAB documentation on “Accessing Single Elements” and “Accessing Multiple Elements” under the “Matrix Indexing” section.

Task 2 Review the documentation for `max` and write a Matlab expression to find the following production values:

- Cucumber in 2001
- Watermelon in 2004
- Tomatoes, cabbage, and corn in 2000
- Broccoli from 2000 to 2003
- Lettuce, cauliflower, carrots, and peppers in odd-numbered years
- The best years for producing tomatoes, broccoli, and corn (you can have year be denoted by the column index)
- The vegetable with the highest production in each year (this should return a row index)
- The year of highest production for each vegetable (should return a column index)

Put a comment after each part of your code, saying which production value it is used to find.

Task 3a Make a vector named `years` containing the years for the data.

Task 3b Use this `years` vector to report the actual best *year* for each vegetable (not its index in the `bushels` matrix).