Math 230 — Final Exam

December 22, 2015

General Directions. This is an open-book, open-notes, open-computer test. However, you may not communicate with any person, except me, during the test. You have 3 hours in which to do the test. Put your answer to each question in the space provided (use the backs of pages if you need more space). Be sure to **show your work**! I give partial credit for incorrect answers if you show correct steps leading up to them; conversely, I do not give full credit even for correct answers if it is not clear that you understand where those answers come from. Good luck.

This test contains 9 questions on 10 pages.

Question 1 (10 Points). Imagine a simple word processor (written in Matlab) that uses a 1-dimensional cell array of strings to represent a document. Specifically, each paragraph of the document is represented by a string in the cell array; the first element of the cell array is the first paragraph, the second element is the second paragraph, and so forth.

Write a group of Matlab statements that prints such a document with blank lines between the paragraphs. In other words, your code should print the first paragraph, then print a blank line, then print the second paragraph, print a blank line, etc. Do not print a blank line before the first paragraph or after the last. Assume that the document is in a variable named `doc`, which contains a cell array in the form described above. Do not initialize this variable. You don't need to do any formatting of individual paragraphs, just print them as strings with `fprintf`, `disp`, or any other way of printing Matlab data.

Question 2 (15 Points). Below are the first and last lines of a Matlab function `removeDigits` that removes all digit characters from a string. More precisely, `removeDigits( s )`, when given a string *s*, returns a string containing all the characters of *s*, in their original order, except for digit characters. For example, `removeDigits( 'de12mo' )` would return the string 'demo'. Fill in the body of `removeDigits`.

```
function [ result ] = removeDigits( string )









end
```

Question 3 (10 Points). The k-norm of a vector $\mathbf{v} = \langle v_1, v_2, \ldots, v_n \rangle$ is defined to be the k^{th} root of the sum of the k^{th} powers of the absolute values of the elements of **v**. In other words, the k-norm of **v** is

$$\sqrt[k]{\sum_{i=1}^{n} |v_i|^k}$$

Write a pseudocode (not Matlab code!) algorithm for calculating the k-norm of a vector **v**. The inputs to your algorithm should be k and **v**, and the output should be the k-norm.

Question 4 (15 Points). Here is a way to use a row vector to store the digits that spell out a decimal fraction: let the first element of the vector be the digit immediately to the right of the decimal point, the second element be the digit to the right of that position, and so forth. All elements of the vector will be integers between 0 and 9. For example, the decimal number 0.123 would be represented by the vector [1, 2, 3]. Conversely, the vector [0, 9, 7, 5] represents the decimal number 0.0975, and so forth.

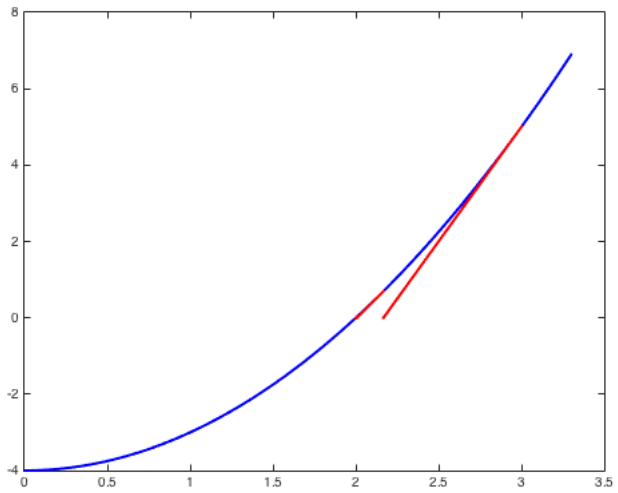
Fill in the following Matlab function definition to create a function that calculates the number represented by vector `digits`.

Do not use any loops in your code; use functions and operations on vectors instead!

Note that the argument to this function (`digits`) is a vector of integers between 0 and 9, while the result (`n`) is a single real number between 0 and 1.

```
function [ n ] = toDecimal( digits )









end
```

Question 5 (20 Points). The next page contains a Matlab function that uses Newton's method to find a root of a function *f*, given *f*, its derivative, and an initial guess at the root. Add statements to the Matlab function so that in addition to finding the root, it also plots the function and the tangent lines used by Newton's method, as in the following example (which shows Newton's method finding a root of the function $f(x) = x^2 - 4$, from an initial guess of 3).



Your code (and the example above) should plot the function in blue, with x values ranging from 0 to 1.1 times the initial guess; it should plot each tangent line used by Newton's method in red. The tangent line used to update estimated root x_i to new estimate x_{i+1} should run from $(x_i, f(x_i))$ to $(x_{i+1}, 0)$.

```
function [ z ] = newton( f, fPrime, x0 )

TOLERANCE = 0.001;

z = x0;

relativeDelta = TOLERANCE * 2;

while relativeDelta > TOLERANCE

    delta = f(z) / fPrime( z );

    relativeDelta = abs( delta / z );

    z = z - delta;

end;

end
```

Question 6 (10 Points). I remember high school teachers teaching a stylized way of writing proofs, in which every proof was written as two columns of text, the left-hand column containing mathematical statements, and the right-hand column containing reasons why those statements followed from previous ones and/or previously proven theorems. Rows in the proof were generally numbered, for easy reference in other rows. For example, a proof in this style of the rule $(x+y)(x-y) = (x^2-y^2)$ might look like

	Statement	**Reason**
1	$(x+y)(x-y)$	Given
2	$= x^2 - xy + xy - y^2$	FOIL, line 1
3	$= x^2 - y^2$	Cancellation, line 2

Suppose you wanted to represent such proofs in a Matlab program. Assuming the statements and reasons are strings, explain how you could use a cell array to represent the overall proof. Your explanation can be a combination of English prose, diagrams, and/or examples.

Question 7 (15 Points). Suppose you are given a vector which you know contains a repeating pattern of numbers, and you want to know how long the pattern is. For example, in the vector [3 2 5 3 2 5] the pattern is 3 numbers long (it's 3 2 5]; in the vector [1 6 12 2 7 1 6 12 2 7 1 6 12 2 7] the pattern is of length 5 (1 6 12 2 7). Assume that no number ever repeats inside the pattern. Write a body for the following Matlab function so that it finds the length of the pattern in vector **v**.

```
function [ len ] = patternLength( v )









end
```

Question 8 (15 Points). Write a series of Matlab statements that solves equations of the form $ax^2 + bx + c = 0$, where any of a, b, or c may be 0. Specifically, your code should print one real root of the equation, or print a message explaining that there are no real roots. Assume that Matlab variables `a`, `b`, and `c` are defined before your code runs (do not include statements to give them values), and process them according to the following rules:

- If all three of a, b, and c are 0, print "0 always equals 0"
- If a and b are 0 but c is not, print the value of c followed by the words "never equals 0"
- If a is 0 but b isn't, print the value of $-c/b$.
- If a is not zero but $4ac > b^2$, print "quadratic with complex roots"
- In all other cases, print the value of $(-b + \sqrt{(b^2-4ac)})/(2a)$.

Question 9 (10 Points). Procopius the Programmer wants a Matlab function that will extract a substring from a string and then reverse the substring. For example, if Procopius wanted to see the 4th through 5th characters of the string 'Hello' in reverse order he would call this function as

```
reverseSubstring( 'Hello', 4, 5 )
```

(The result would be the string 'ol'.) Similarly, to get the 1st through 3rd characters of 'substring' in reverse order ('bus'), Procopius would call

```
reverseSubstring( 'substring', 1, 3 )
```

A Matlab statement that extracts and reverses the substring of `string` between character positions `first` and `last`, putting the result in variable `sub`, could be:

```
sub = string( last : -1 : first );
```

Add Matlab code before and/or after this statement to make a complete function definition for the function Procopius needs. Your function should be callable as shown in the examples above, and, if called as in those examples, should produce the results given in the examples.

Happy Holidays!