Math 230 — Final Exam

May 6, 2016

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 2 ½ 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 7 questions (one with 2 parts) on 8 pages.

Question 1 (15 Points). The Math 230 Squiggle function, $S(x)$, is defined piecewise as follows:

$$S(x) = \begin{cases} -(x+\pi) \text{ if } x < -\pi \\ \sin x \text{ if } -\pi \le x \le \pi \\ -(x-\pi) \text{ if } x > \pi \end{cases}$$

Write a Matlab function, `squiggle`, that takes the value of x as its only argument and that returns the value of $S(x)$. Your code should be based on the above equation, and should not simply assume that S is already defined somewhere in Matlab.

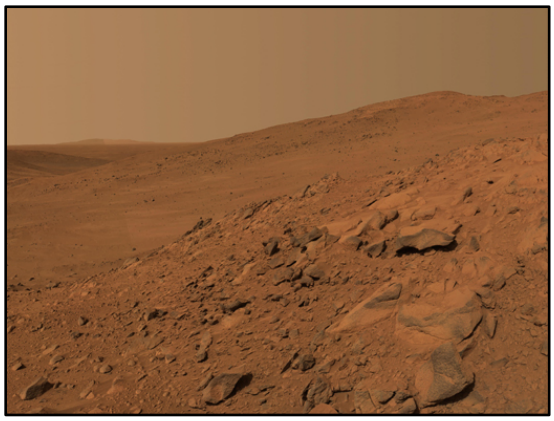
Question 2 (15 Points). Netflix Jr., a hypothetical streaming video service, uses a Matlab cell array to keep track of their movies. Each row in the cell array represents a different movie, and the columns represent the title, genre, and length, respectively. Title and genre are strings, while length is an integer representing the duration of the movie in minutes. For example, the first few rows in the cell array might be…

Gone Girl	Drama	149
Interstellar	Science Fiction	169
Bridge of Spies	Drama	142

The entire movie array is, of course, much longer than this.

Assuming that the movie cell array is stored in a variable named `movies`, write Matlab code that calculates and displays the total running time of all the dramas (genre 'Drama') in the cell array. (For example, for the movies shown above this total would be 149 + 142 = 291.)

Question 3. Suppose you have 2 digital images, *Image1* and *Image2*, for example…



Assume both images are the same size. You want to create a new image whose left half is the left half of *Image1*, and whose right half is the right half of *Image2*, like this:

Part A (15 Points). Here is a skeleton of a Matlab function to combine *Image1* and *Image2* as described:

```
function [ newImage ] = combineVectorized( image1, image2 )









end
```

Fill in the body of this function so that it returns the combined image. Do not use any loop statements in your function.

(Part B on next page)

Question 3, Part B (15 Points). Here is a skeleton for another Matlab function to combine the images:

```
function [ newImage ] = combineLooping( image1, image2 )

   % Initialize an empty result image
   newImage = uint8( zeros( size(image1) ) );









end
```

Finish the body of this function so that it fills `newImage` with the combined image. You may use loops in this function, but do not use any vectorized operations (i.e., statements or operations that access or operate on more than one element of an array at a time).

Question 4 (15 Points). Imagine that you want to represent an appointment calendar for a single day as a Matlab variable. Specifically, the calendar needs to be able to store multiple appointments, each of which has a description (short but arbitrary text, e.g., "meet with Jane"), a starting time (e.g., 9:30) and an ending time (e.g., 10:00). Describe how you would represent the appointment calendar in terms of the kinds of data and variables Matlab can work with. Specifically, explain how you would use Matlab classes or variables to represent descriptions and times, to group descriptions and times together into appointments, and to group appointments together to form the complete calendar.

Question 5 (15 Points). The position of a car driving on an elliptical race track is given by the equations

$$x(t) = 20\cos t$$
$$y(t) = 12\sin t$$

Assuming that there is a function `drawCar(x,y)` that draws a car at point (x,y) and that has no return value, write Matlab statements that draw an animation of the car making one circuit around the track. You should call the `drawCar` function, but do not need to write it. You should include statements that draw the track.

Question 6 (15 Points). A hypothetical programming language uses the pound sign (#) to introduce comments (analogous to how Matlab uses the % character). For example, in this language the line

```
x = 0       # start counting from 0
```

tells the computer to store 0 in a variable named `x`, but the computer pays no attention to the text from the # to the end of the line.

Here is the skeleton of the Matlab function that is supposed to remove comments from lines in this hypothetical language. The function's argument, `line`, is a string that contains the text of a line from a program. The result, `command`, should be the first part of that line, up to the character just before the first #, if any. If there is no # in `line`, then the result should be equal to `line`. If `line` begins with a #, the result should be an empty string. Fill in the body of this function.

```
function [ command ] = decomment( line )









end
```

Question 7 (15 Points). An alternating infinite series is a sum of the form

$$S = \sum_{i=1}^{\infty} (-1)^{i+1} u_i$$

For example,

$$S = 3 - \frac{3}{2} + \frac{3}{3} - \frac{3}{4} + \cdots = \sum_{i=1}^{\infty} (-1)^{i+1} \frac{3}{i}$$

is an alternating infinite series that I will call the *alternating Math 230 harmonic series*. (It's made up for this exam, but inspired by a real series called the alternating harmonic series.)

Computer programs that need to estimate the value of any infinite series can do so by adding up the first *n* terms of the series, for some finite *n*. One convenient fact about alternating series is that the error in estimating an alternating infinite series this way (i.e., the difference between the estimated value and the actual value if the sum were carried out for infinitely many terms) is at most the absolute value of the $(n+1)^{\text{th}}$ term, i.e., $|u_{n+1}|$.

Here is a skeleton for a Matlab function that is supposed to estimate the value of the alternating Math 230 harmonic series with an error, as estimated above, of no more than *E*, the function's argument. Fill in the body of this function. Note that in the notation used above to describe alternating series, u_i for the alternating Math 230 harmonic series is just $3/i$.

```
function [estimate] = ahs( E )




end
```

Have a Good Summer!