COGS 119 – Fall 2014

Assignment 3 – Due Friday November 7 by Midnight

Remember to comment your m files using % or Matlab → Text → Comment or using %

help is your friend. Submit the indicated files via email to cogs119a@gmail.com. YourLastName should be replaced by your actual last name!

Please read the instructions *carefully*. Everything you need to complete the homework is specified. If you are uncertain about something, read the instructions again. If you are still unsure, ask for clarification.

Part 1. We recommend you read the following carefully in its entirety before starting programming. It can help to structure your program, data structures, and functions (consider the inputs, the outputs, variable scope) conceptually before implementation.

Experiment description: You will be programming an experiment that can be run for a given list of subject names and number of trials. In each trial, the program will present the current subject with an integer between 1 and 40, ask them to indicate as fast and as accurately as possible whether or not it is divisible by 5, record the subject's reaction time (RT) and response, and calculate their accuracy.

Specifications: Your main program will be a function called **YourLastName_MainExp.m**. You will also create another function, **CollectData**. For each function, we provide detailed specifications below. Include **CollectData** in the same m-file as **YourLastName_MainExp.m**, not in a separate m-file of its own. When run, your program will create a struct called *MyData* and **save** this as the mat file **YourLastName_MyDataHW3.mat**.

What to submit: Two files should be submitted for this part. 1) Submit the main experiment function as an m-file (**YourLastName_MainExp.m**). Any other function(s) you create should be included therein (i.e., not saved separately as m-file). 2) You are also asked to submit the data file after running the code with specific inputs. Save the resulting struct *MyData* as a .mat file called **YourLastName_MyDataHW3.mat**, and submit.

Notes on submitting the data file: 1) When you finish coding, run the experiment using the inputs {'Jane', 'Joe', 'Jill'} for the list of subjects, and 10 for the number of trials. 2) Pretend to be Jane, Joe and Jill in turn, and enter "their" responses. 2) Although we want you to submit the mat file after running it for these specific inputs, your experiment should work for any cell array of strings and positive integer as inputs. 3) To save *MyData*, use Matlab's **save** function. The output filename should be specified when you call **save** (see **help save**).

YourLastName_MainExp: Function specifications

The function takes two inputs:

Subjects: list of names, a cell array of strings – e.g., {'Sam', 'Pam', 'Fred', 'Mildred'} or {'Sam'} or {'S1', 'S2'}.
Numtrials: number of trials, a positive integer – e.g., 2 or 10 or 200.

The function should do the following for each subject in *Subjects* (for a total of *Nsub* subjects):

- Generate an array of random integers between 1 and 40, called *Trials*, which has a size of 1 x *Numtrials*.
- Call the function **CollectData**, which runs the experiment for this subject and this array (see specs below).
- Based on the output of **CollectData**, calculate accuracy for each trial.
- Organize the subject's data in the struct *MyData* as described.

The function does not have an output argument, but it does save the 1 x *Nsub* struct called *MyData* as a mat file (see details above on data file).

MyData: Struct specifications

MyData will have the fields *Name* (data type string) and *Data* (data type struct) for each subject. For example, if run with *Subjects* input as {'Sally', 'Steve', 'Sharon'} and *Numtrials* input as 10, *MyData* would look like:

```
1x3 struct array with fields:
    Name
    Data
```

MyData(2) or *MyData(1,2)* would look like:

```
Name: 'Steve'
Data: [1x1 struct]
```

The field *Data* should itself be a struct with *Trials, RT, Response* and *Accuracy* as fields. Each field should be an array of size 1x *Numtrials*. For example, *MyData(2).Data* might look like this:

```
Trials:   [21 16 8 35 12 7 25 14 37 30]
RT:       [1.6192 1.2721 0.9372 1.1214 0.9912 1.1012 0.8101 1.8561 1.4812 1.3823]
Response: [0 0 0 1 0 0 0 0 0 1]
Accuracy: [1 1 1 1 1 1 0 1 1 1]
```

Data.Trials is an array of *Numtrials* random integers between 1 and 40 and is generated for each subject within **YourLastName_MainExp**. The function **CollectData** will present these to the subject, and record a response and a reaction time. *Data.RT* holds the reaction times, and *Data.Response* holds the responses (0 for no, 1 for yes); these arrays will have been returned as output by **CollectData**. *Data.Accuracy* holds whether or not the subject's response was correct (1 for correct, 0 for incorrect). Note that *Data.Accuracy* is calculated in this function, not in **CollectData**. You will do so by using the *Trials* and *Response* subfields. You may want to use the function **mod**.

In the example above, in the first trial, Steve was asked 'Is 21 divisible by 5?'; responded 1.6192 sec after the onset of the trial with a 0 (indicating 21 is not divisible by 5); the accuracy for this trial was calculated as 1 (since this response was correct, 21 is indeed not divisible by 5). In the second trial, he was asked 'Is 16 divisible by 5?' and so on. It looks like Steve made one mistake: on trial 7, he indicated 25 is not divisible by 5.

CollectData: Function specifications

The function takes two inputs:

Name: a string (character array) – e.g., 'Sam' or 'Pam' or 'S1'.
Trials: a 1 x *Numtrials* array – e.g., [23 34 10] for *Numtrials* = 3.

The function should do the following:

- Display a welcome message as follows, with the variable *Name* from the function input used for <Name>: `'Hello <Name>. For each question, please respond as fast and as accurately as possible. Enter 1 for yes, 0 for no. Press any key when you are ready.'` Insert a blank line using the command `fprintf ('\n');` and wait for a key press.
- Once the subject presses a key, start going through *Trials* as follows: For the j^{th} element of *Trials*
  - Display `'Is <num> divisible by 5? Enter 1 for yes, 0 for no.'` and wait for subject input, where <num> is the j^{th} element of *Trials.* If the subject responds with any key other than 0 or 1, display `'Invalid response, try again.'` Repeat if needed until a response of 0 or 1 is received.
  - Once a valid key is received, store the response and reaction time (RT, calculated from the beginning of the current trial using **tic** and **toc**) for the j^{th} trial appropriately, and insert a blank line before moving to the next trial.

The function returns two outputs:

ResponseArray: a 1 x *Numtrials* array containing the subject's responses. (should contain 0s and 1s).
RTArray: a 1 x *Numtrials* array containing the subject's reaction times.

Take a moment to think about how you will program the experiment. If you create any temporary variables, arrays, or structures, please explain your reasoning using comments in your m file (using %).

Below is pseudocode that describes the flow of your program to help you organize your code. Please follow the order specified here and perform the operations in the functions described. Note: this is pseudocode so it is not meant to be used literally nor is the syntax the same as actual Matlab code. Use it as a guide, not as part of code.

function definition for Jones_MainExp – see specs for inputs and outputs
% Do not forget H1
% Do not forget help text
For each subject (up to the number of subjects, *NSub*)
    Set up trials of the experiment (random array of integers).
    Call **CollectData** with the appropriate input and output arguments. Collect data for this subject.
    You should now have a response and RT for every trial. Create an element in *MyData* for this subject and fill the corresponding fields with the appropriate variables in the workspace (see specs for *MyData*).
    Calculate the accuracy of the subject for each trial and assign to the *Accuracy* subfield of *MyData.*
End
Save the 1 x *NSub* struct as Jones_MyDataHW3.mat.
End

function definition for CollectData – see specs for inputs and outputs
Greet subject, show experiment instructions, and insert blank line on screen.
Wait until a key is pressed.
For each trial
    Ask subject if the number corresponding to the trial in the random array is divisible by 5.
    Wait for an input that is 0 or 1.
    Once 0 or 1 is input by subject, record this response and the time elapsed since the beginning of trial.
    Insert blank line on screen.
End
Return outputs
End

Once you are done, run your program by calling **YourLastName_MainExp** on the command window as follows:

```
>> YourLastName_MainExp({'Jane', 'Joe', 'Jill'}), 10);
```

Then pretend to be Jane, Joe and Jill in turn and respond to the prompts. E-mail the files requested in the instructions to the email address specified in the instructions.

Extra credit: Calculate the average reaction time for correct responses only for each subject, and assign it to a field called *MeanRTCorrect* in *MyData*. Submit as above.

Part 2

Download cute_dogs.jpg on your computer.

Read in the image as a variable *Myimg* and display in a figure window using **imread.**

Check out dimensions and values in *Myimg*. Create a new image *Myimg2* that displays the middle dog in *Myimg* vertically. (You will need to calculate a little to find the indices that will give you the middle dog. Your new image should therefore differ only in the number of elements in the second dimension). If you view *Myimg2* in the figure window (axis image and axis off), it should look something like this:

Save the image (from the figure window or using the function **saveas**) as YourLastName_Myimg2.jpg.

Next, create *Myimg3*, which gets rid of all the green in the original image of all dogs. Save the image (from the figure window or using **saveas**) YourLastName_Myimg3.jpg.

Save the commands you used in **YourLastName_Images.m** and submit it together with YourLastName_Myimg2.jpg and YourLastName_Myimg3.jpg.