Assignment 5 – Due DEC 2 Tuesday @ Midnight (but you can try finishing before thanksgiving and enjoy the holiday with one less thing to worry about)

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.

Question 1 (3.5 points)

We studied a program in class that will do a whole bunch of things on the screen in class: **FunScreen.m**. Go into the code and follow what is happening. Do you see how you make a circle? A rectangle? Put text on the screen? Read the help for all Screen functions that are called in **FunScreen.m** (Screen('OpenWindow?') or Screen OpenWindow?).

Now, make your own fun screen. Call it **YourLastName_FunScreen.m**. There are no rules, except your program must be at least as long and complex as the example **FunScreen**. If you want to do more or more complicated visuals, that's fine (just keep it under a minute or so!)

We want to see your comments under *every* call to Screen explaining what you are doing (e.g., I am drawing a yellow square with a red circle inside it at the center of the screen).

Use an 800 x 600 screen that doesn't cover the entire screen like we did in class. To submit, email **YourLastName_FunScreen.m**

Question 2 (6.5 points)

(A) Write a program called **YourLastName_ImageQuadrants.m**

This is what your program should do:

Use Screen to open a screen and prepare it. Use a screen size of [0,0,800,600]. Use background color black throughout; text color that can be any color you want that is clearly visible over black. When there is text to display, you can use any size or font as long as it's legible and all of the text fits in the screen. Text should be displayed *approximately centered* on the screen in the horizontal (x) dimension, meaning you should roughly try and center each sentence, but it is not necessary to be precise (unlike where you present the image, see below).

Read in the image cornucopia (download jpg file from the website).

There are 4 possible locations to present the image. Upper left quadrant, upper right quadrant, lower left quadrant and lower right quadrant of the screen, referred to as locations 1-4.

Use a loop to present the image at each of these locations for 3 seconds. You can go in this order or vary your order; it doesn't matter, as long as you display the image in each quadrant. Your image should be *precisely centered within each quadrant*. (Hint: One way to do this is to prepare an array for the locations that correspond to the center of each quadrant that you can access in your loop). Your code should be able to run at different screen sizes, always displaying the images centered in each quadrant. This means you should not hard code these locations. You will need to calculate these coordinates taking screen size into account.

To center precisely you will need to take the size of the image cornucopia.jpg into account – remember LeTteRBox.

When we run the program this is what we should see:

A Psychtoolbox screen opens with a black background (which stays black throughout).
This text on the screen on a black background for 3 seconds:
`I will display a Thanksgiving cornucopia at four locations`

Repeat the following for the four locations:

- Black screen for 1 second.
- This text on the screen for 2 seconds:
  `Now displaying location <n>` (where n goes from 1 to 4).
- The image is displayed precisely centered at that location for 3 seconds.

When done displaying 4 times in the different locations:

- Black screen for 1 second.
- Display on the screen on a black background for 3 seconds:
  `That was a Thanksgiving cornucopia at four locations`
- Close screen and exit.

Bonus: Update the program above such that whenever you show "Now displaying location <n>", this text is displayed in black text inside a blue rectangle horizontally and vertically centered on the screen. The rectangle can be any size as long as it's centered and the text fits in it. The blue can be any shade as long as the black text is legible.

(B) Create a new file that extends the program you wrote in Part A that also gets key presses after each image display. Hint: See **keyboardexamplescreen.m** on the class website as an example to do this (except you do not need to write the data into a file). Anything not specified here should stay the same as Part A. Your new program **YourLastName_ImageQuadrantsResponse.m** should work as follows:

At the beginning of the program, display the following text on Screen for 3 seconds:
`If the image is on the left, press z; if it is on the right, press m`

Then, for each quadrant repeat the following:

- Display the cornucopia in one of the four quadrants as in Part A for 3 seconds.
- Get a key press, record reaction time and the key pressed for that location using **KbCheck**.

- Check the accuracy of the response and display text feedback to the user (1 second long presentation, approximately centered on the screen):
    - If the image was on the left and the subject responded with 'z', display `Correct` on the screen, otherwise display `Incorrect`.
    - If the image was on the right and the subject responded with 'm', display `Correct` on the screen, otherwise display `Incorrect`.
    - If the subject does not respond within 3 seconds during the image display, then display `No response` on the screen and move to the next trial (which will display image in the next quadrant).

Bonus: Extend the program in Part B to run as a function that runs in the same way except it also returns the user responses in two arrays, *AccArray* and *RTArray*, each of size 1x4 where the elements are the accuracy (0 or 1) and the reaction time (in seconds) of the user for the 4 locations. In the event user does not press a key within 3 seconds, set accuracy and RT for that location to 9.

E-mail **YourLastName_ImageQuadrants.m** and **YourLastName_ImageQuadrantsResponse.m.**