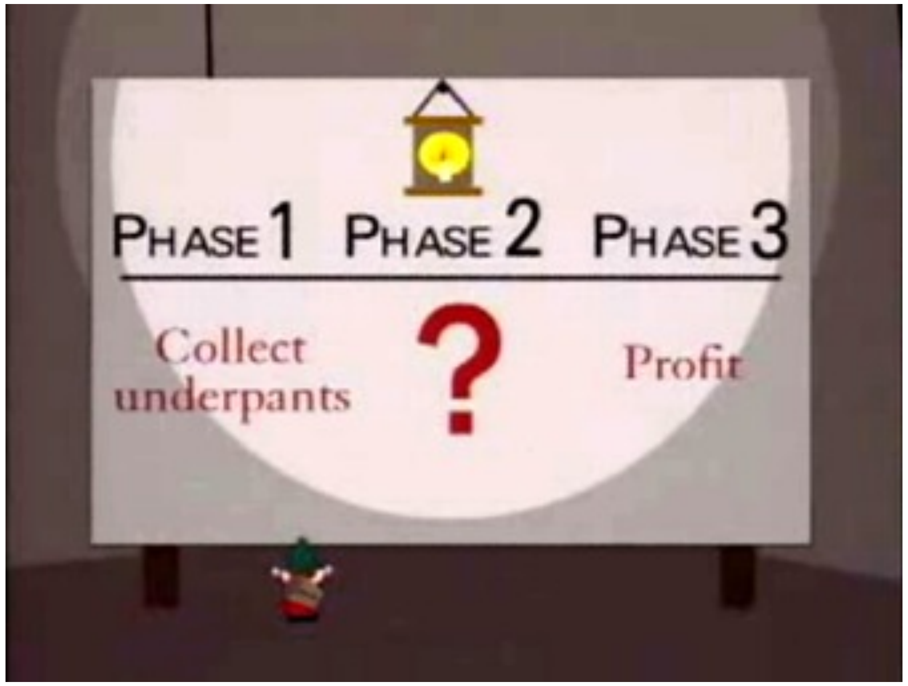
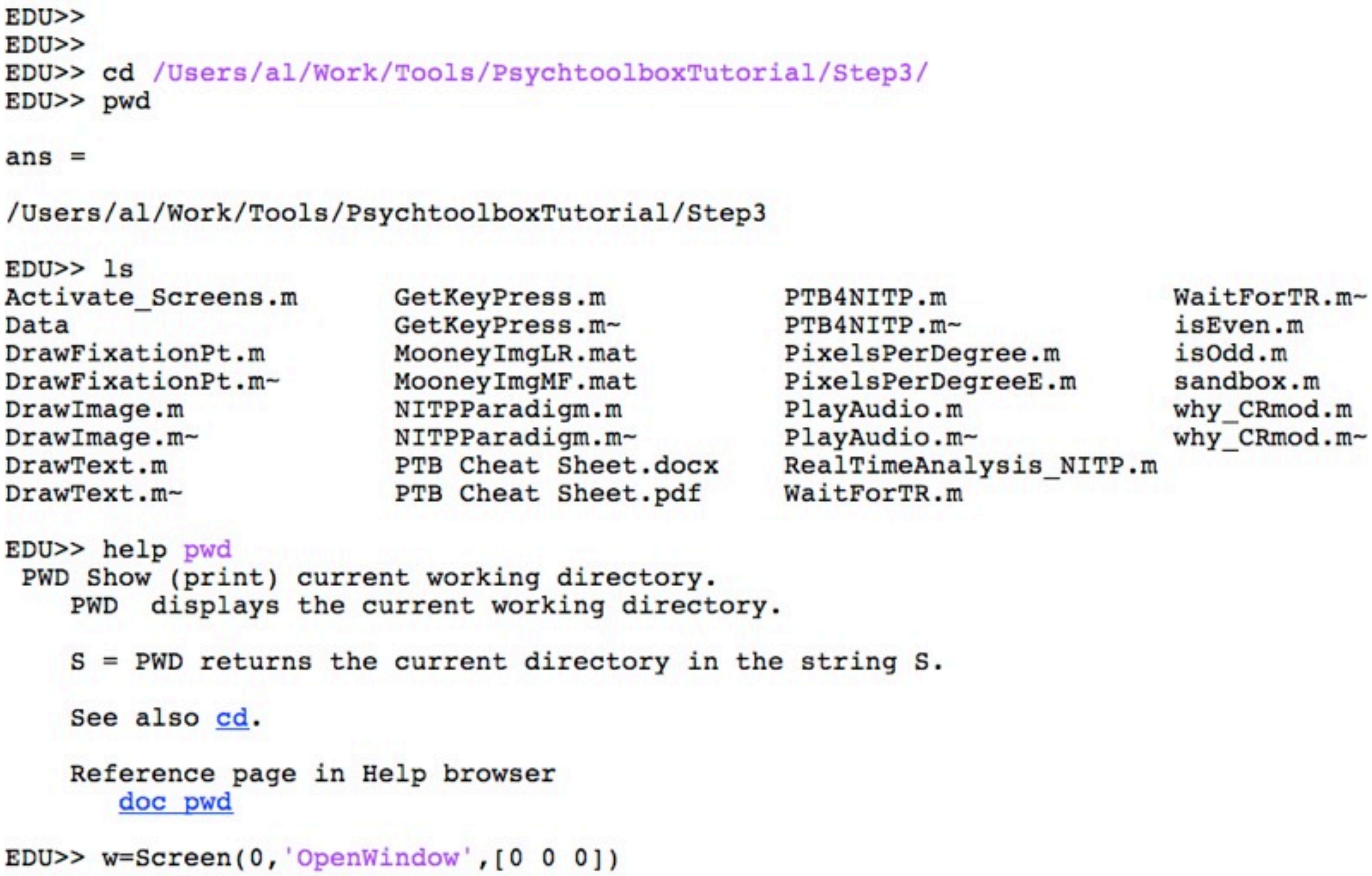
WELCOME TO THE NITP Psychtoolbox Tutorial 2012!

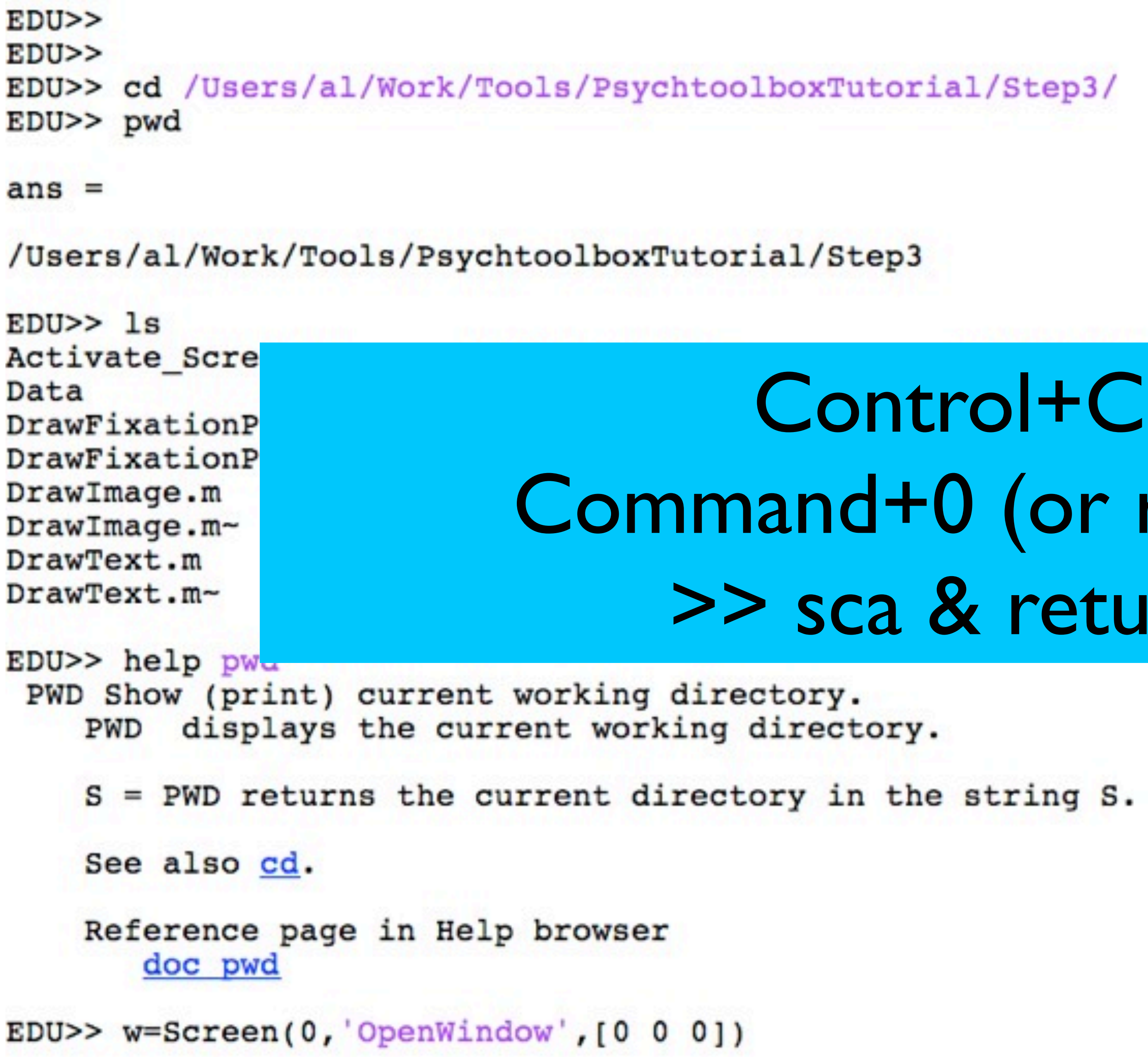


Basic Steps



```
EDU>>
EDU>>
EDU>> cd /Users/al/Work/Tools/PsychtoolboxTutorial/Step3/
EDU>> pwd

ans =

/Users/al/Work/Tools/PsychtoolboxTutorial/Step3

EDU>> ls
Activate_Screens.m      GetKeyPress.m           PTB4NITP.m              WaitForTR.m~
Data                    GetKeyPress.m~          PTB4NITP.m~             isEven.m
DrawFixationPt.m        MooneyImgLR.mat         PixelsPerDegree.m       isOdd.m
DrawFixationPt.m~       MooneyImgMF.mat         PixelsPerDegreeE.m      sandbox.m
DrawImage.m             NITPParadigm.m          PlayAudio.m             why_CRmod.m
DrawImage.m~            NITPParadigm.m~         PlayAudio.m~            why_CRmod.m~
DrawText.m              PTB Cheat Sheet.docx    RealTimeAnalysis_NITP.m
DrawText.m~             PTB Cheat Sheet.pdf     WaitForTR.m

EDU>> help pwd
 PWD Show (print) current working directory.
    PWD  displays the current working directory.

    S = PWD returns the current directory in the string S.

    See also cd.

    Reference page in Help browser
       doc pwd

EDU>> w=Screen(0,'OpenWindow',[0 0 0])
```

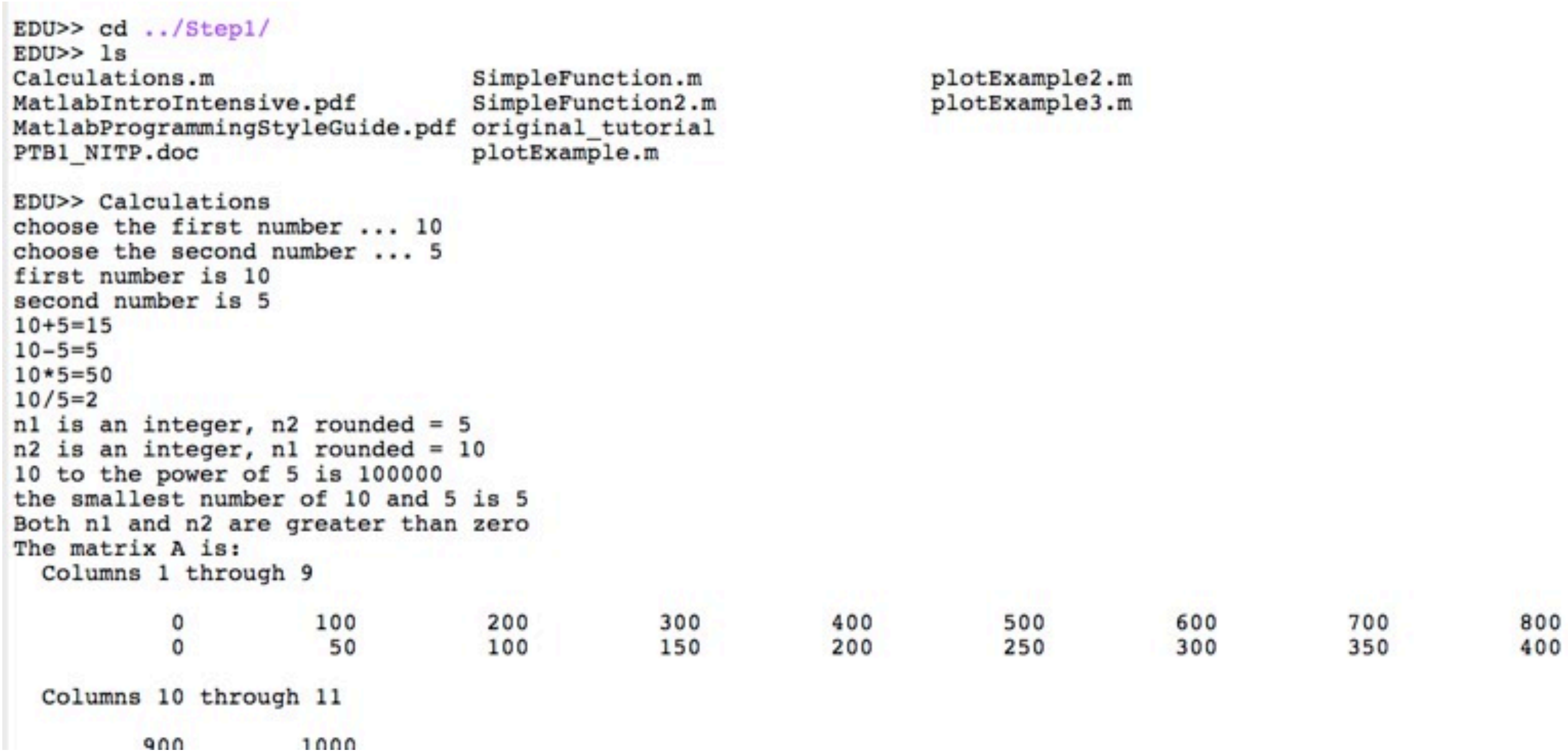
Basic Steps



```
EDU>>
EDU>>
EDU>> cd /Users/al/Work/Tools/PsychtoolboxTutorial/Step3/
EDU>> pwd

ans =

/Users/al/Work/Tools/PsychtoolboxTutorial/Step3

EDU>> ls
Activate_Scre
Data
DrawFixationP
DrawFixationP
DrawImage.m
DrawImage.m~
DrawText.m
DrawText.m~

EDU>> help pw
 PWD Show (print) current working directory.
    PWD  displays the current working directory.

    S = PWD returns the current directory in the string S.

    See also cd.

    Reference page in Help browser
       doc pwd

EDU>> w=Screen(0,'OpenWindow',[0 0 0])
```

Control+C
Command+0 (or return)
>> sca & return

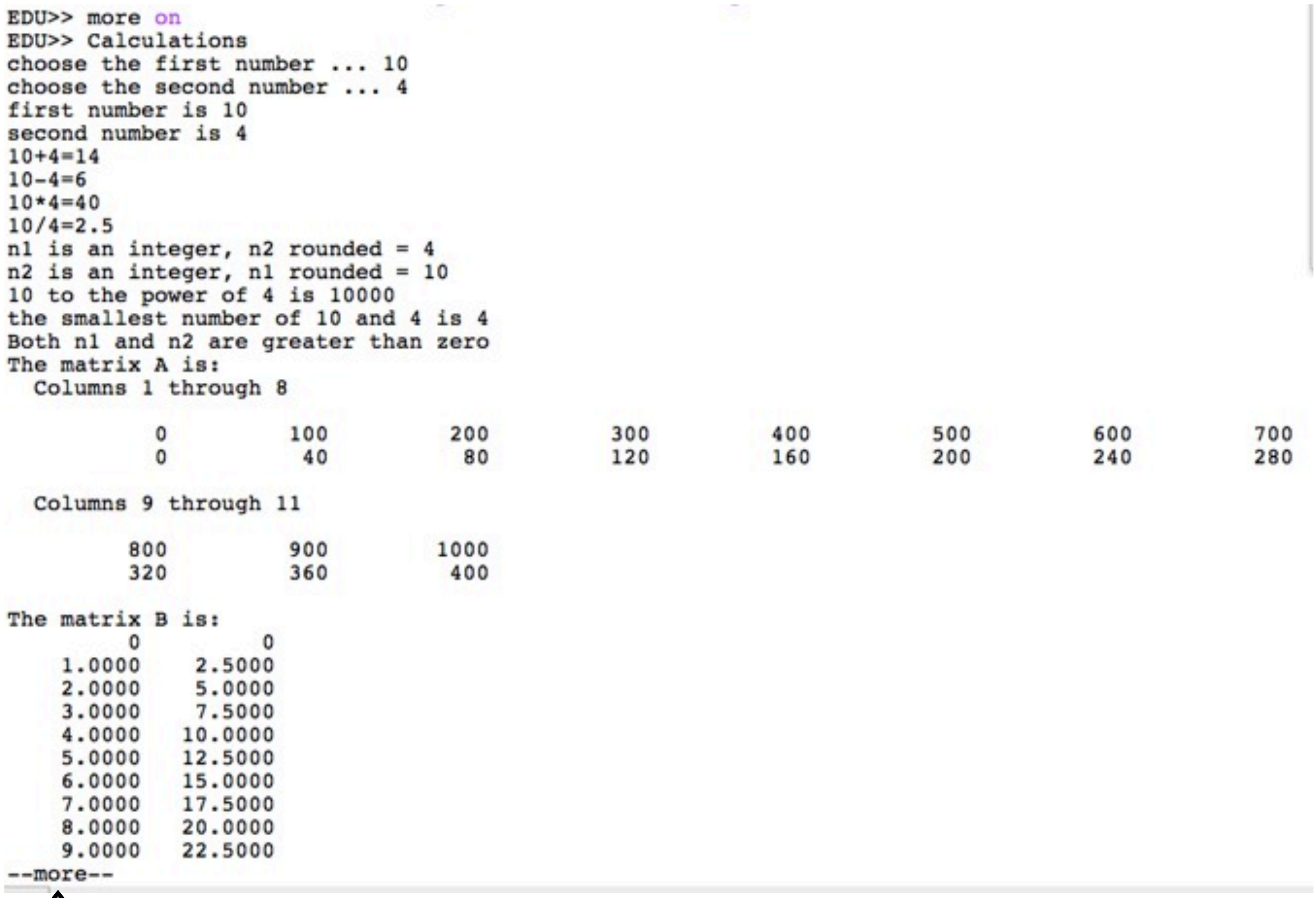
Step 1: Matlab



```
EDU>> cd ../Step1/
EDU>> ls
Calculations.m                 SimpleFunction.m               plotExample2.m
MatlabIntroIntensive.pdf       SimpleFunction2.m              plotExample3.m
MatlabProgrammingStyleGuide.pdf original_tutorial
PTB1_NITP.doc                  plotExample.m

EDU>> Calculations
choose the first number ... 10
choose the second number ... 5
first number is 10
second number is 5
10+5=15
10-5=5
10*5=50
10/5=2
n1 is an integer, n2 rounded = 5
n2 is an integer, n1 rounded = 10
10 to the power of 5 is 100000
the smallest number of 10 and 5 is 5
Both n1 and n2 are greater than zero
The matrix A is:
  Columns 1 through 9

         0         100         200         300         400         500         600         700         800
         0          50         100         150         200         250         300         350         400

  Columns 10 through 11

       900        1000
```

too much output
...make less with more

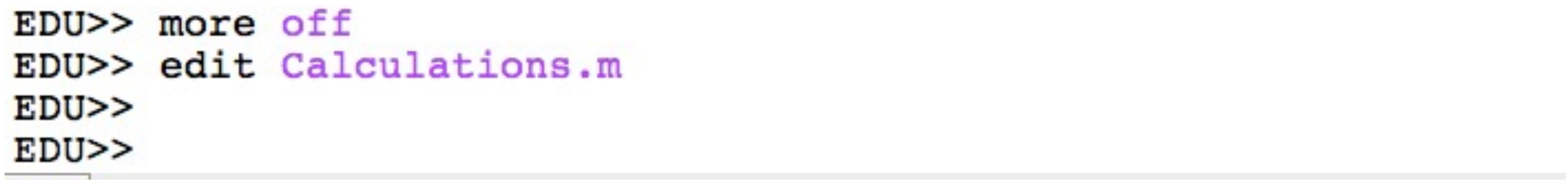
Step 1: Matlab



```
EDU>> more on
EDU>> Calculations
choose the first number ... 10
choose the second number ... 4
first number is 10
second number is 4
10+4=14
10-4=6
10*4=40
10/4=2.5
n1 is an integer, n2 rounded = 4
n2 is an integer, n1 rounded = 10
10 to the power of 4 is 10000
the smallest number of 10 and 4 is 4
Both n1 and n2 are greater than zero
The matrix A is:
  Columns 1 through 8

         0         100         200         300         400         500         600         700
         0          40          80         120         160         200         240         280

  Columns 9 through 11

       800         900        1000
       320         360         400

The matrix B is:
         0         0
    1.0000    2.5000
    2.0000    5.0000
    3.0000    7.5000
    4.0000   10.0000
    5.0000   12.5000
    6.0000   15.0000
    7.0000   17.5000
    8.0000   20.0000
    9.0000   22.5000
--more--
```

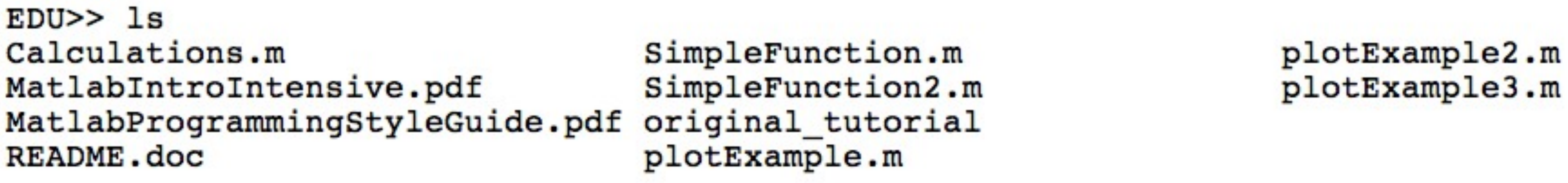
...make less with more

Step 1: Matlab



```
EDU>> more off
EDU>> edit Calculations.m
EDU>>
EDU>>
```

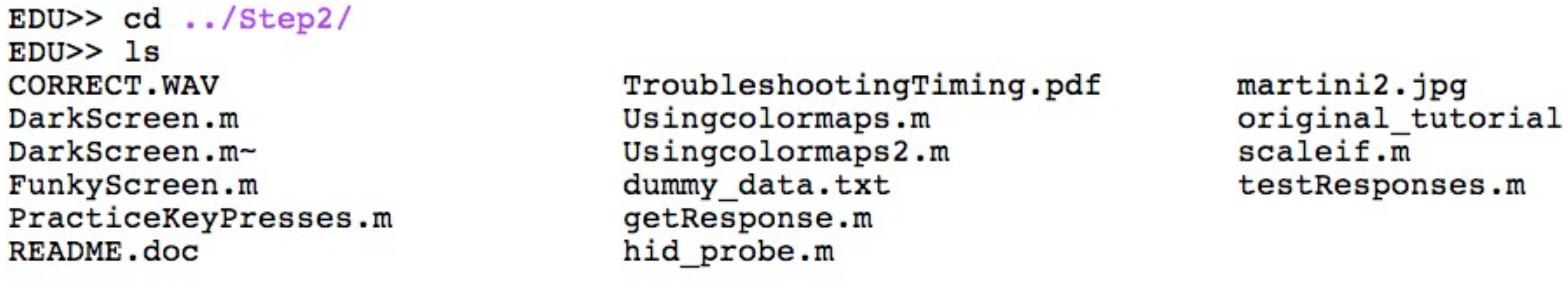
...edit to view files in editor (.m files?)



```
EDU>> ls
Calculations.m                    SimpleFunction.m               plotExample2.m
MatlabIntroIntensive.pdf          SimpleFunction2.m              plotExample3.m
MatlabProgrammingStyleGuide.pdf original_tutorial
README.doc                        plotExample.m
```

...follow README.doc
...run functions/scripts by typing name
...edit to understand code
...check out MatlabIntroIntensive.PDF for a serious intro

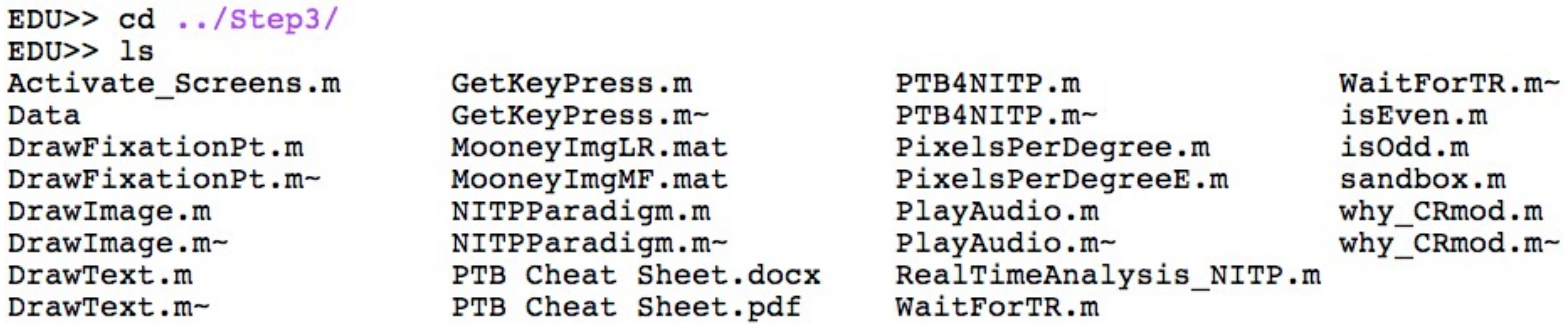
Step 2: ??



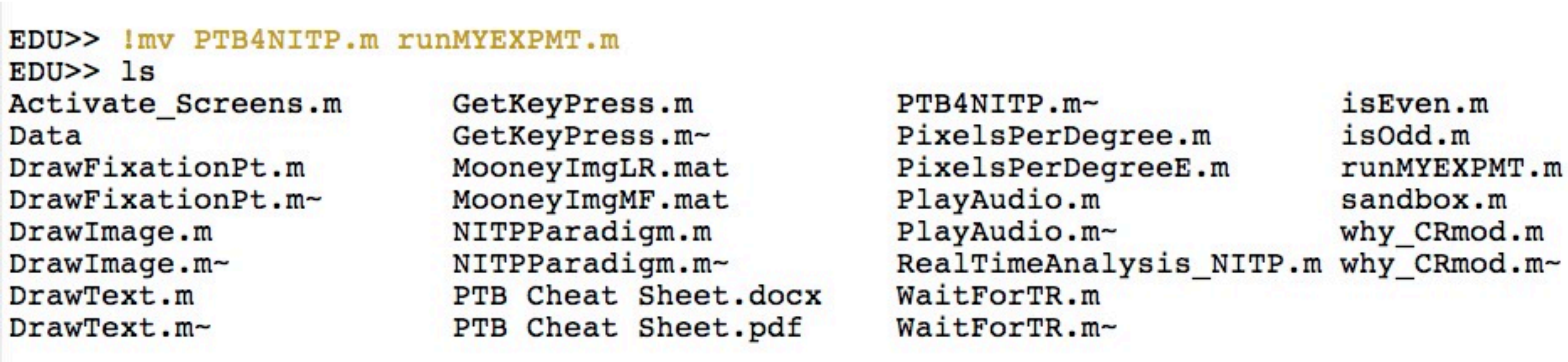
```
EDU>> cd ../Step2/
EDU>> ls
CORRECT.WAV                   TroubleshootingTiming.pdf     martini2.jpg
DarkScreen.m                  Usingcolormaps.m              original_tutorial
DarkScreen.m~                 Usingcolormaps2.m             scaleif.m
FunkyScreen.m                 dummy_data.txt                testResponses.m
PracticeKeyPresses.m          getResponse.m
README.doc                    hid_probe.m
```

README.doc is the tutorial
describing the core of PTB functionality including:
- demos, stimuli, responses
- timing control, code checks, keyboard checks etc

Step 3: Profit

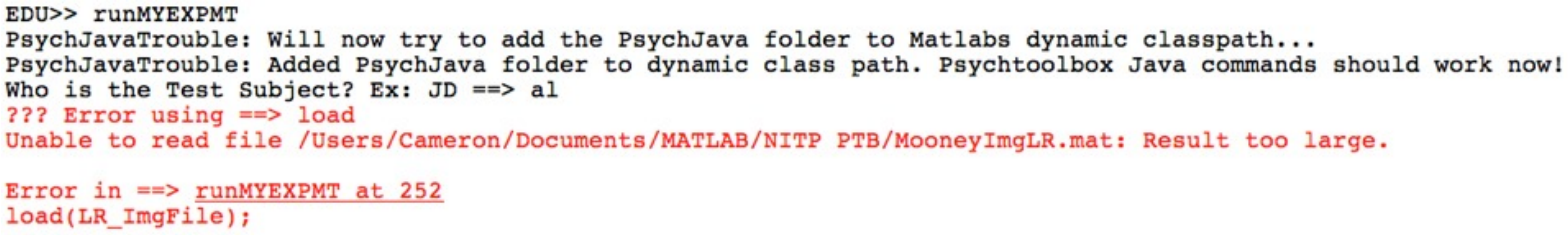


```
EDU>> cd ../Step3/
EDU>> ls
Activate_Screens.m      GetKeyPress.m           PTB4NITP.m              WaitForTR.m~
Data                    GetKeyPress.m~          PTB4NITP.m~             isEven.m
DrawFixationPt.m        MooneyImgLR.mat         PixelsPerDegree.m       isOdd.m
DrawFixationPt.m~       MooneyImgMF.mat         PixelsPerDegreeE.m      sandbox.m
DrawImage.m             NITPParadigm.m          PlayAudio.m             why_CRmod.m
DrawImage.m~            NITPParadigm.m~         PlayAudio.m~            why_CRmod.m~
DrawText.m              PTB Cheat Sheet.docx    RealTimeAnalysis_NITP.m
DrawText.m~             PTB Cheat Sheet.pdf     WaitForTR.m
```



```
EDU>> !mv PTB4NITP.m runMYEXPMT.m
EDU>> ls
Activate_Screens.m      GetKeyPress.m           PTB4NITP.m~             isEven.m
Data                    GetKeyPress.m~          PixelsPerDegree.m       isOdd.m
DrawFixationPt.m        MooneyImgLR.mat         PixelsPerDegreeE.m      runMYEXPMT.m
DrawFixationPt.m~       MooneyImgMF.mat         PlayAudio.m             sandbox.m
DrawImage.m             NITPParadigm.m          PlayAudio.m~            why_CRmod.m
DrawImage.m~            NITPParadigm.m~         RealTimeAnalysis_NITP.m why_CRmod.m~
DrawText.m              PTB Cheat Sheet.docx    WaitForTR.m
DrawText.m~             PTB Cheat Sheet.pdf     WaitForTR.m~
```

Paths

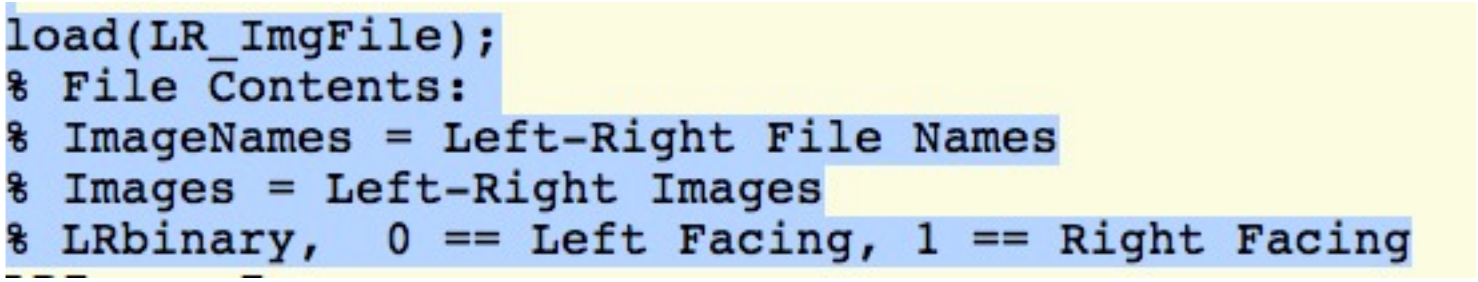


```
EDU>> runMYEXPMT
PsychJavaTrouble: Will now try to add the PsychJava folder to Matlabs dynamic classpath...
PsychJavaTrouble: Added PsychJava folder to dynamic class path. Psychtoolbox Java commands should work now!
Who is the Test Subject? Ex: JD ==> al
??? Error using ==> load
Unable to read file /Users/Cameron/Documents/MATLAB/NITP PTB/MooneyImgLR.mat: Result too large.

Error in ==> runMYEXPMT at 252
load(LR_ImgFile);
```

...the problem is that the wrong path is set and the program can't find the file it's trying to load (click on the error link to get to line 252)

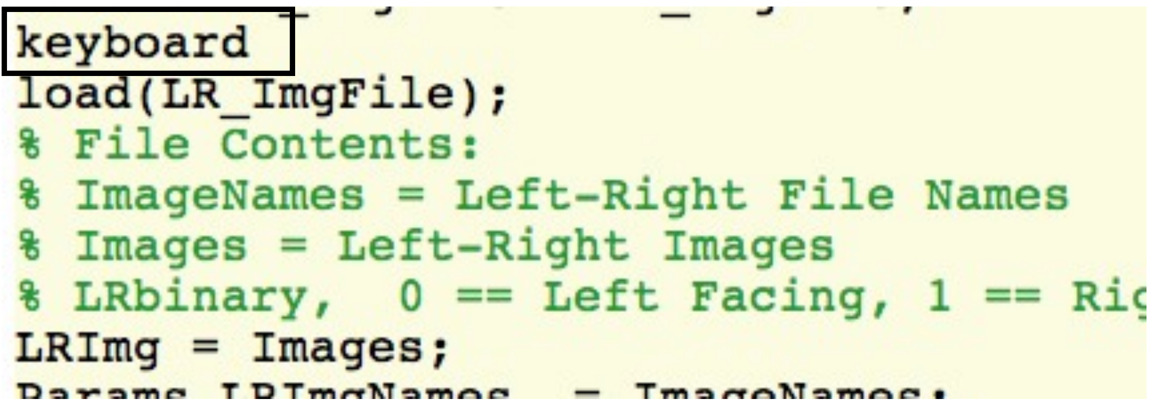
--- psst, runMYEXPMT.m is a *script* ---

Paths



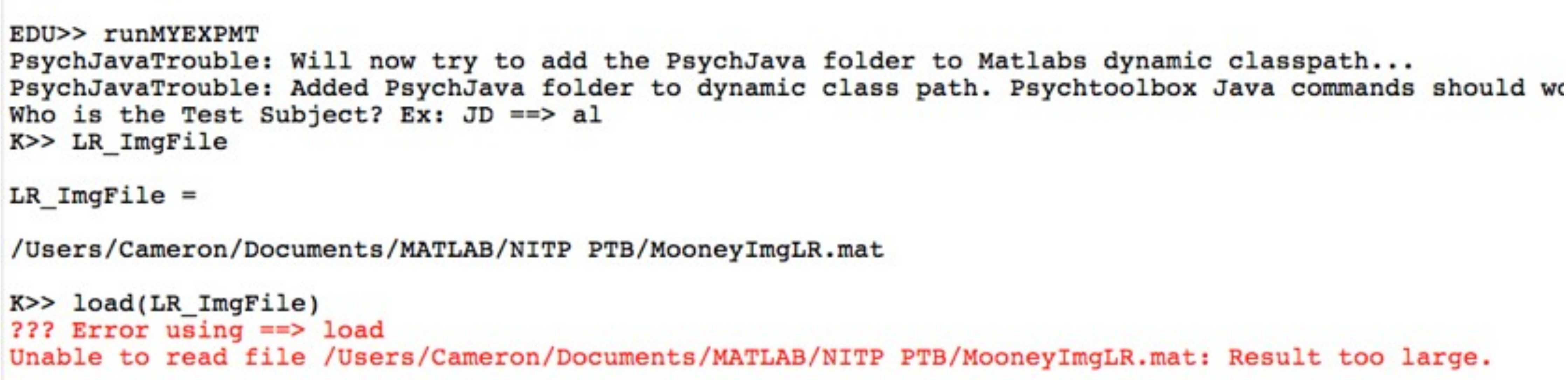
```
load(LR_ImgFile);
% File Contents:
% ImageNames = Left-Right File Names
% Images = Left-Right Images
% LRbinary,  0 == Left Facing, 1 == Right Facing
```

ADD
KEYBOARD



```
keyboard
load(LR_ImgFile);
% File Contents:
% ImageNames = Left-Right File Names
% Images = Left-Right Images
% LRbinary,  0 == Left Facing, 1 == Rig
LRImg = Images;
```

RUN IT AGAIN - INSPECT PROBLEM

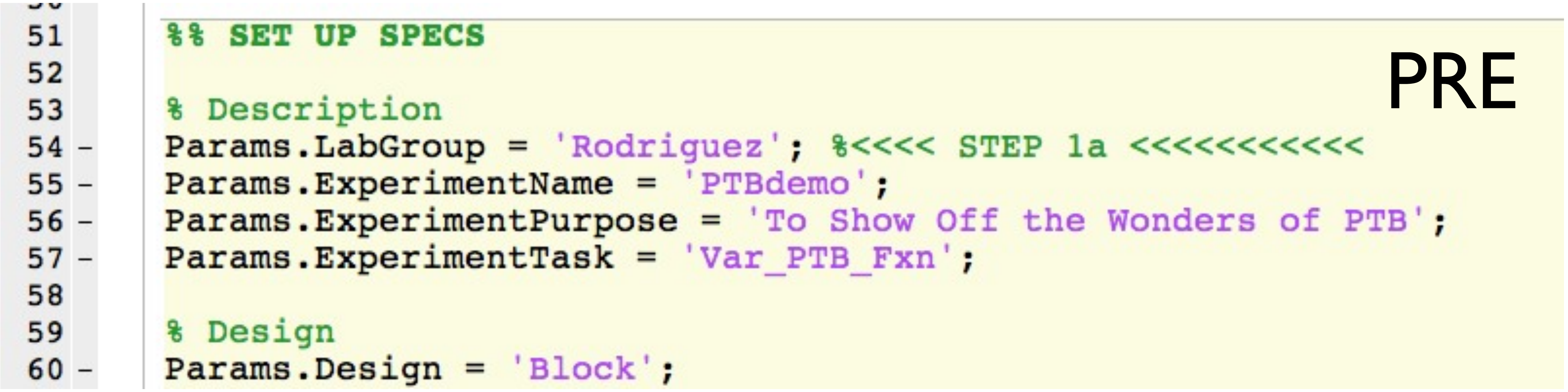


```
EDU>> runMYEXPMT
PsychJavaTrouble: Will now try to add the PsychJava folder to Matlabs dynamic classpath...
PsychJavaTrouble: Added PsychJava folder to dynamic class path. Psychtoolbox Java commands should w
Who is the Test Subject? Ex: JD ==> al
K>> LR_ImgFile

LR_ImgFile =

/Users/Cameron/Documents/MATLAB/NITP PTB/MooneyImgLR.mat

K>> load(LR_ImgFile)
??? Error using ==> load
Unable to read file /Users/Cameron/Documents/MATLAB/NITP PTB/MooneyImgLR.mat: Result too large.

K>> dbquit
```

DON'T FORGET TO REMOVE KEYBOARD

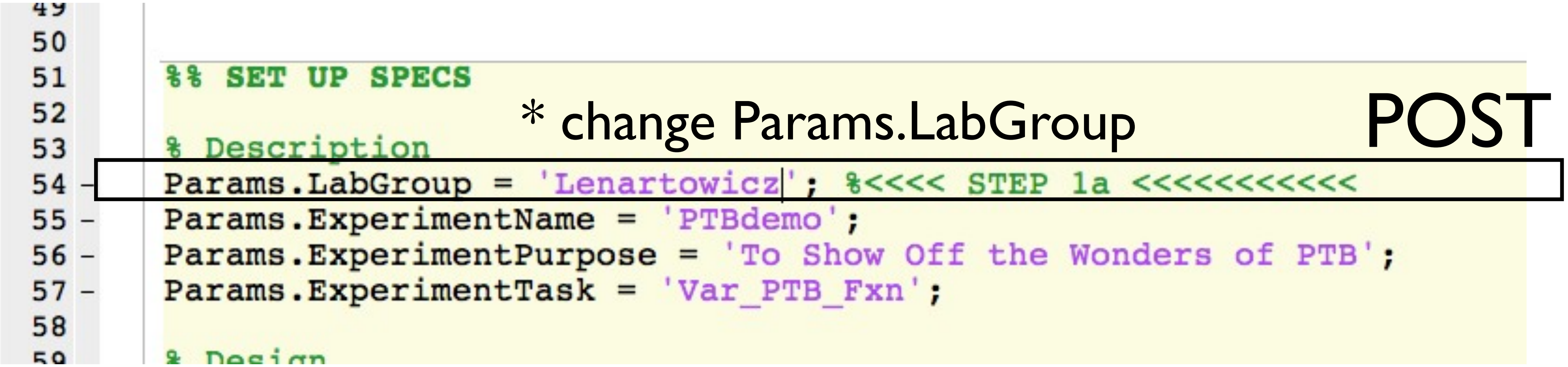
CORRECT PROBLEM by EDITING FILE

STEP 1a <<<<<



PRE

```
%% SET UP SPECS

% Description
Params.LabGroup = 'Rodriguez'; %<<<< STEP 1a <<<<<<<<<<<
Params.ExperimentName = 'PTBdemo';
Params.ExperimentPurpose = 'To Show Off the Wonders of PTB';
Params.ExperimentTask = 'Var_PTB_Fxn';

% Design
Params.Design = 'Block';
```

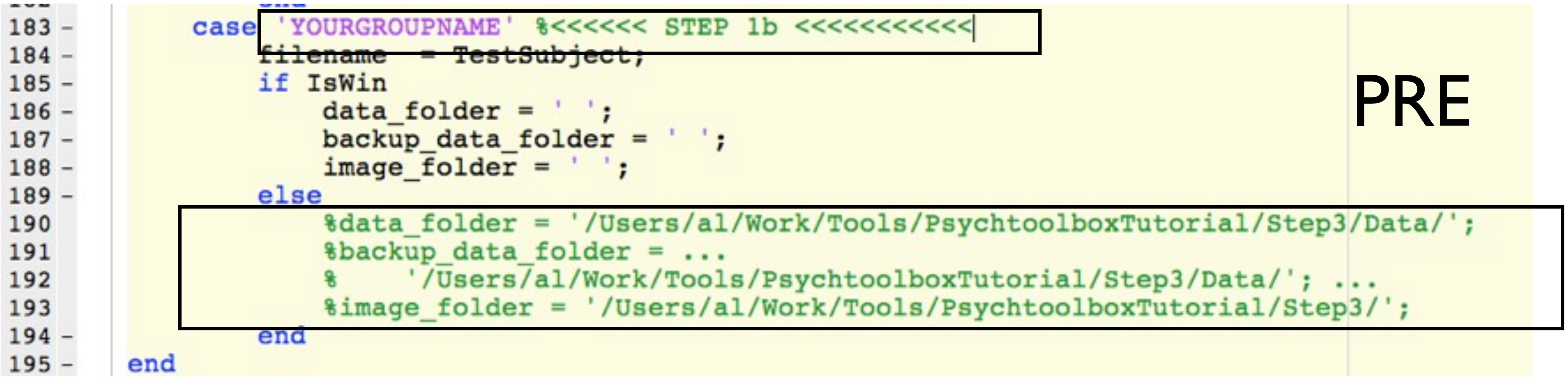


POST

* change Params.LabGroup

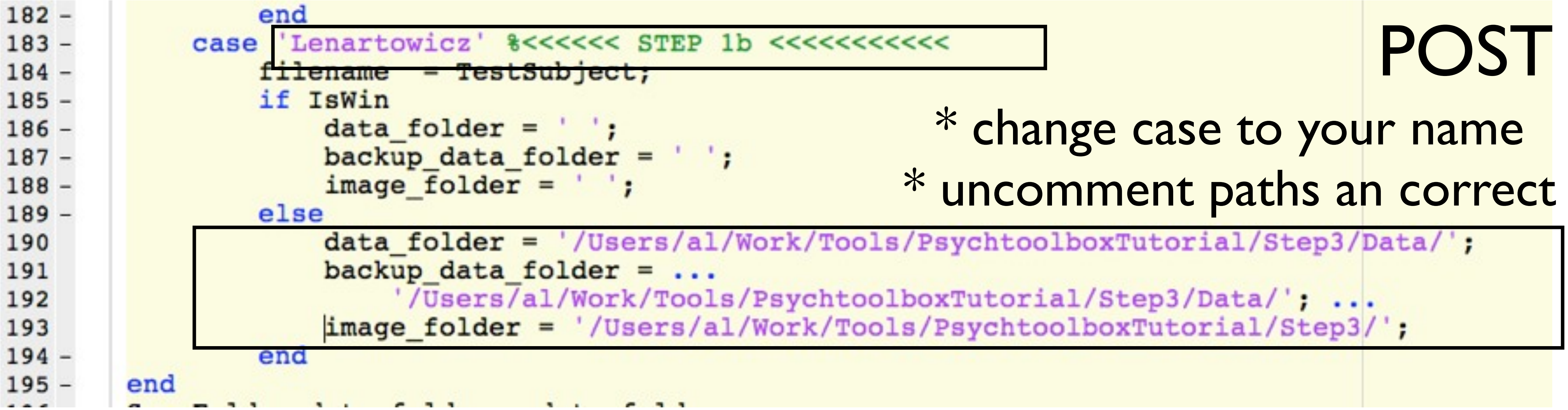
```


%% SET UP SPECS

% Description
Params.LabGroup = 'Lenartowicz'; %<<<< STEP 1a <<<<<<<<<<<
Params.ExperimentName = 'PTBdemo';
Params.ExperimentPurpose = 'To Show Off the Wonders of PTB';
Params.ExperimentTask = 'Var_PTB_Fxn';

% Design
```

CORRECT PROBLEM by EDITING FILE
STEP 1b <<<<<



PRE

```
    case 'YOURGROUPNAME' %<<<<<< STEP 1b <<<<<<<<<<<<
        filename  = TestSubject;
        if IsWin
            data_folder = ' ';
            backup_data_folder = ' ';
            image_folder = ' ';
        else
            %data_folder = '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/Data/';
            %backup_data_folder = ...
            %     '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/Data/'; ...
            %image_folder = '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/';
        end
end
```



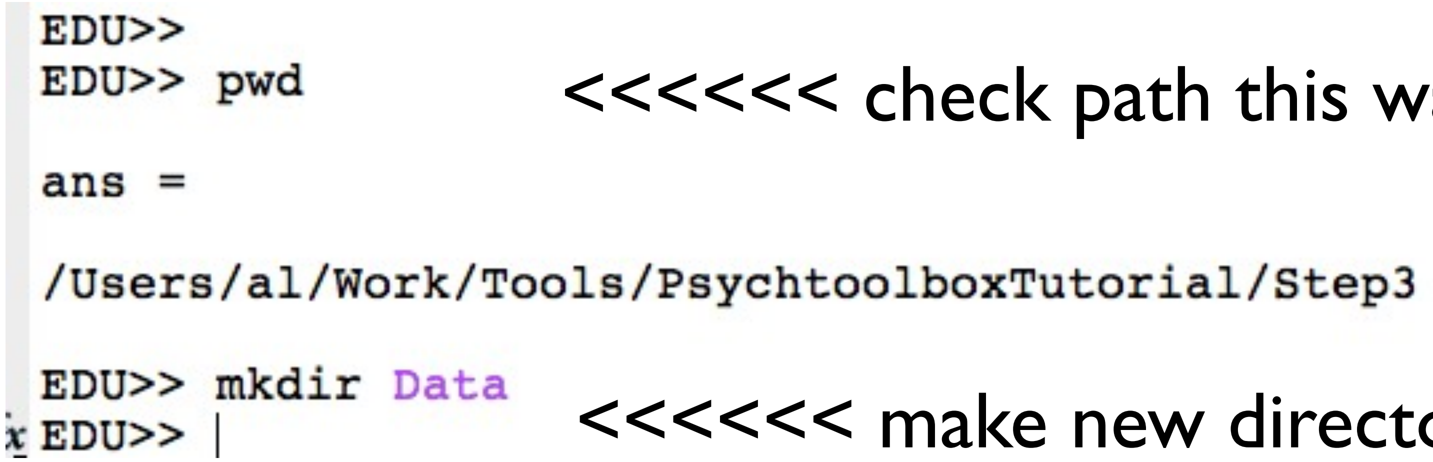
POST

- * change case to your name
- * uncomment paths an correct

```
        end
    case 'Lenartowicz' %<<<<<< STEP 1b <<<<<<<<<<<<
        filename  = TestSubject;
        if IsWin
            data_folder = ' ';
            backup_data_folder = ' ';
            image_folder = ' ';
        else
            data_folder = '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/Data/';
            backup_data_folder = ...
                '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/Data/'; ...
            image_folder = '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/';
        end
end
```

CORRECT PROBLEM by EDITING FILE

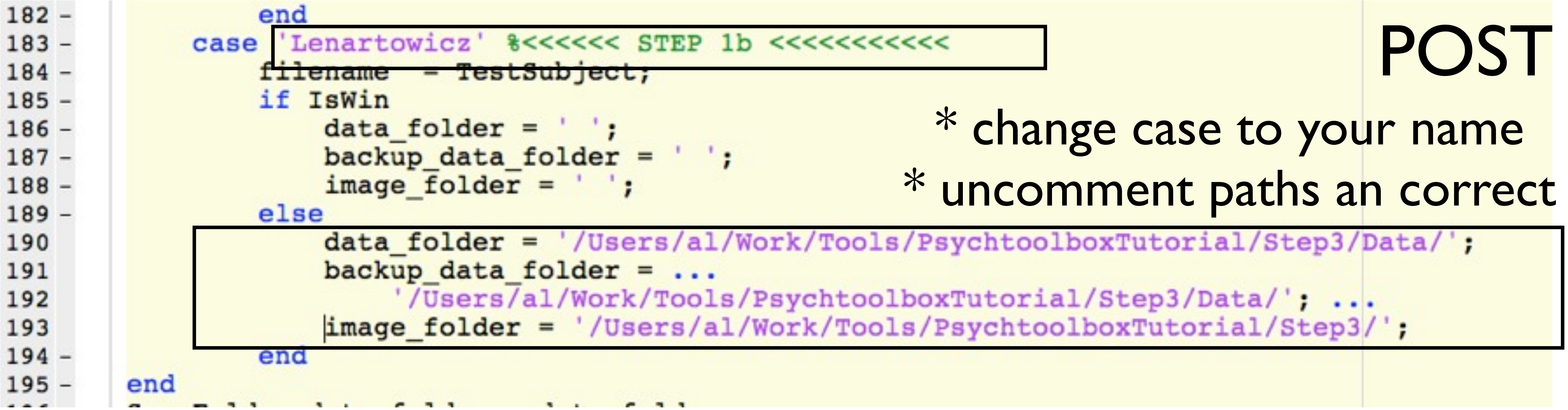
STEP 1b <<<<<



```
EDU>>
EDU>> pwd

ans =

/Users/al/Work/Tools/PsychtoolboxTutorial/Step3

EDU>> mkdir Data
EDU>> |
```

<<<<<< check path this way

<<<<<< make new directory this way



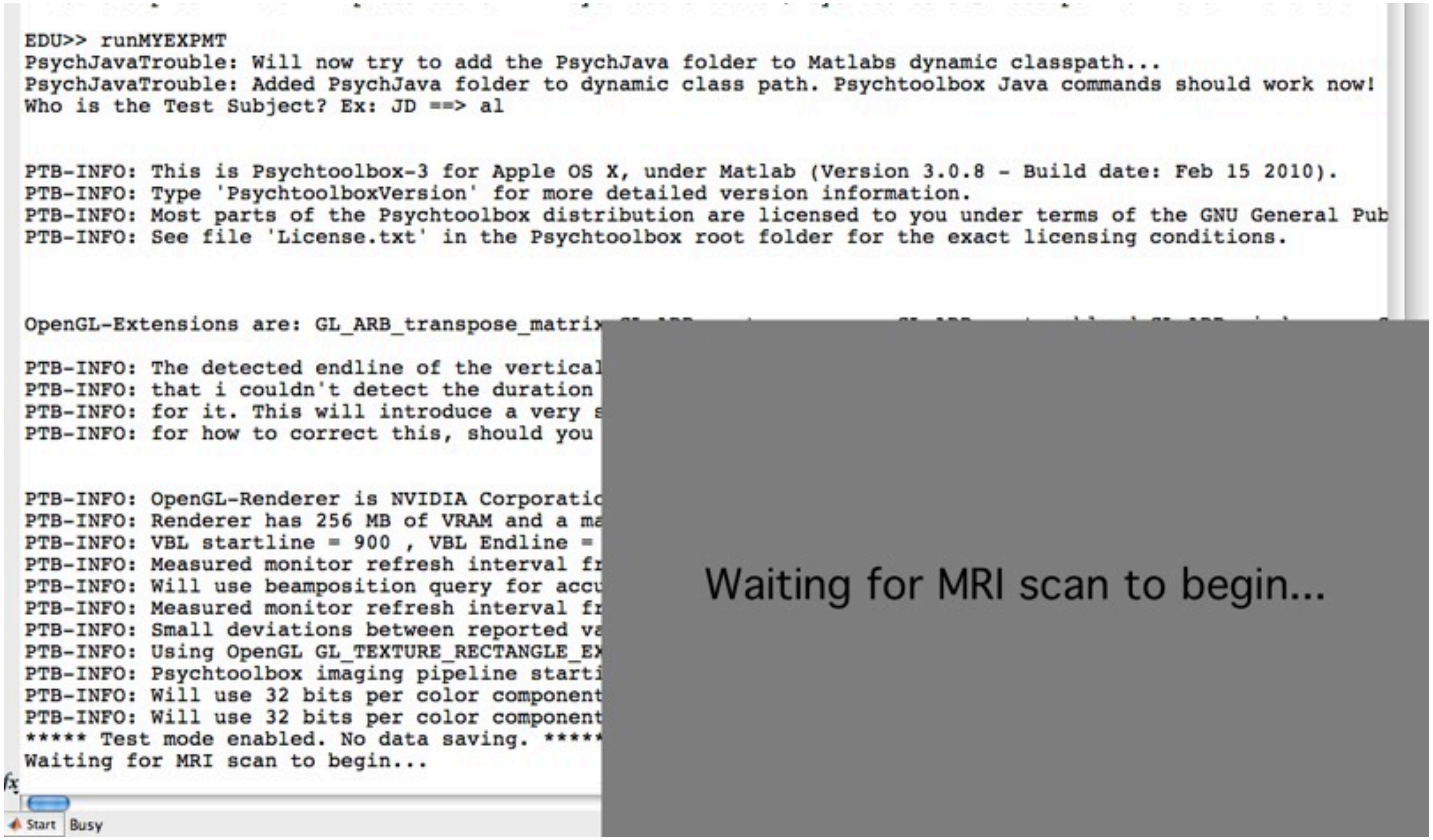
```
        end
    case 'Lenartowicz' %<<<<<< STEP 1b <<<<<<<<<<<<
        filename   = TestSubject;
        if IsWin
            data_folder = ' ';
            backup_data_folder = ' ';
            image_folder = ' ';
        else
            data_folder = '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/Data/';
            backup_data_folder = ...
                '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/Data/'; ...
            image_folder = '/Users/al/Work/Tools/PsychtoolboxTutorial/Step3/';
        end
end
```

POST

* change case to your name

* uncomment paths an correct

RUN YOUR EXPERIMENT



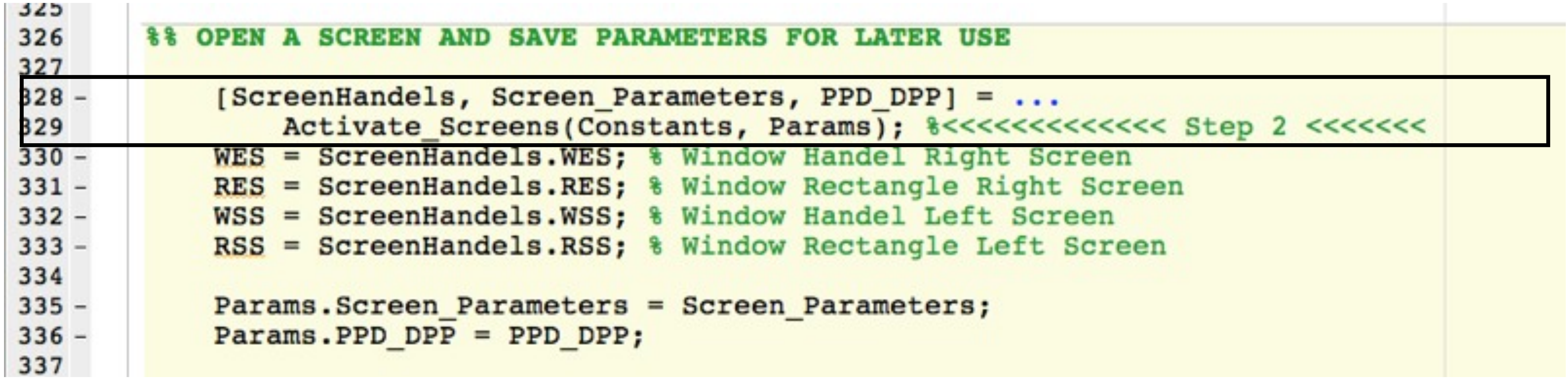
```
EDU>> runMYEXPMT
PsychJavaTrouble: Will now try to add the PsychJava folder to Matlabs dynamic classpath...
PsychJavaTrouble: Added PsychJava folder to dynamic class path. Psychtoolbox Java commands should work now!
Who is the Test Subject? Ex: JD ==> al


PTB-INFO: This is Psychtoolbox-3 for Apple OS X, under Matlab (Version 3.0.8 - Build date: Feb 15 2010).
PTB-INFO: Type 'PsychtoolboxVersion' for more detailed version information.
PTB-INFO: Most parts of the Psychtoolbox distribution are licensed to you under terms of the GNU General Pub
PTB-INFO: See file 'License.txt' in the Psychtoolbox root folder for the exact licensing conditions.



OpenGL-Extensions are: GL_ARB_transpose_matrix

PTB-INFO: The detected endline of the vertical
PTB-INFO: that i couldn't detect the duration
PTB-INFO: for it. This will introduce a very s
PTB-INFO: for how to correct this, should you


PTB-INFO: OpenGL-Renderer is NVIDIA Corporatio
PTB-INFO: Renderer has 256 MB of VRAM and a ma
PTB-INFO: VBL startline = 900 , VBL Endline =
PTB-INFO: Measured monitor refresh interval fr
PTB-INFO: Will use beamposition query for accu
PTB-INFO: Measured monitor refresh interval fr
PTB-INFO: Small deviations between reported va
PTB-INFO: Using OpenGL GL_TEXTURE_RECTANGLE_EX
PTB-INFO: Psychtoolbox imaging pipeline starti
PTB-INFO: Will use 32 bits per color component
PTB-INFO: Will use 32 bits per color component
***** Test mode enabled. No data saving. *****
Waiting for MRI scan to begin...
```

Waiting for MRI scan to begin...

PRESS 5 to pretend you are the scanner!

SCRIPT OPENS A DEBUGGING WINDOW

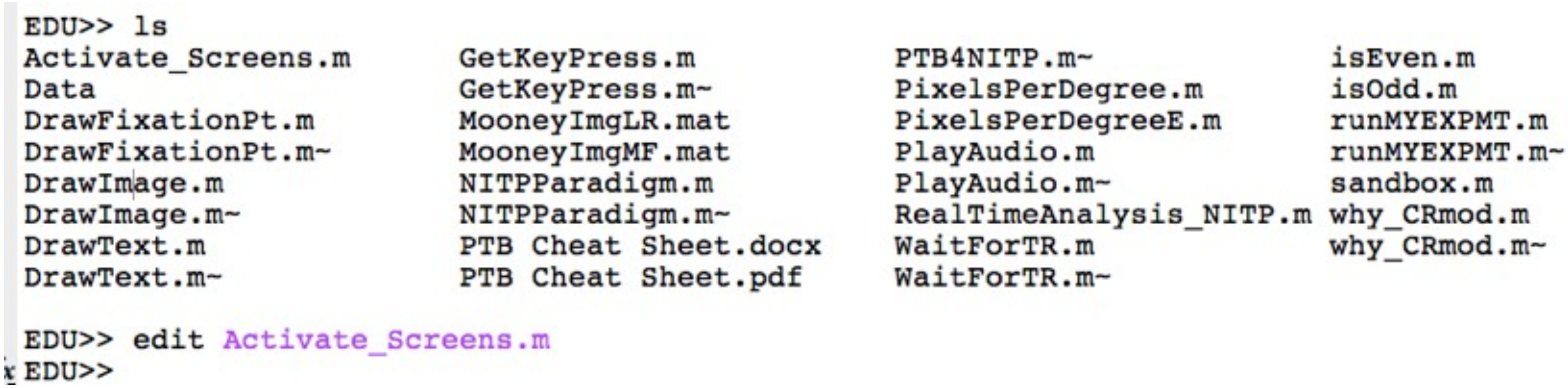
RUN YOUR EXPERIMENT



```
325
326     %% OPEN A SCREEN AND SAVE PARAMETERS FOR LATER USE
327
328 -       [ScreenHandels, Screen_Parameters, PPD_DPP] = ...
329             Activate_Screens(Constants, Params); %<<<<<<<<<<<<<< Step 2 <<<<<<<
330 -       WES = ScreenHandels.WES; % Window Handel Right Screen
331 -       RES = ScreenHandels.RES; % Window Rectangle Right Screen
332 -       WSS = ScreenHandels.WSS; % Window Handel Left Screen
333 -       RSS = ScreenHandels.RSS; % Window Rectangle Left Screen
334
335 -       Params.Screen_Parameters = Screen_Parameters;
336 -       Params.PPD_DPP = PPD_DPP;
337
```

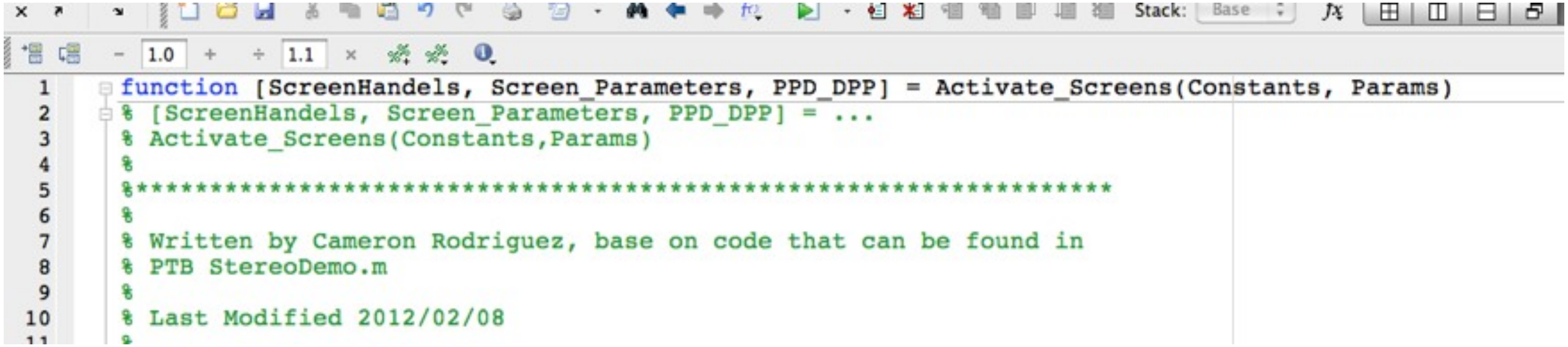
Let's go to ~Line 328 (Step 2)

We see a *function* is called. It's called Activate_Screens.m and takes two parameters (Constants, Params) and spits out 3 variables (left hand side).



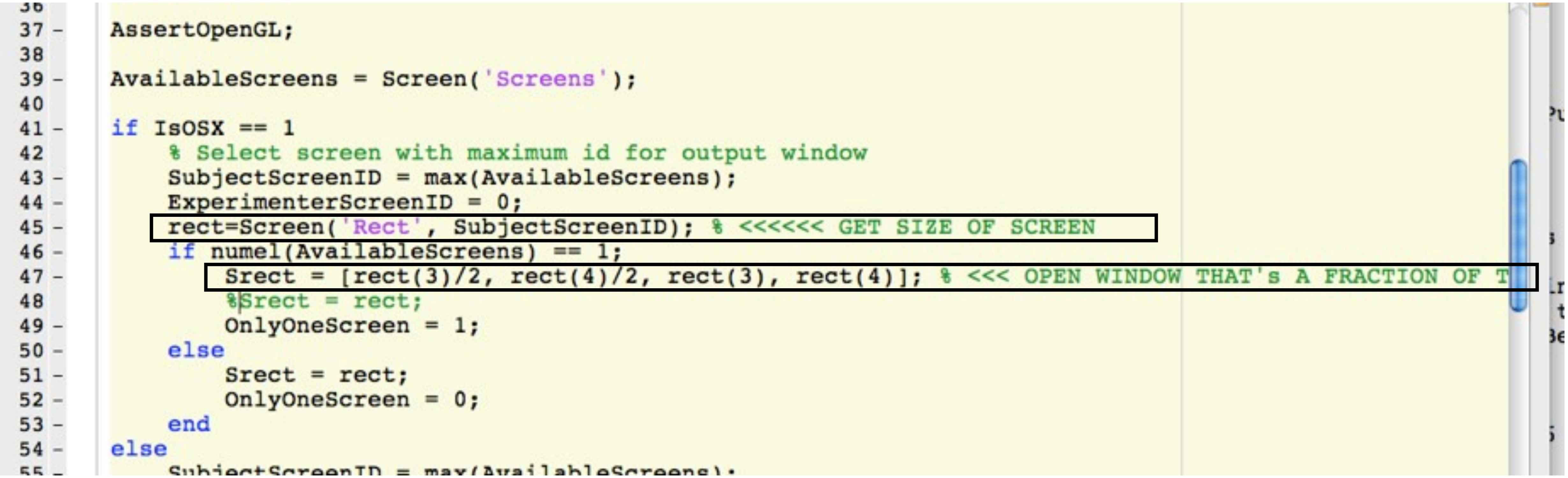
```
EDU>> ls
Activate_Screens.m       GetKeyPress.m            PTB4NITP.m~              isEven.m
Data                     GetKeyPress.m~           PixelsPerDegree.m        isOdd.m
DrawFixationPt.m         MooneyImgLR.mat          PixelsPerDegreeE.m       runMYEXPMT.m
DrawFixationPt.m~        MooneyImgMF.mat          PlayAudio.m              runMYEXPMT.m~
DrawImage.m              NITPParadigm.m           PlayAudio.m~             sandbox.m
DrawImage.m~             NITPParadigm.m~          RealTimeAnalysis_NITP.m  why_CRmod.m
DrawText.m               PTB Cheat Sheet.docx     WaitForTR.m              why_CRmod.m~
DrawText.m~              PTB Cheat Sheet.pdf      WaitForTR.m~

EDU>> edit Activate_Screens.m
EDU>>
```

RUN YOUR EXPERIMENT



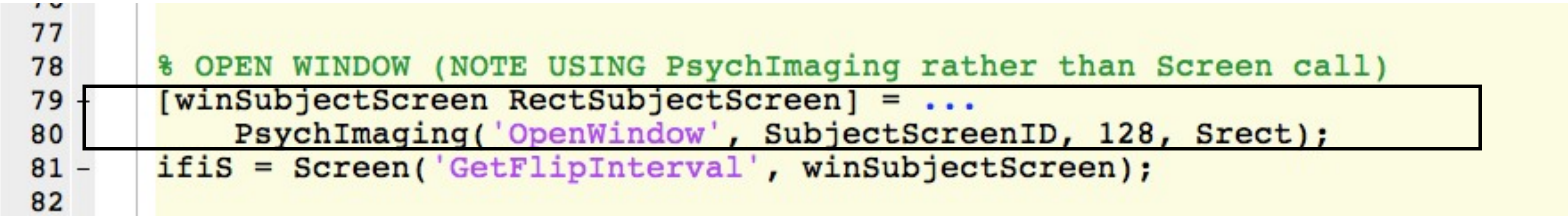
```
function [ScreenHandels, Screen_Parameters, PPD_DPP] = Activate_Screens(Constants, Params)
% [ScreenHandels, Screen_Parameters, PPD_DPP] = ...
% Activate_Screens(Constants,Params)
%
%*****************************************************************
%
% Written by Cameron Rodriguez, base on code that can be found in
% PTB StereoDemo.m
%
% Last Modified 2012/02/08
```

This is a *function* - it takes parameters!



```
37 - AssertOpenGL;
38
39 - AvailableScreens = Screen('Screens');
40
41 - if IsOSX == 1
42        % Select screen with maximum id for output window
43 -      SubjectScreenID = max(AvailableScreens);
44 -      ExperimenterScreenID = 0;
45 -      rect=Screen('Rect', SubjectScreenID); % <<<<<< GET SIZE OF SCREEN
46 -      if numel(AvailableScreens) == 1;
47 -          Srect = [rect(3)/2, rect(4)/2, rect(3), rect(4)]; % <<< OPEN WINDOW THAT's A FRACTION OF T
48            %Srect = rect;
49 -          OnlyOneScreen = 1;
50 -      else
51 -          Srect = rect;
52 -          OnlyOneScreen = 0;
53 -      end
54 - else
55 -      SubjectScreenID = max(AvailableScreens);
```

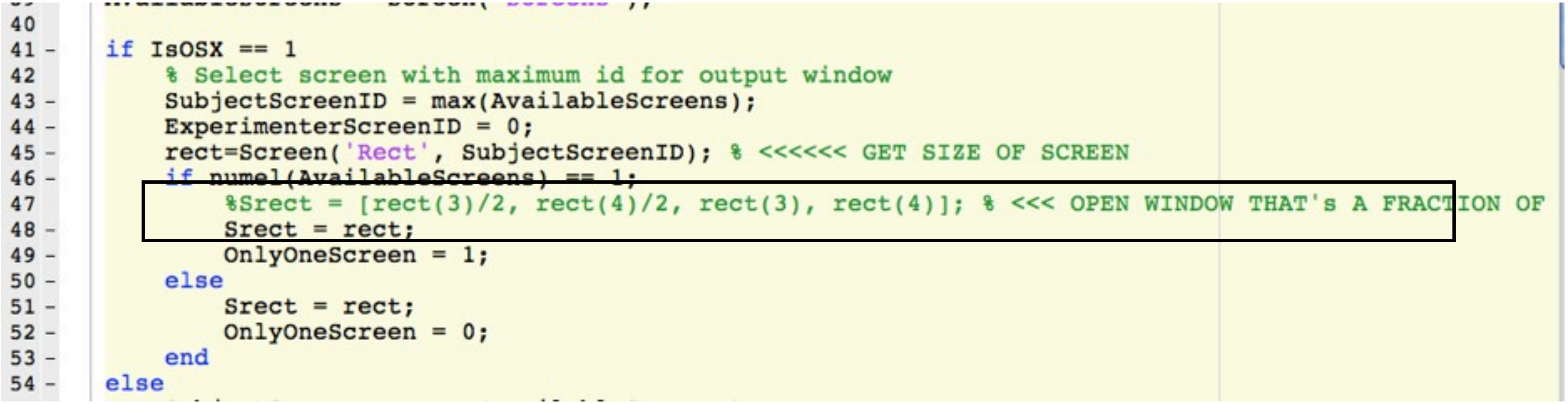
Line 45 gets size of screen. Line 47 calculates dimensions for a fraction of the screen.



```
77
78     % OPEN WINDOW (NOTE USING PsychImaging rather than Screen call)
79 -   [winSubjectScreen RectSubjectScreen] = ...
80         PsychImaging('OpenWindow', SubjectScreenID, 128, Srect);
81 -   ifiS = Screen('GetFlipInterval', winSubjectScreen);
82
```

Line 79 (continued on 80) actually calls a function that will open a window for you. The Srect dimensions that we previously specified are used in this call to specify size. Line 81 (FYI) gets info on the inter-frame-interval of the Screen refresh rate.

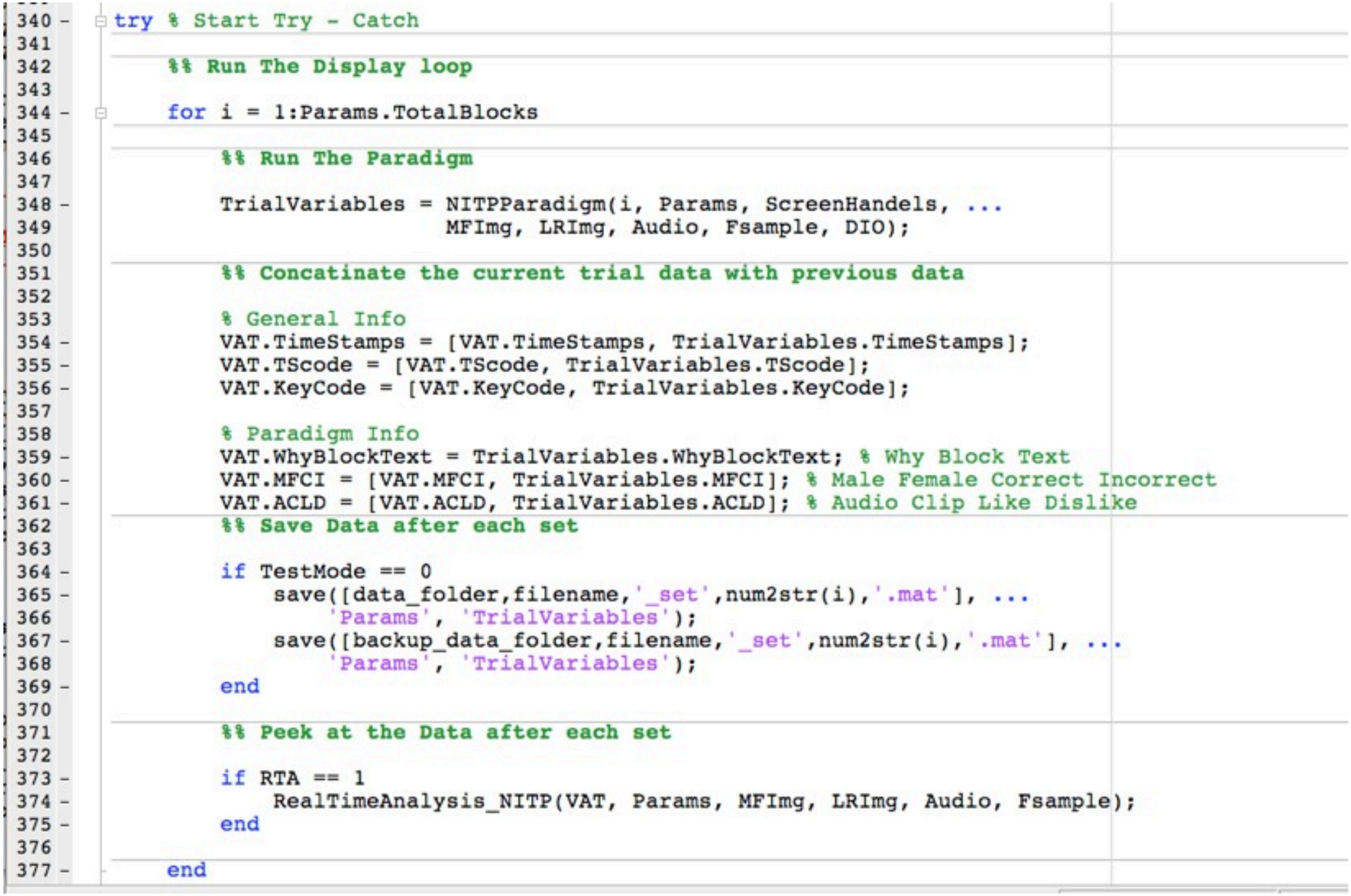
Now - go back up to line 48 and comment out the funny dimensions, setting Srect to rect:



```
40
41 -  if IsOSX == 1
42        % Select screen with maximum id for output window
43 -      SubjectScreenID = max(AvailableScreens);
44 -      ExperimenterScreenID = 0;
45 -      rect=Screen('Rect', SubjectScreenID); % <<<<<< GET SIZE OF SCREEN
46 -      if numel(AvailableScreens) == 1;
47            %Srect = [rect(3)/2, rect(4)/2, rect(3), rect(4)]; % <<< OPEN WINDOW THAT's A FRACTION OF
48 -          Srect = rect;
49 -          OnlyOneScreen = 1;
50 -      else
51 -          Srect = rect;
52 -          OnlyOneScreen = 0;
53 -      end
54 -  else
```

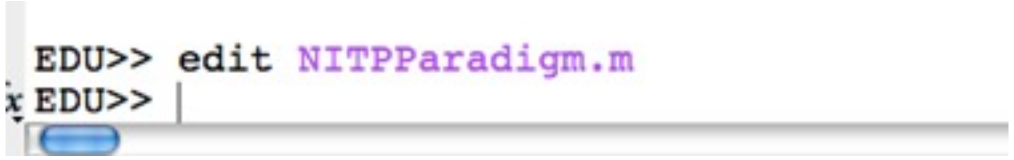
Save file and run the script (runMYEXPMT.m) again... you should now see a full window display. You can leave this as is, or -- for debugging -- changing this setting back to a mini-window.

The experimental guts starts at line 344 - where you see a loop which, for each block, called a function called NITPParadigm, does some stuff, and then saves a bunch of data.

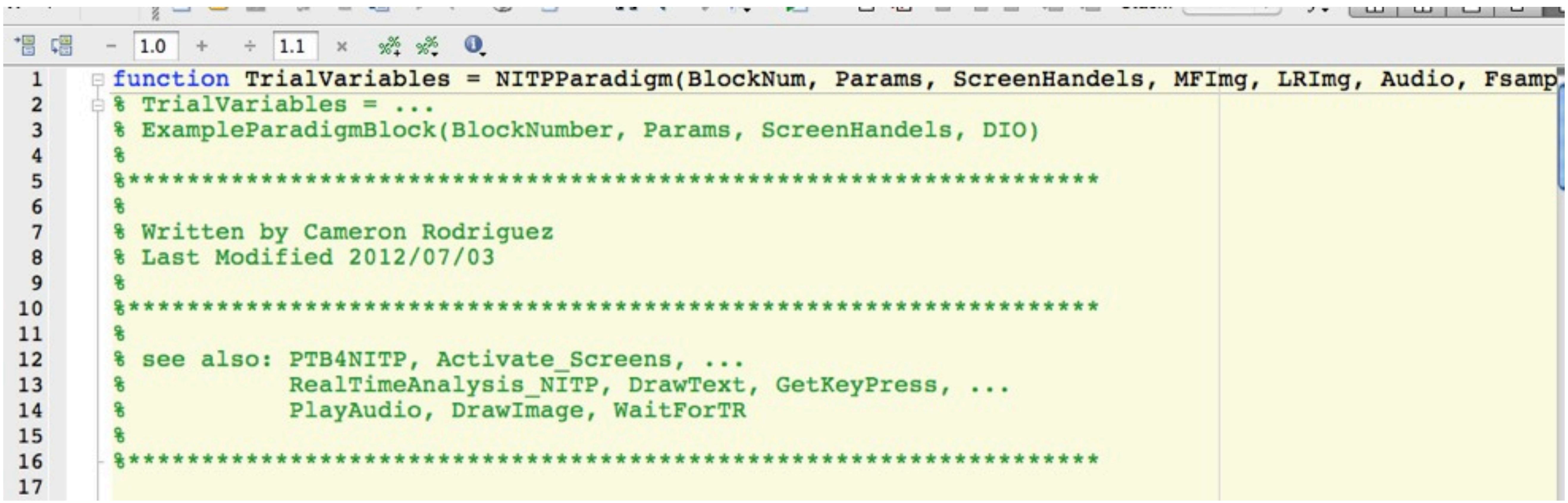


```
340 -  try % Start Try - Catch
341
342        %% Run The Display loop
343
344 -      for i = 1:Params.TotalBlocks
345
346            %% Run The Paradigm
347
348 -          TrialVariables = NITPParadigm(i, Params, ScreenHandels, ...
349                                  MFImg, LRImg, Audio, Fsample, DIO);
350
351            %% Concatinate the current trial data with previous data
352
353            % General Info
354 -          VAT.TimeStamps = [VAT.TimeStamps, TrialVariables.TimeStamps];
355 -          VAT.TScode = [VAT.TScode, TrialVariables.TScode];
356 -          VAT.KeyCode = [VAT.KeyCode, TrialVariables.KeyCode];
357
358            % Paradigm Info
359 -          VAT.WhyBlockText = TrialVariables.WhyBlockText; % Why Block Text
360 -          VAT.MFCI = [VAT.MFCI, TrialVariables.MFCI]; % Male Female Correct Incorrect
361 -          VAT.ACLD = [VAT.ACLD, TrialVariables.ACLD]; % Audio Clip Like Dislike
362            %% Save Data after each set
363
364 -          if TestMode == 0
365 -              save([data_folder,filename,'_set',num2str(i),'.mat'], ...
366                        'Params', 'TrialVariables');
367 -              save([backup_data_folder,filename,'_set',num2str(i),'.mat'], ...
368                        'Params', 'TrialVariables');
369 -          end
370
371            %% Peek at the Data after each set
372
373 -          if RTA == 1
374 -              RealTimeAnalysis_NITP(VAT, Params, MFImg, LRImg, Audio, Fsample);
375 -          end
376
377 -      end
```

Let’s open NITPparadigm.m and mess with the displays.



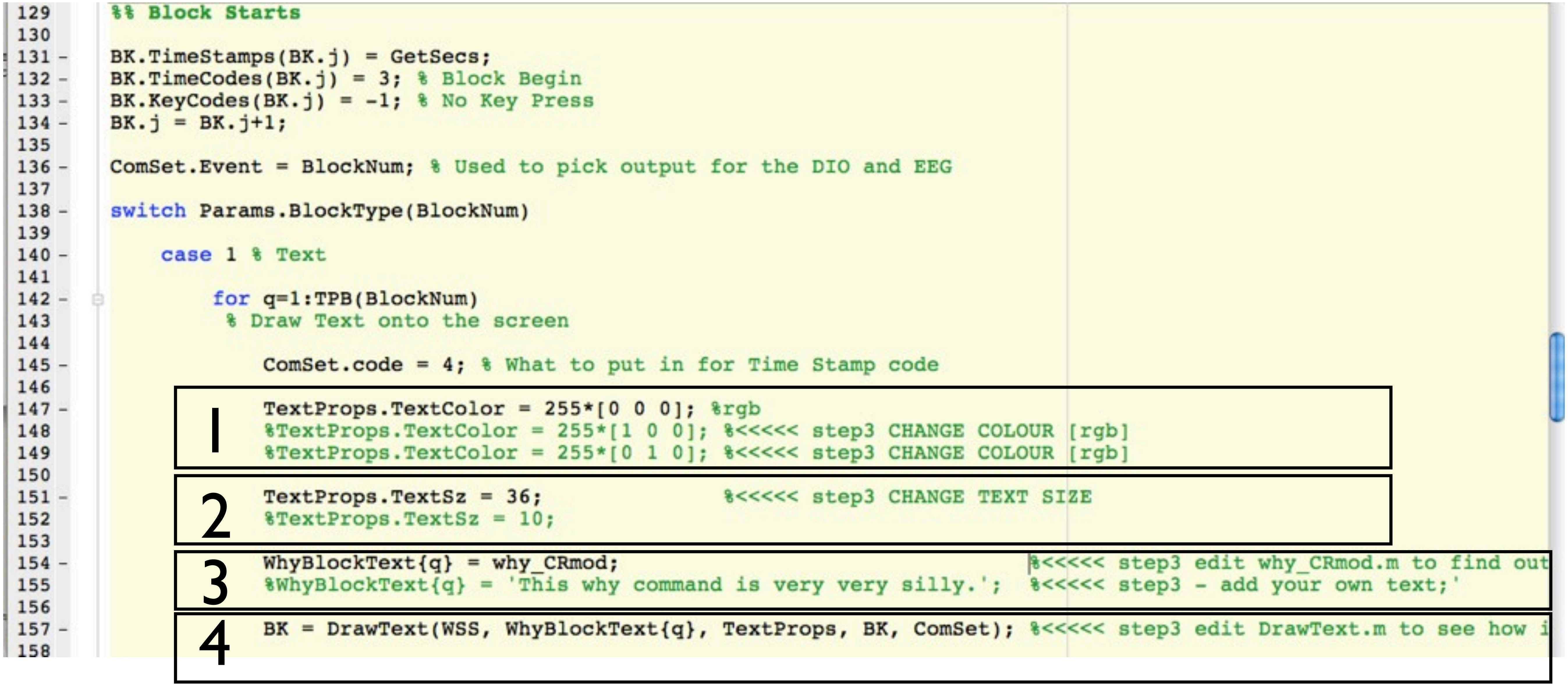
```
EDU>> edit NITPParadigm.m
EDU>> |
```



```
function TrialVariables = NITPParadigm(BlockNum, Params, ScreenHandels, MFImg, LRImg, Audio, Fsamp
% TrialVariables = ...
% ExampleParadigmBlock(BlockNumber, Params, ScreenHandels, DIO)
%
%*******************************************************************
%
% Written by Cameron Rodriguez
% Last Modified 2012/07/03
%
%*******************************************************************
%
% see also: PTB4NITP, Activate_Screens, ...
%           RealTimeAnalysis_NITP, DrawText, GetKeyPress, ...
%           PlayAudio, DrawImage, WaitForTR
%
%*******************************************************************

```

Step 3 <<< explore Text

go to ~line 130 where block display loop starts starts



```
129      %% Block Starts
130
131 -    BK.TimeStamps(BK.j) = GetSecs;
132 -    BK.TimeCodes(BK.j) = 3; % Block Begin
133 -    BK.KeyCodes(BK.j) = -1; % No Key Press
134 -    BK.j = BK.j+1;
135
136 -    ComSet.Event = BlockNum; % Used to pick output for the DIO and EEG
137
138 -    switch Params.BlockType(BlockNum)
139
140 -        case 1 % Text
141
142 -            for q=1:TPB(BlockNum)
143               % Draw Text onto the screen
144
145 -                ComSet.code = 4; % What to put in for Time Stamp code
146
147 -                TextProps.TextColor = 255*[0 0 0]; %rgb
148                  %TextProps.TextColor = 255*[1 0 0]; %<<<<< step3 CHANGE COLOUR [rgb]
149                  %TextProps.TextColor = 255*[0 1 0]; %<<<<< step3 CHANGE COLOUR [rgb]
150
151 -                TextProps.TextSz = 36;              %<<<<< step3 CHANGE TEXT SIZE
152                  %TextProps.TextSz = 10;
153
154 -                WhyBlockText{q} = why_CRmod;                             %<<<<< step3 edit why_CRmod.m to find out
155                  %WhyBlockText{q} = 'This why command is very very silly.';  %<<<<< step3 - add your own text;'
156
157 -                BK = DrawText(WSS, WhyBlockText{q}, TextProps, BK, ComSet); %<<<<< step3 edit DrawText.m to see how i
158
```

1. try to change color, save, run

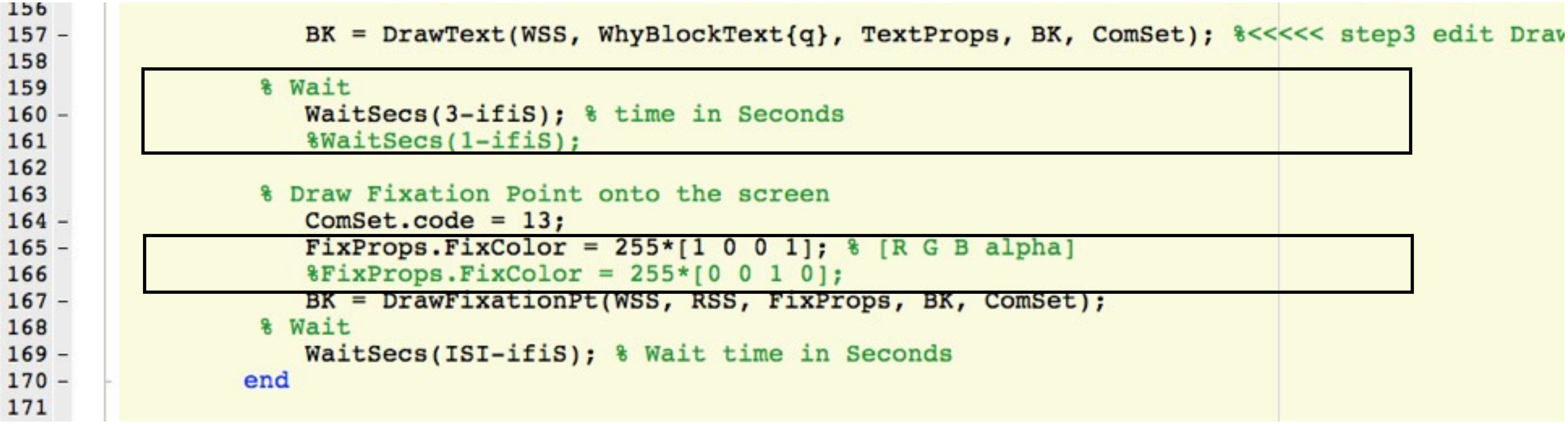
2. try change size, save, run

3a. try to change content, save, run (edit why_CRmod.m to see how it works)

3b. type 'why' at the command prompt and press return (it calls why_CRmod.m)

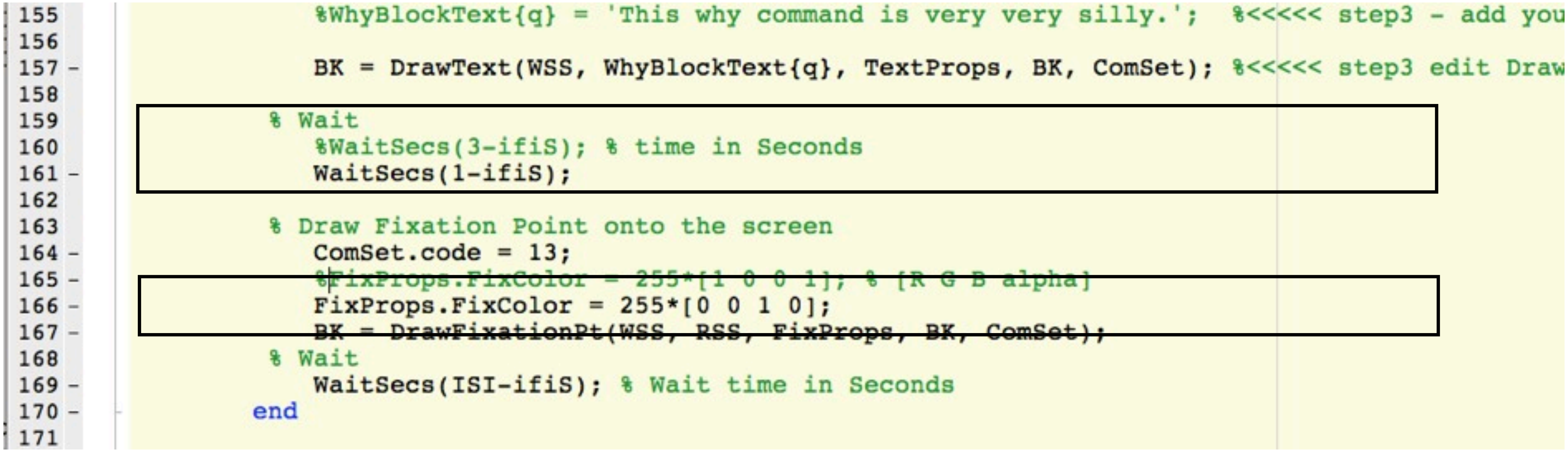
4. edit Cameron's DrawText.m function to see how he uses PTB to draw text (hint - it's drawn using DrawFormattedText PTB command at line 50).

what else happens after text? Waiting period and fixation point.



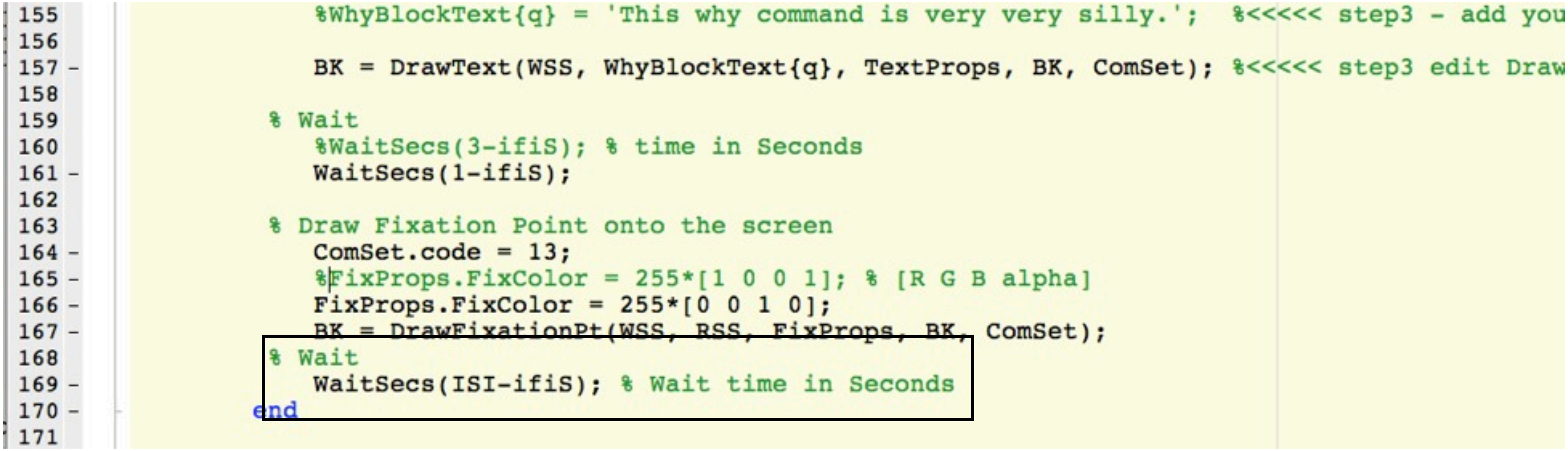
```
      BK = DrawText(WSS, WhyBlockText{q}, TextProps, BK, ComSet); %<<<<< step3 edit Draw

     % Wait
      WaitSecs(3-ifiS); % time in Seconds
      %WaitSecs(1-ifiS);

     % Draw Fixation Point onto the screen
      ComSet.code = 13;
      FixProps.FixColor = 255*[1 0 0 1]; % [R G B alpha]
      %FixProps.FixColor = 255*[0 0 1 0];
      BK = DrawFixationPt(WSS, RSS, FixProps, BK, ComSet);
     % Wait
      WaitSecs(ISI-ifiS); % Wait time in Seconds
    end
```

how would you change the inter-stimulus interval?
how would you change the Fixation color?
check out Cameron’s DrawFixationPt.m to see how he uses PTB to draw a fixation point.



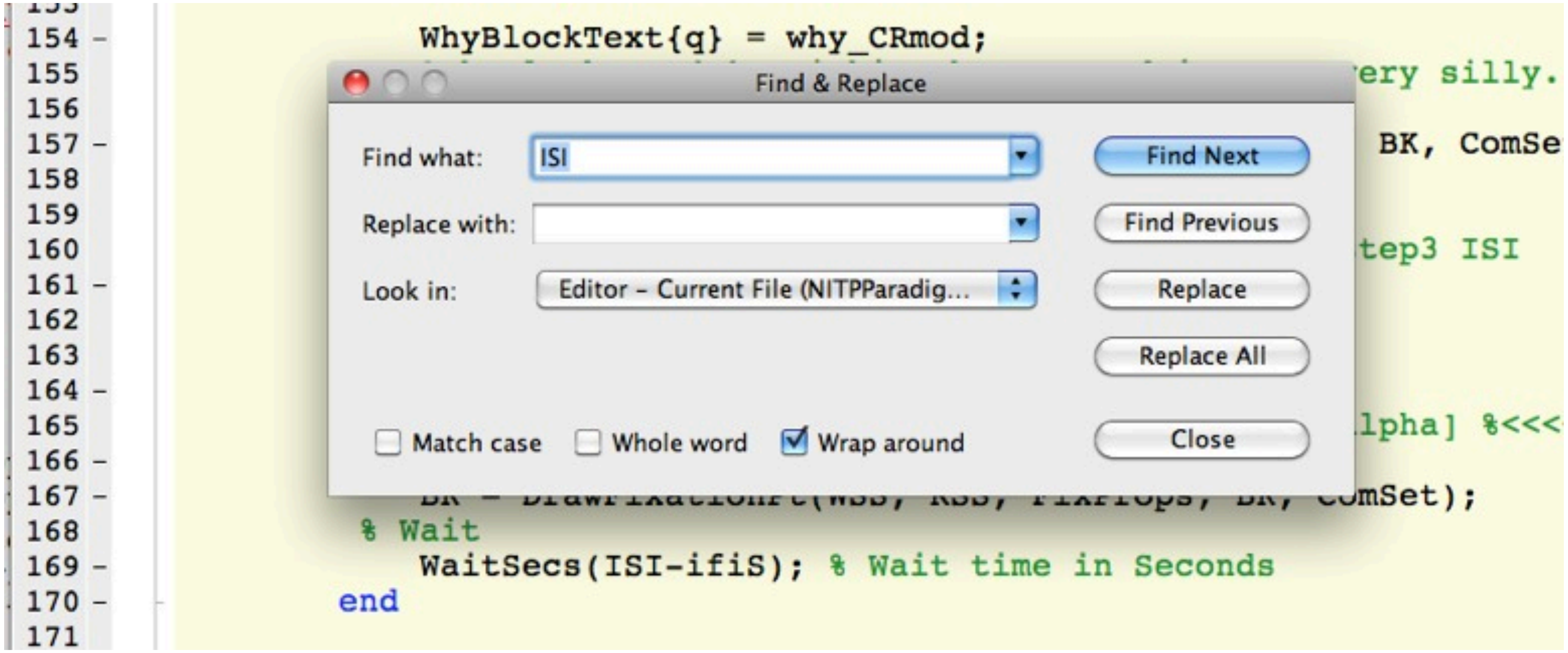
```
      %WhyBlockText{q} = 'This why command is very very silly.';  %<<<<< step3 - add you

      BK = DrawText(WSS, WhyBlockText{q}, TextProps, BK, ComSet); %<<<<< step3 edit Draw

     % Wait
      %WaitSecs(3-ifiS); % time in Seconds
      WaitSecs(1-ifiS);

     % Draw Fixation Point onto the screen
      ComSet.code = 13;
      %FixProps.FixColor = 255*[1 0 0 1]; % [R G B alpha]
      FixProps.FixColor = 255*[0 0 1 0];
      BK = DrawFixationPt(WSS, RSS, FixProps, BK, ComSet);
     % Wait
      WaitSecs(ISI-ifiS); % Wait time in Seconds
    end
```

but look - there is another WaitSecs call and it slows everything down... but what's ISI



```
            %WhyBlockText{q} = 'This why command is very very silly.';  %<<<<< step3 - add you
            BK = DrawText(WSS, WhyBlockText{q}, TextProps, BK, ComSet); %<<<<< step3 edit Draw

        % Wait
            %WaitSecs(3-ifiS); % time in Seconds
            WaitSecs(1-ifiS);

        % Draw Fixation Point onto the screen
            ComSet.code = 13;
            %FixProps.FixColor = 255*[1 0 0 1]; % [R G B alpha]
            FixProps.FixColor = 255*[0 0 1 0];
            BK = DrawFixationPt(WSS, RSS, FixProps, BK, ComSet);
        % Wait
            WaitSecs(ISI-ifiS); % Wait time in Seconds
        end
```

do a search to find ISI in the code



```
            WhyBlockText{q} = why_CRmod;
                                                       ery silly.
                                                      BK, ComSe

                                                     tep3 ISI



                                                     lpha] %<<<

                                                     omSet);
        % Wait
            WaitSecs(ISI-ifiS); % Wait time in Seconds
        end
```

Find & Replace

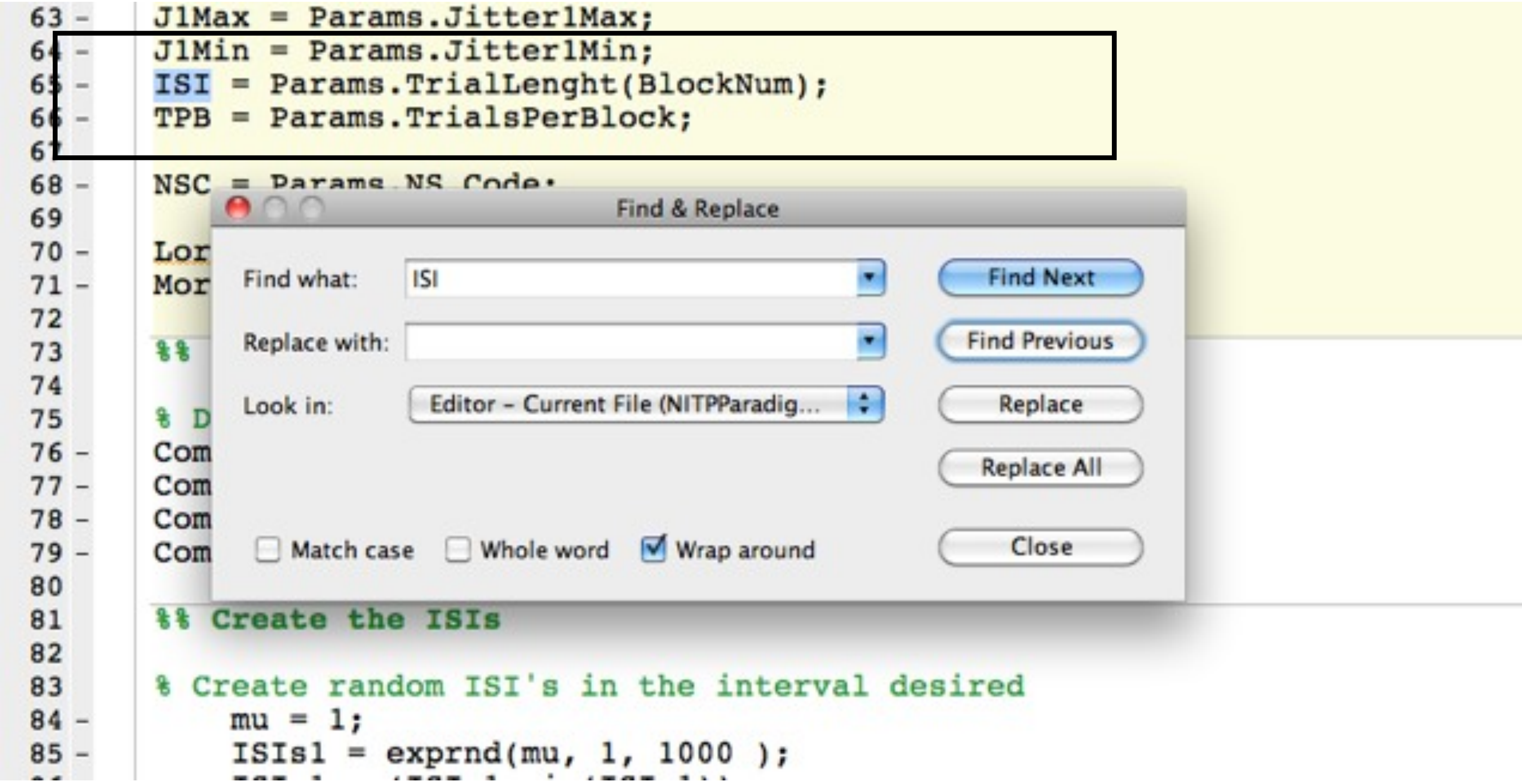
Find what: ISI

Replace with:

Look in: Editor - Current File (NITPParadig...

☐ Match case ☐ Whole word ☑ Wrap around

that'll take you to line 65 and here you see that the ISI is obtained from a Params variable



```
63 -  J1Max = Params.Jitter1Max;
64 -  J1Min = Params.Jitter1Min;
65 -  ISI = Params.TrialLenght(BlockNum);
66 -  TPB = Params.TrialsPerBlock;
67
68 -  NSC = Params.NS_Code;
69
70 -  Lor
71 -  Mor
72
73    %%
74
75    % D
76 -  Com
77 -  Com
78 -  Com
79 -  Com
80
81    %% Create the ISIs
82
83    % Create random ISI's in the interval desired
84 -      mu = 1;
85 -      ISIs1 = exprnd(mu, 1, 1000 );
```

Find & Replace

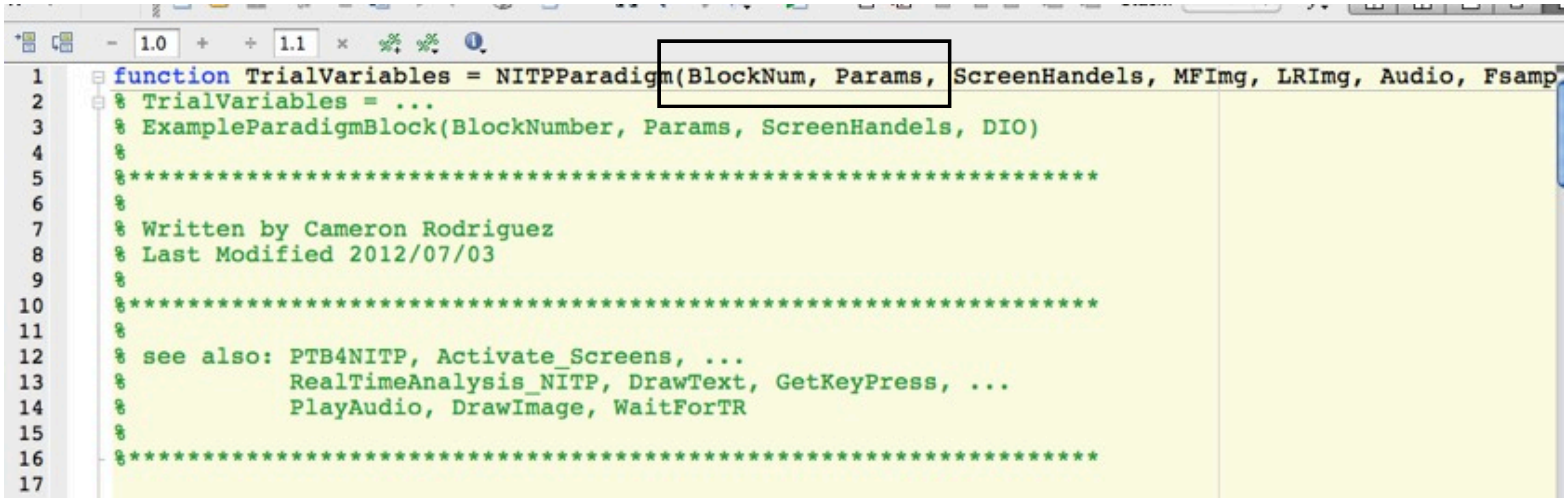
Find what: ISI

Replace with:

Look in: Editor - Current File (NITPParadig...

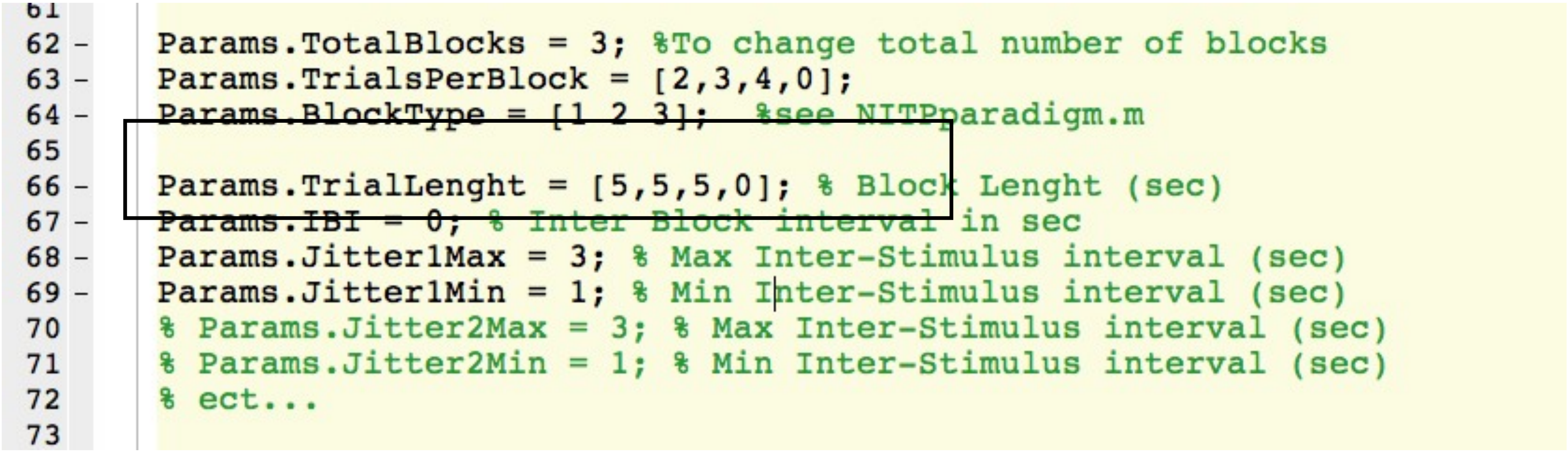
☐ Match case ☐ Whole word ☑ Wrap around

which btw you passed into this function in the inputs



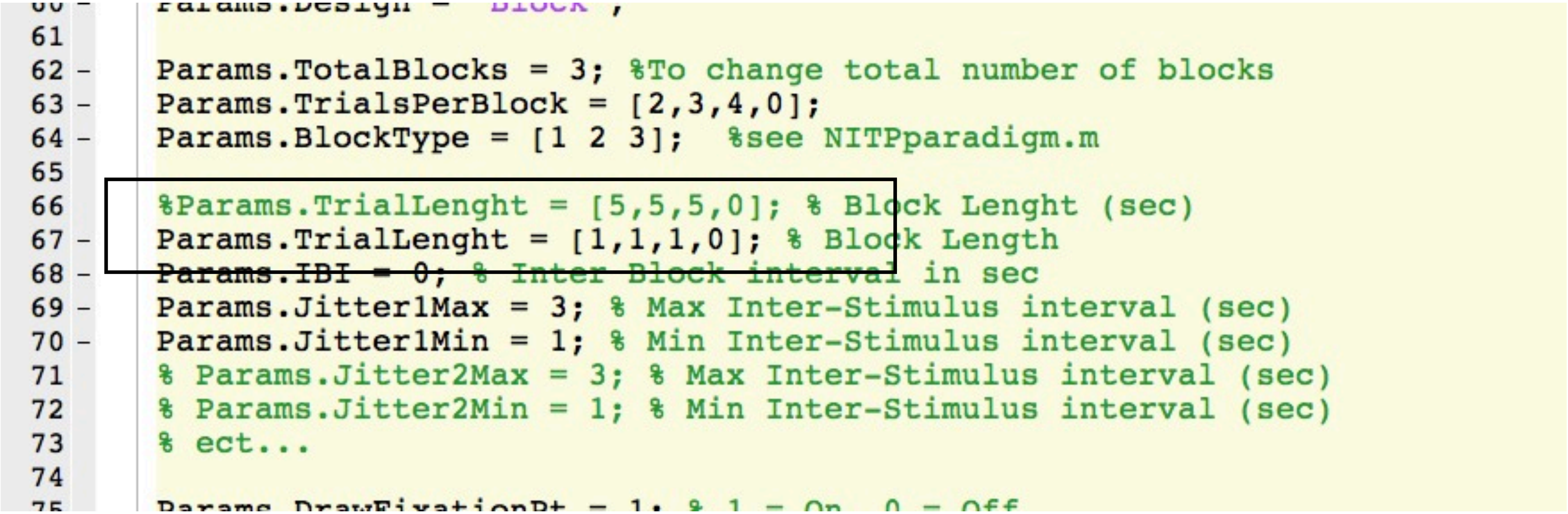
```
function TrialVariables = NITPParadigm(BlockNum, Params, ScreenHandels, MFImg, LRImg, Audio, Fsamp
% TrialVariables = ...
% ExampleParadigmBlock(BlockNumber, Params, ScreenHandels, DIO)
%
%*******************************************************************
%
% Written by Cameron Rodriguez
% Last Modified 2012/07/03
%
%*******************************************************************
%
% see also: PTB4NITP, Activate_Screens, ...
%           RealTimeAnalysis_NITP, DrawText, GetKeyPress, ...
%           PlayAudio, DrawImage, WaitForTR
%
%*******************************************************************
```

so head on out ...back to runMYEXPMT.m ..around line 66



```
61
62 -   Params.TotalBlocks = 3; %To change total number of blocks
63 -   Params.TrialsPerBlock = [2,3,4,0];
64 -   Params.BlockType = [1 2 3];   %see NITPparadigm.m
65
66 -   Params.TrialLenght = [5,5,5,0]; % Block Lenght (sec)
67 -   Params.IBI = 0; % Inter Block interval in sec
68 -   Params.Jitter1Max = 3; % Max Inter-Stimulus interval (sec)
69 -   Params.Jitter1Min = 1; % Min Inter-Stimulus interval (sec)
70      % Params.Jitter2Max = 3; % Max Inter-Stimulus interval (sec)
71      % Params.Jitter2Min = 1; % Min Inter-Stimulus interval (sec)
72      % ect...
73
```

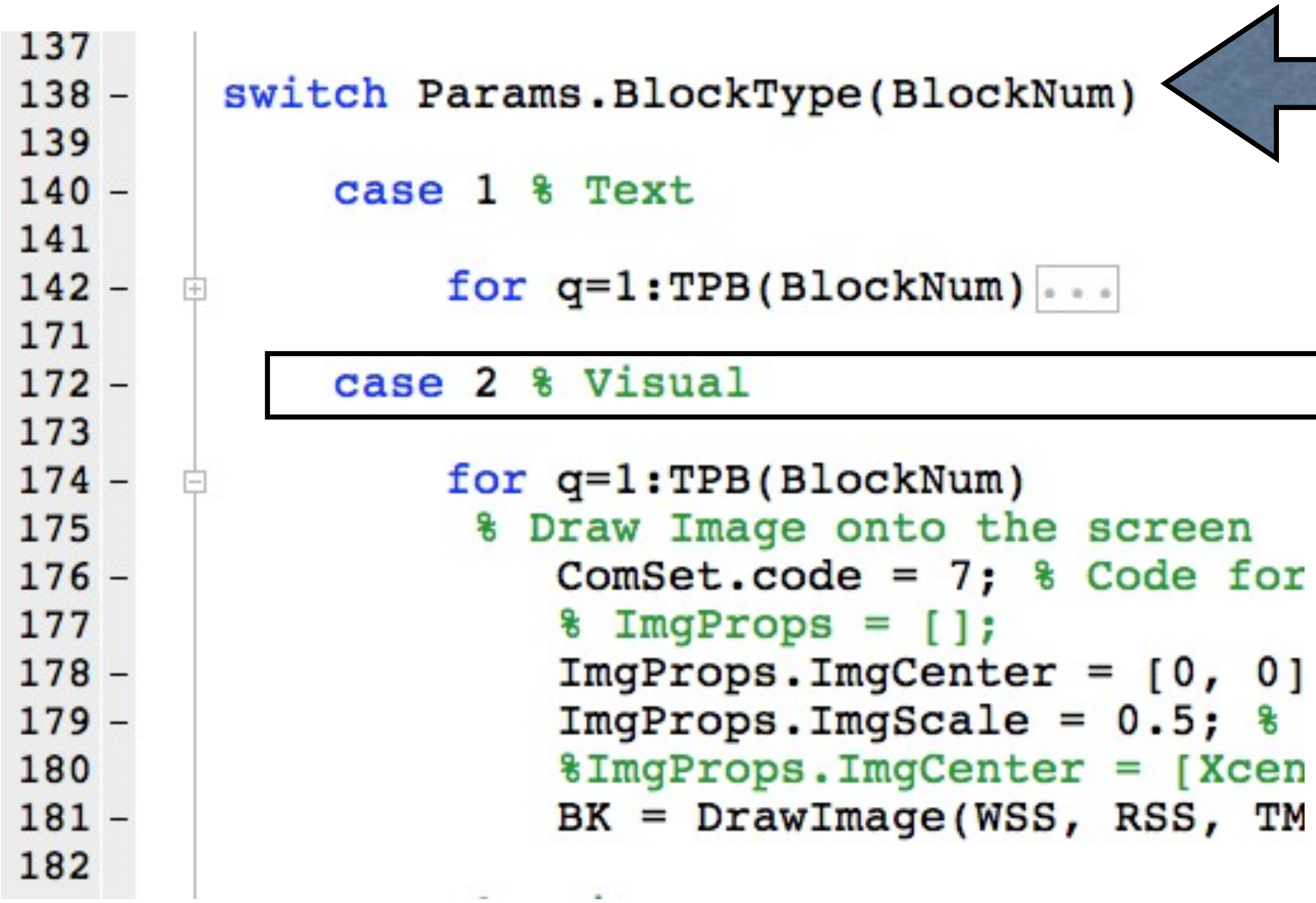
and change the block length (which is the trial length) to 1 sec



```
60 -   Params.Design = 'Block';
61
62 -   Params.TotalBlocks = 3; %To change total number of blocks
63 -   Params.TrialsPerBlock = [2,3,4,0];
64 -   Params.BlockType = [1 2 3];   %see NITPparadigm.m
65
66      %Params.TrialLenght = [5,5,5,0]; % Block Lenght (sec)
67 -   Params.TrialLenght = [1,1,1,0]; % Block Length
68 -   Params.IBI = 0; % Inter Block interval in sec
69 -   Params.Jitter1Max = 3; % Max Inter-Stimulus interval (sec)
70 -   Params.Jitter1Min = 1; % Min Inter-Stimulus interval (sec)
71      % Params.Jitter2Max = 3; % Max Inter-Stimulus interval (sec)
72      % Params.Jitter2Min = 1; % Min Inter-Stimulus interval (sec)
73      % ect...
74
75 -   Params.DrawFixationPt = 1; % 1 = On, 0 = Off
```

save and run the experiment.

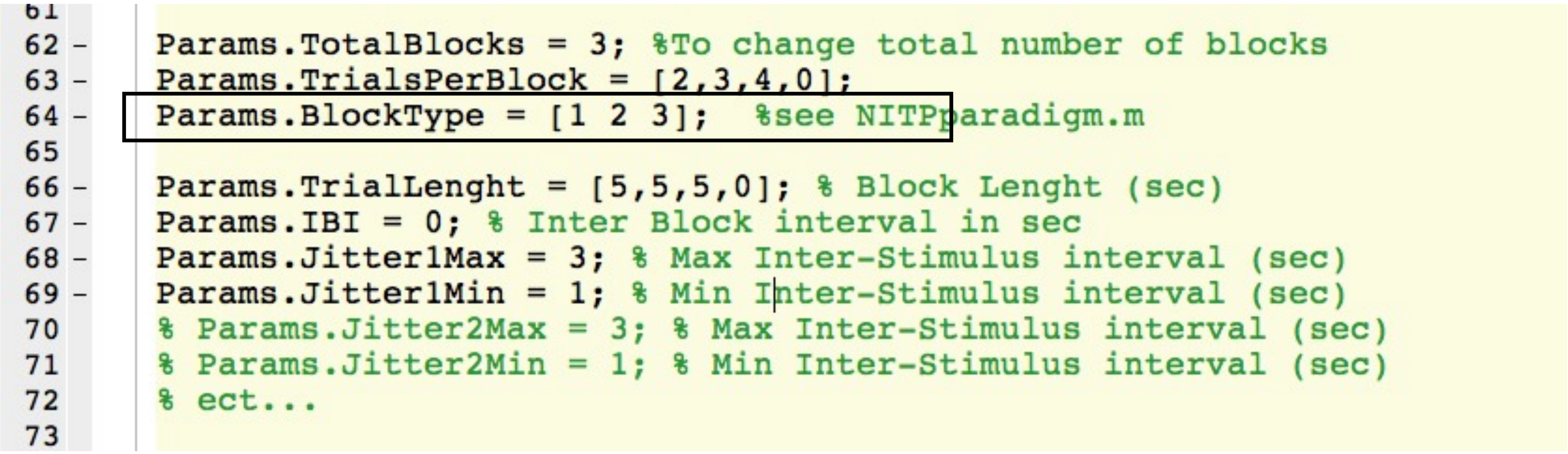
step 4 <<<< explore visual



in NITPparadigm.m

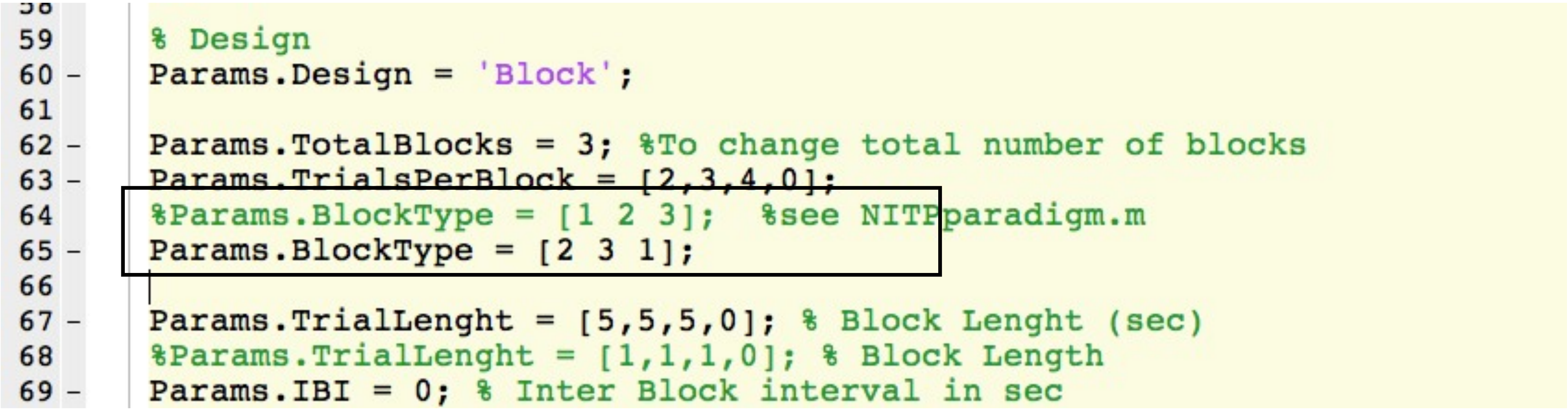
```
137
138 -  switch Params.BlockType(BlockNum)
139
140 -      case 1 % Text
141
142 -          for q=1:TPB(BlockNum)...
171
172 -      case 2 % Visual
173
174 -          for q=1:TPB(BlockNum)
175             % Draw Image onto the screen
176 -               ComSet.code = 7; % Code for
177                 % ImgProps = [];
178 -               ImgProps.ImgCenter = [0, 0]
179 -               ImgProps.ImgScale = 0.5; %
180                 %ImgProps.ImgCenter = [Xcen
181 -               BK = DrawImage(WSS, RSS, TM
182
```

What you see here is that the code will present visual stimuli if case is 2
Which block is that? Go back to runMYEXPMT.m ~line 64 to find:



```
61
62 -  Params.TotalBlocks = 3; %To change total number of blocks
63 -  Params.TrialsPerBlock = [2,3,4,0];
64 -  Params.BlockType = [1 2 3];   %see NITPparadigm.m
65
66 -  Params.TrialLenght = [5,5,5,0]; % Block Lenght (sec)
67 -  Params.IBI = 0; % Inter Block interval in sec
68 -  Params.Jitter1Max = 3; % Max Inter-Stimulus interval (sec)
69 -  Params.Jitter1Min = 1; % Min Inter-Stimulus interval (sec)
70    % Params.Jitter2Max = 3; % Max Inter-Stimulus interval (sec)
71    % Params.Jitter2Min = 1; % Min Inter-Stimulus interval (sec)
72    % ect...
73
```

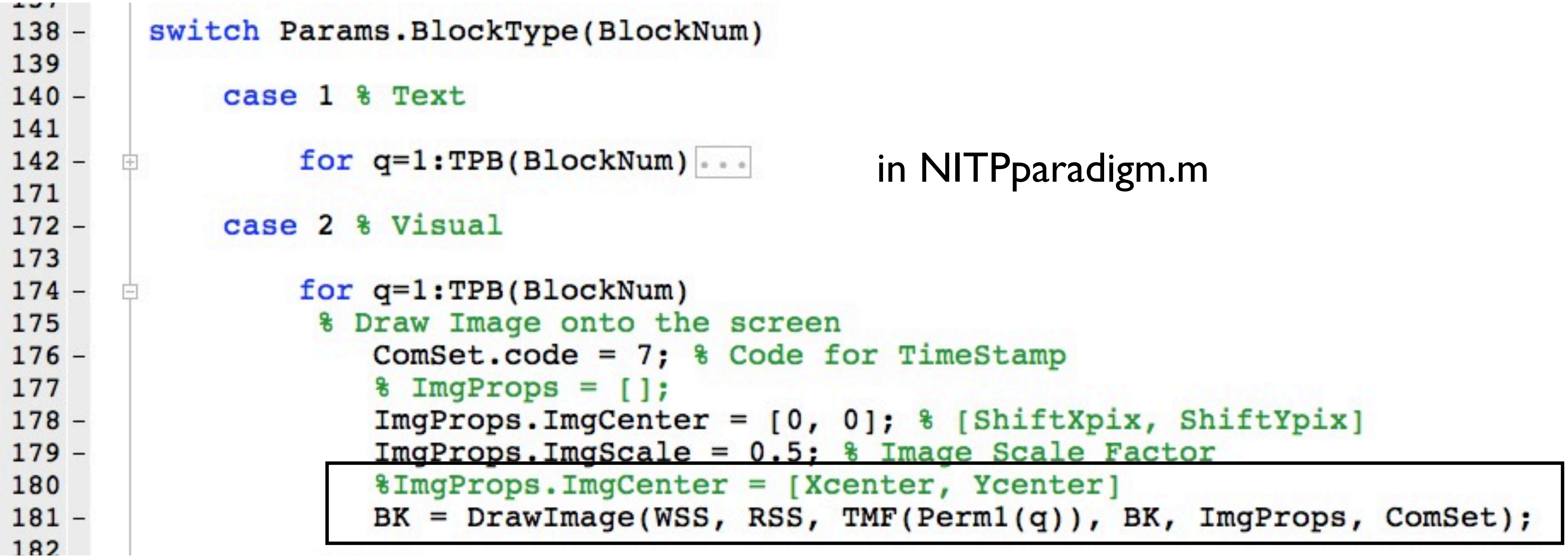
If you change BlockType - you'll change which stimuli are shown during which block. I am impatient so I'd like to move up the visual stimuli to block 1. I will put a '2' in the block one position in the BlockType matrix. This will make sure that case '2' occurs first.



```
% Design
Params.Design = 'Block';

Params.TotalBlocks = 3; %To change total number of blocks
Params.TrialsPerBlock = [2,3,4,0];
%Params.BlockType = [1 2 3];   %see NITPparadigm.m
Params.BlockType = [2 3 1];

Params.TrialLenght = [5,5,5,0]; % Block Lenght (sec)
%Params.TrialLenght = [1,1,1,0]; % Block Length
Params.IBI = 0; % Inter Block interval in sec
```

Rerun the experiment - are you still seeing text stimuli?

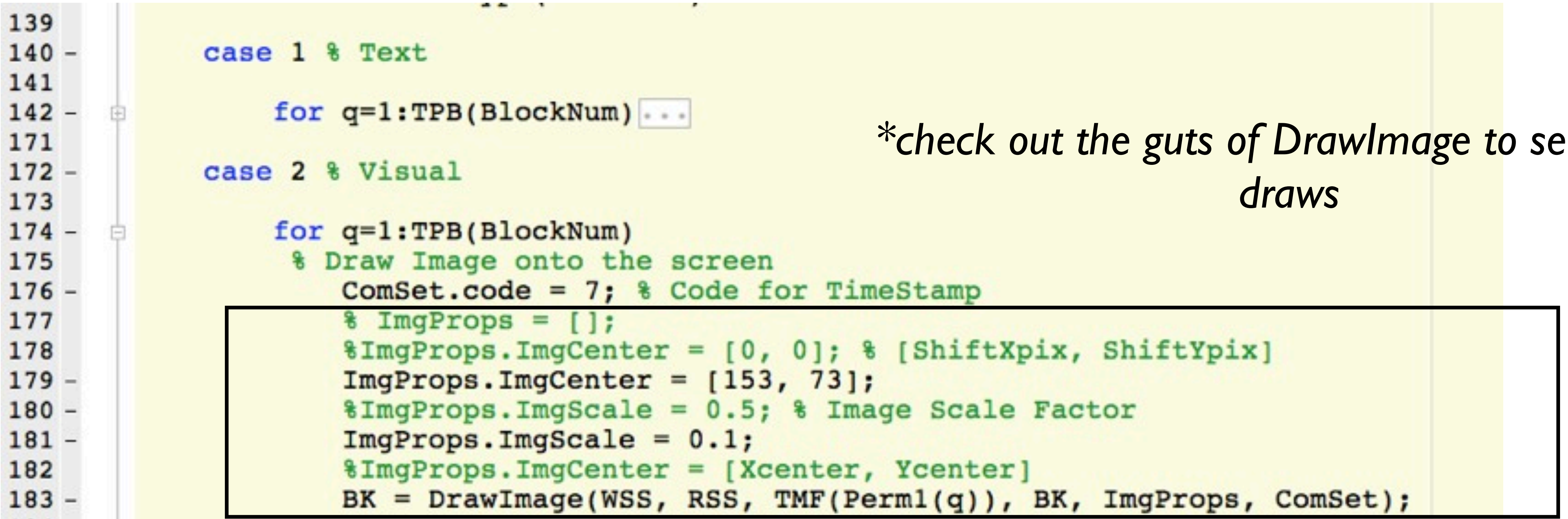
OK ready to see how images are drawn?



```
switch Params.BlockType(BlockNum)

    case 1 % Text

        for q=1:TPB(BlockNum) ...

    case 2 % Visual

        for q=1:TPB(BlockNum)
         % Draw Image onto the screen
            ComSet.code = 7; % Code for TimeStamp
            % ImgProps = [];
            ImgProps.ImgCenter = [0, 0]; % [ShiftXpix, ShiftYpix]
            ImgProps.ImgScale = 0.5; % Image Scale Factor
            %ImgProps.ImgCenter = [Xcenter, Ycenter]
            BK = DrawImage(WSS, RSS, TMF(Perm1(q)), BK, ImgProps, ComSet);
```

in NITPparadigm.m

Try to change location and size of image by changing the ImgProps parameters!

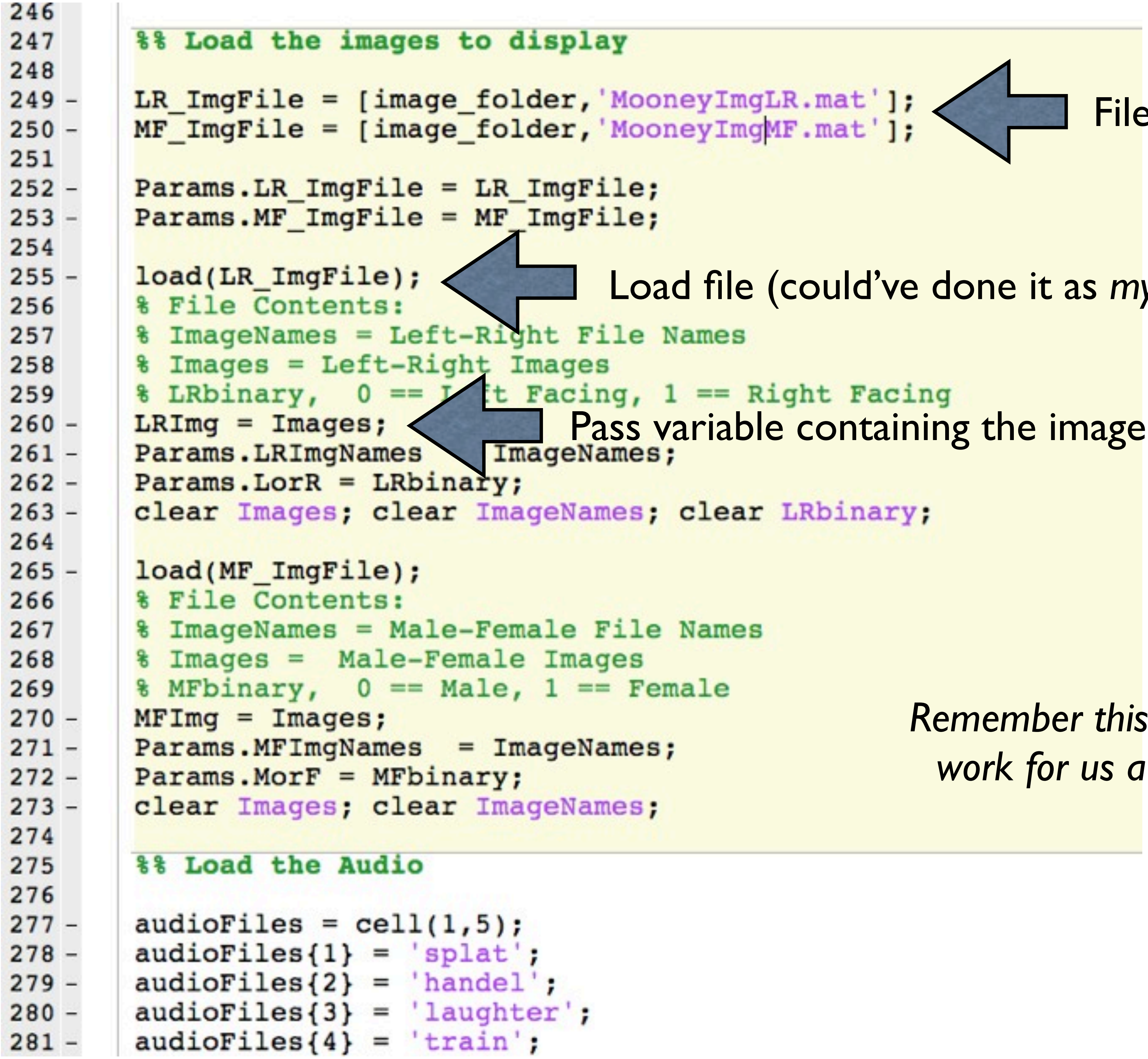


```

    case 1 % Text

        for q=1:TPB(BlockNum) ...

    case 2 % Visual

        for q=1:TPB(BlockNum)
         % Draw Image onto the screen
            ComSet.code = 7; % Code for TimeStamp
            % ImgProps = [];
            %ImgProps.ImgCenter = [0, 0]; % [ShiftXpix, ShiftYpix]
            ImgProps.ImgCenter = [153, 73];
            %ImgProps.ImgScale = 0.5; % Image Scale Factor
            ImgProps.ImgScale = 0.1;
            %ImgProps.ImgCenter = [Xcenter, Ycenter]
            BK = DrawImage(WSS, RSS, TMF(Perm1(q)), BK, ImgProps, ComSet);
```

**check out the guts of DrawImage to see how it draws*

Three important steps to showing images this way. First - images are actually preloaded.

...in runEXPMT.m



```
%% Load the images to display

LR_ImgFile = [image_folder,'MooneyImgLR.mat'];
MF_ImgFile = [image_folder,'MooneyImgMF.mat'];

Params.LR_ImgFile = LR_ImgFile;
Params.MF_ImgFile = MF_ImgFile;

load(LR_ImgFile);
% File Contents:
% ImageNames = Left-Right File Names
% Images = Left-Right Images
% LRbinary,  0 == Left Facing, 1 == Right Facing
LRImg = Images;
Params.LRImgNames  = ImageNames;
Params.LorR = LRbinary;
clear Images; clear ImageNames; clear LRbinary;

load(MF_ImgFile);
% File Contents:
% ImageNames = Male-Female File Names
% Images =  Male-Female Images
% MFbinary,  0 == Male, 1 == Female
MFImg = Images;
Params.MFImgNames  = ImageNames;
Params.MorF = MFbinary;
clear Images; clear ImageNames;

%% Load the Audio

audioFiles = cell(1,5);
audioFiles{1} = 'splat';
audioFiles{2} = 'handel';
audioFiles{3} = 'laughter';
audioFiles{4} = 'train';
```

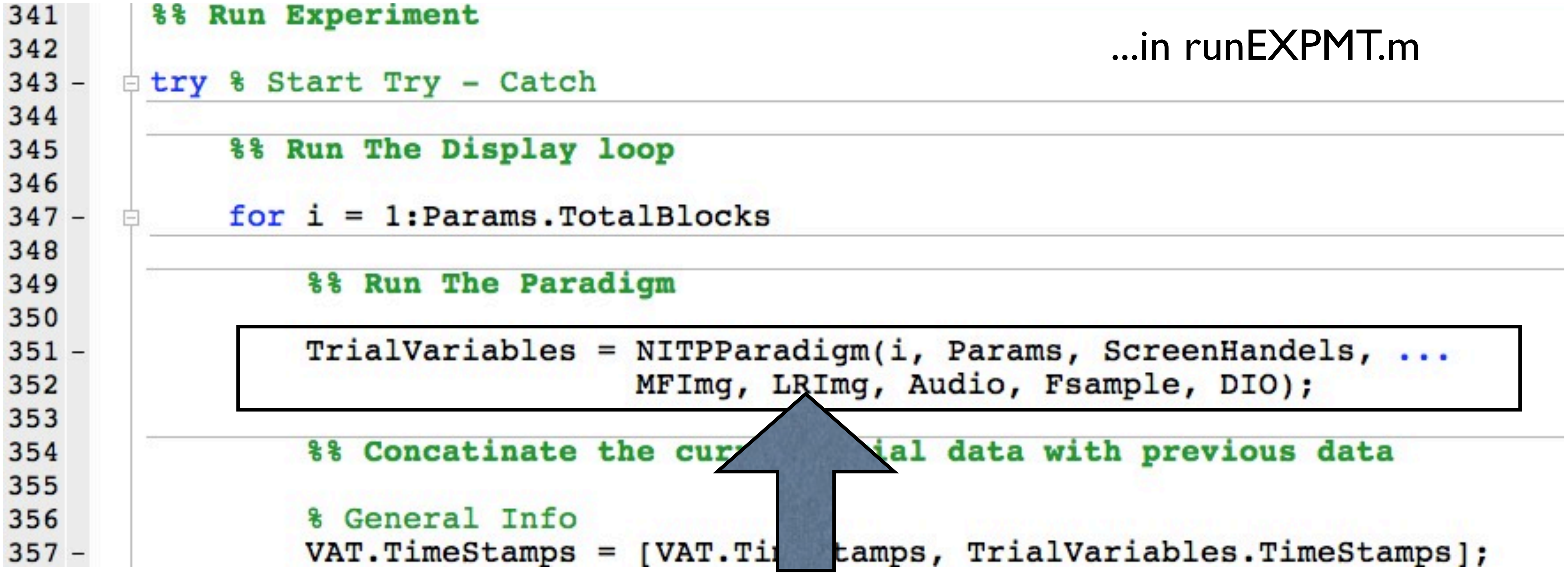
File name

Load file (could've done it as *myfile* = *load(LR_ImgFile)*);

Pass variable containing the image(s) to LRImg.

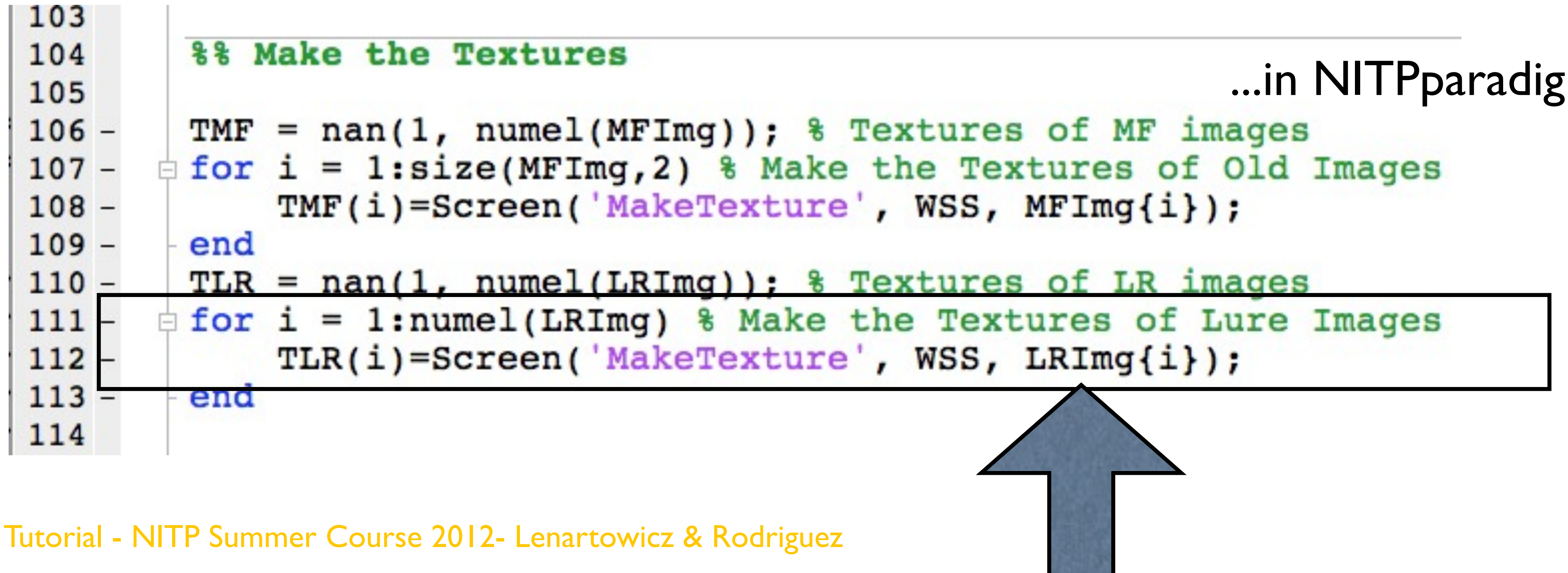
Remember this? It wouldn't work for us a while ago.

Second, image variable gets passed to the NITPparadigm.m function. And third, we use the MakeTexture call to 'prep' the images (this is all to speed up timing).



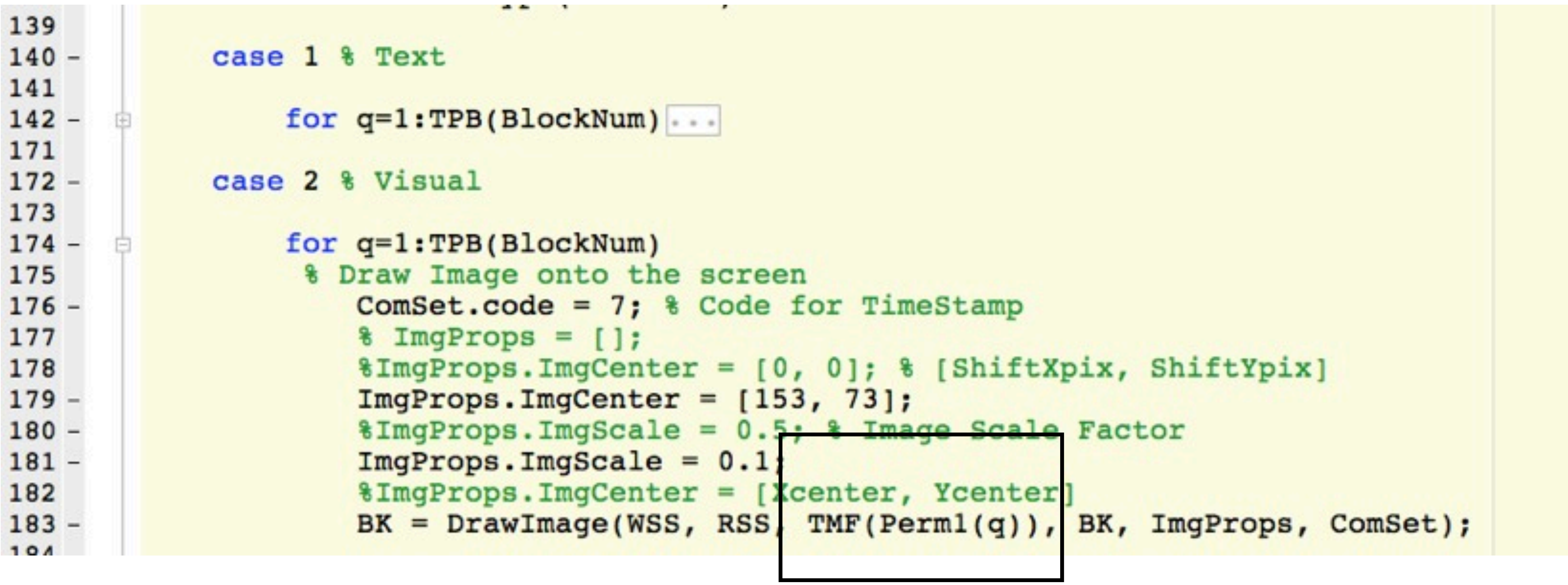
...in runEXPMT.m

```
%% Run Experiment

try % Start Try - Catch

    %% Run The Display loop

    for i = 1:Params.TotalBlocks

        %% Run The Paradigm

        TrialVariables = NITPParadigm(i, Params, ScreenHandels, ...
                         MFImg, LRImg, Audio, Fsample, DIO);

        %% Concatinate the cur     ial data with previous data

        % General Info
        VAT.TimeStamps = [VAT.Ti    tamps, TrialVariables.TimeStamps];
```



...in NITPparadigm.m

```
%% Make the Textures

TMF = nan(1, numel(MFImg)); % Textures of MF images
for i = 1:size(MFImg,2) % Make the Textures of Old Images
    TMF(i)=Screen('MakeTexture', WSS, MFImg{i});
end
TLR = nan(1, numel(LRImg)); % Textures of LR images
for i = 1:numel(LRImg) % Make the Textures of Lure Images
    TLR(i)=Screen('MakeTexture', WSS, LRImg{i});
end

```



```
    case 1 % Text

        for q=1:TPB(BlockNum)...

    case 2 % Visual

        for q=1:TPB(BlockNum)
         % Draw Image onto the screen
            ComSet.code = 7; % Code for TimeStamp
            % ImgProps = [];
            %ImgProps.ImgCenter = [0, 0]; % [ShiftXpix, ShiftYpix]
            ImgProps.ImgCenter = [153, 73];
            %ImgProps.ImgScale = 0.5; % Image Scale Factor
            ImgProps.ImgScale = 0.1;
            %ImgProps.ImgCenter = [Xcenter, Ycenter]
            BK = DrawImage(WSS, RSS, TMF(Perm1(q)), BK, ImgProps, ComSet);
```

THIS TMF VARIABLE IS YOUR IMAGE.
I'll let you figure out what Perm1(q) is... hint - Perm1 is created a few lines about in this code & q is just below the case 2 statement.

OK. So it's a bit complicated (blame Cameron) but it gives you wonderful control over timing and visual display. If you want to keep it simple just comment out most of this crazy stuff and keep it neat. We'll help.

HINT ---- Step2 tutorial shows you other fun and smart ways to present images. Also check out the Demos.

step 5 <<<< explore audio

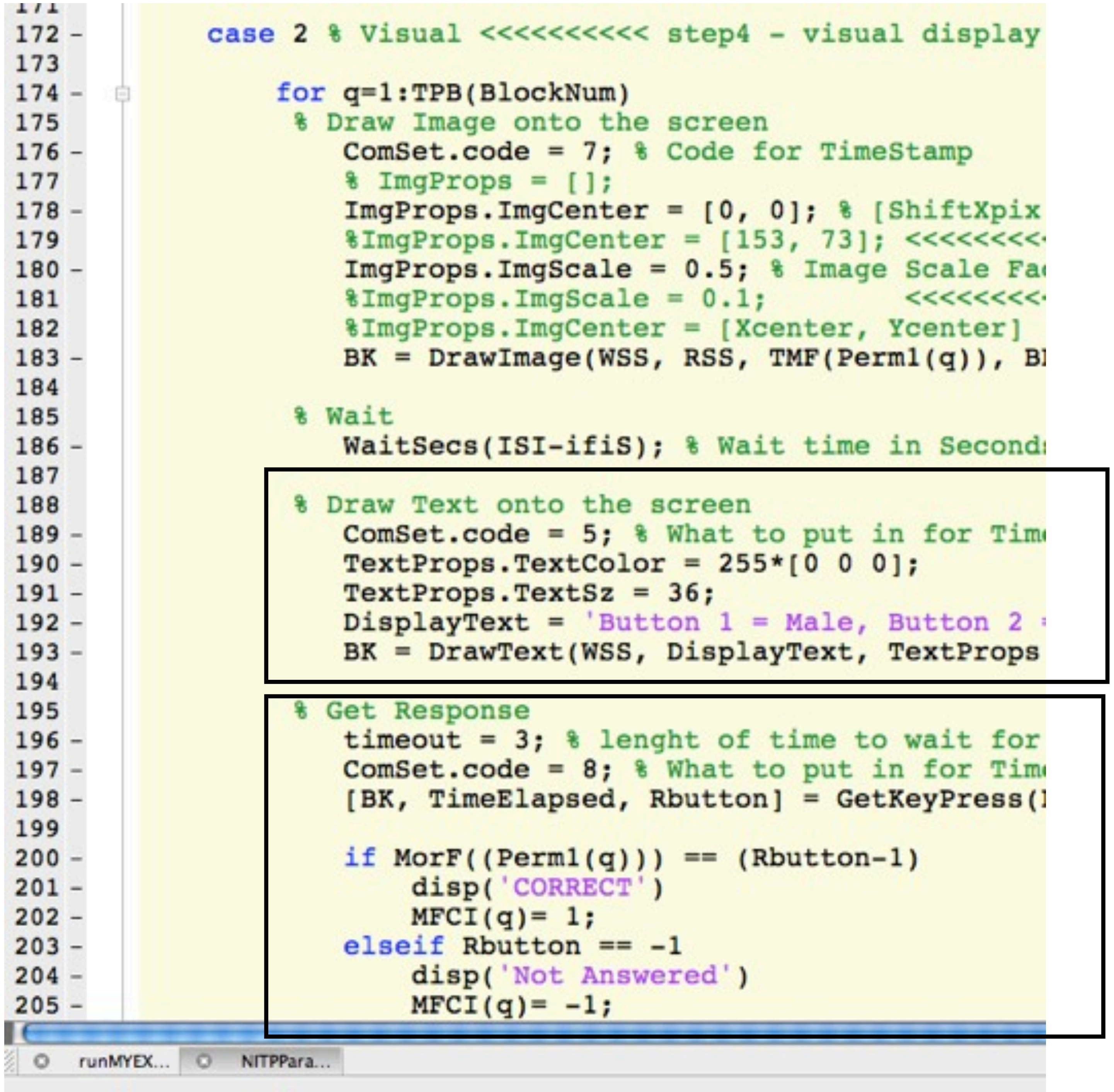
NOW MOVE AUDIO TO BE IN THE FIRST BLOCK AND CHECK OUT THE AUDITORY STIMULI
(hint - modify Params.BlockType to be [3 2 1] or [3 3 3] or any [3 x x] combo)

Your task... walk through the audio presentation display code to figure out:
- where in NITPparadigm.m audio gets shown (hint - case 3)
- where in the runEXPMT.m code the audio files get loaded (hint - after the visual stim get loaded)

HINT ---- Step2 tutorial shows you other fun and smart ways to present audio.

step 6 <<<< responses

Ultimately you'll want to collect some responses.

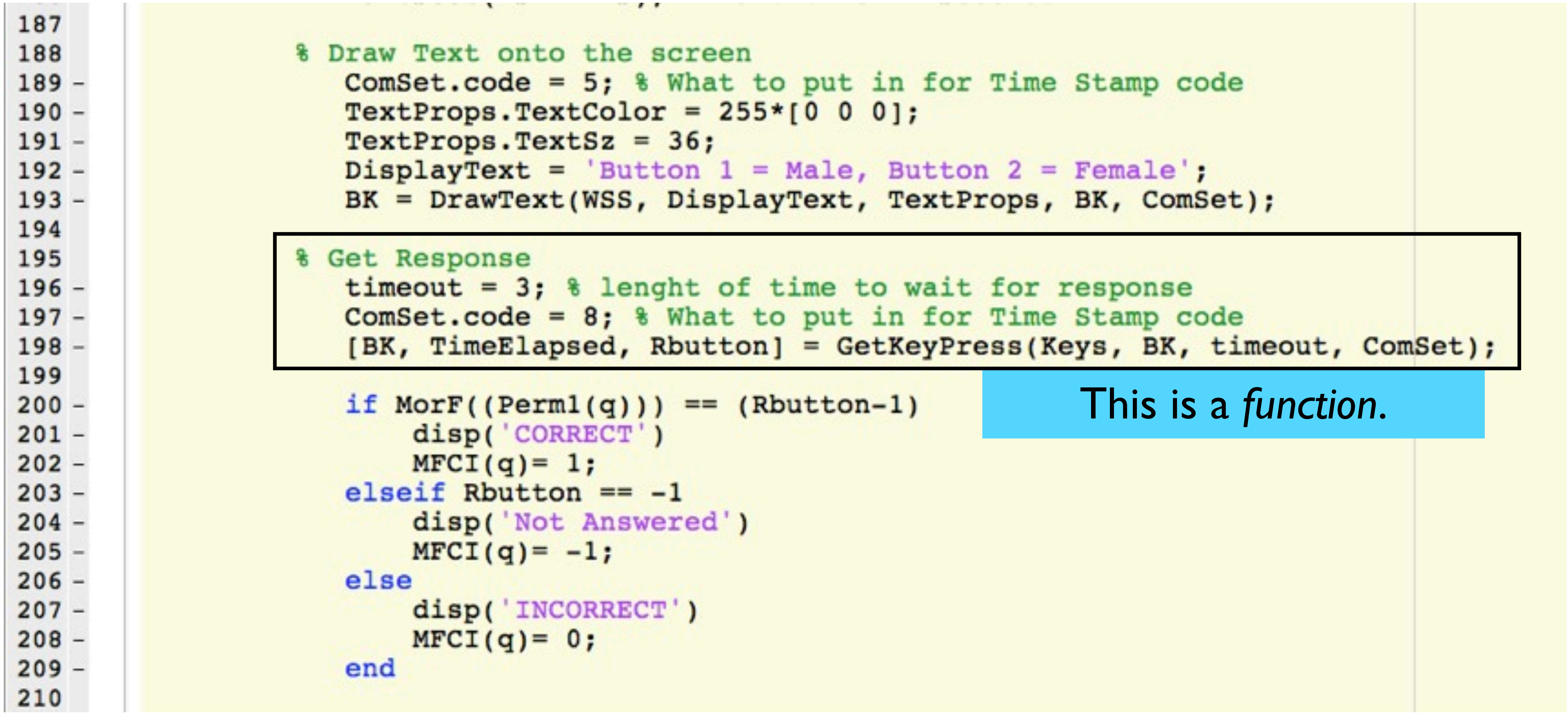


```
    case 2 % Visual <<<<<<<<<< step4 - visual display
        
        for q=1:TPB(BlockNum)
         % Draw Image onto the screen
            ComSet.code = 7; % Code for TimeStamp
            % ImgProps = [];
            ImgProps.ImgCenter = [0, 0]; % [ShiftXpix
            %ImgProps.ImgCenter = [153, 73]; <<<<<<<<
            ImgProps.ImgScale = 0.5; % Image Scale Fa
            %ImgProps.ImgScale = 0.1;         <<<<<<<<
            %ImgProps.ImgCenter = [Xcenter, Ycenter]
            BK = DrawImage(WSS, RSS, TMF(Perm1(q)), B
            
         % Wait
            WaitSecs(ISI-ifiS); % Wait time in Second
            
         % Draw Text onto the screen
            ComSet.code = 5; % What to put in for Tim
            TextProps.TextColor = 255*[0 0 0];
            TextProps.TextSz = 36;
            DisplayText = 'Button 1 = Male, Button 2
            BK = DrawText(WSS, DisplayText, TextProps
            
         % Get Response
            timeout = 3; % lenght of time to wait for
            ComSet.code = 8; % What to put in for Tim
            [BK, TimeElapsed, Rbutton] = GetKeyPress(
            
            if MorF((Perm1(q))) == (Rbutton-1)
                disp('CORRECT')
                MFCI(q)= 1;
            elseif Rbutton == -1
                disp('Not Answered')
                MFCI(q)= -1;
```

Present Inquiry Screen

Collect Response

Let’s Look at this closer.

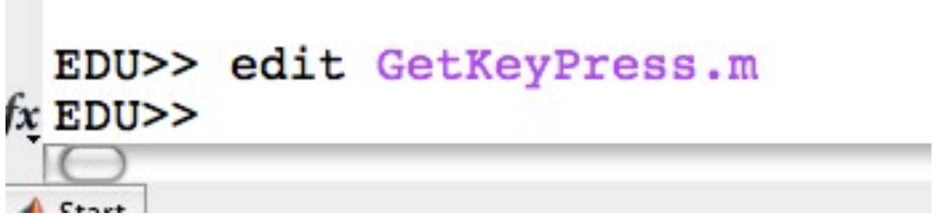


```
187
188          % Draw Text onto the screen
189 -           ComSet.code = 5; % What to put in for Time Stamp code
190 -           TextProps.TextColor = 255*[0 0 0];
191 -           TextProps.TextSz = 36;
192 -           DisplayText = 'Button 1 = Male, Button 2 = Female';
193 -           BK = DrawText(WSS, DisplayText, TextProps, BK, ComSet);
194
195          % Get Response
196 -           timeout = 3; % lenght of time to wait for response
197 -           ComSet.code = 8; % What to put in for Time Stamp code
198 -           [BK, TimeElapsed, Rbutton] = GetKeyPress(Keys, BK, timeout, ComSet);
199
200 -           if MorF((Perm1(q))) == (Rbutton-1)
201 -               disp('CORRECT')
202 -               MFCI(q)= 1;
203 -           elseif Rbutton == -1
204 -               disp('Not Answered')
205 -               MFCI(q)= -1;
206 -           else
207 -               disp('INCORRECT')
208 -               MFCI(q)= 0;
209 -           end
210
```

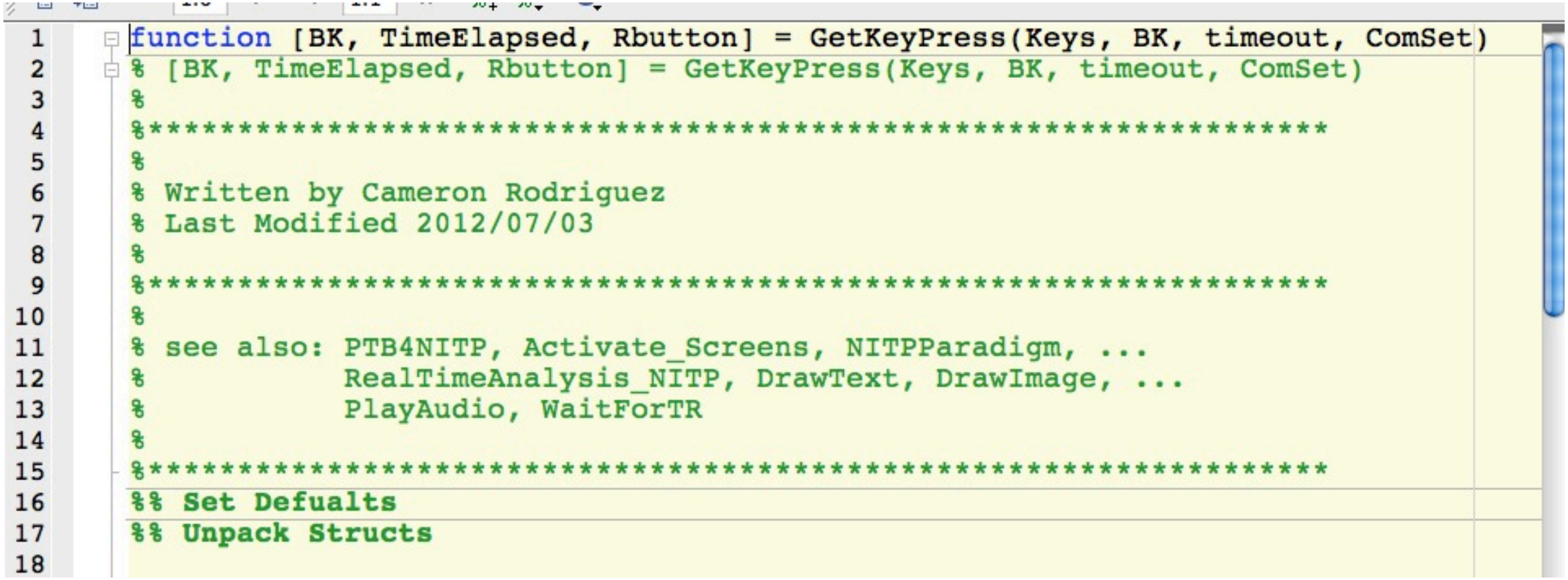
This is a *function.*

What is timeout? (line 196)

Let’s edit GetKeyPress.m

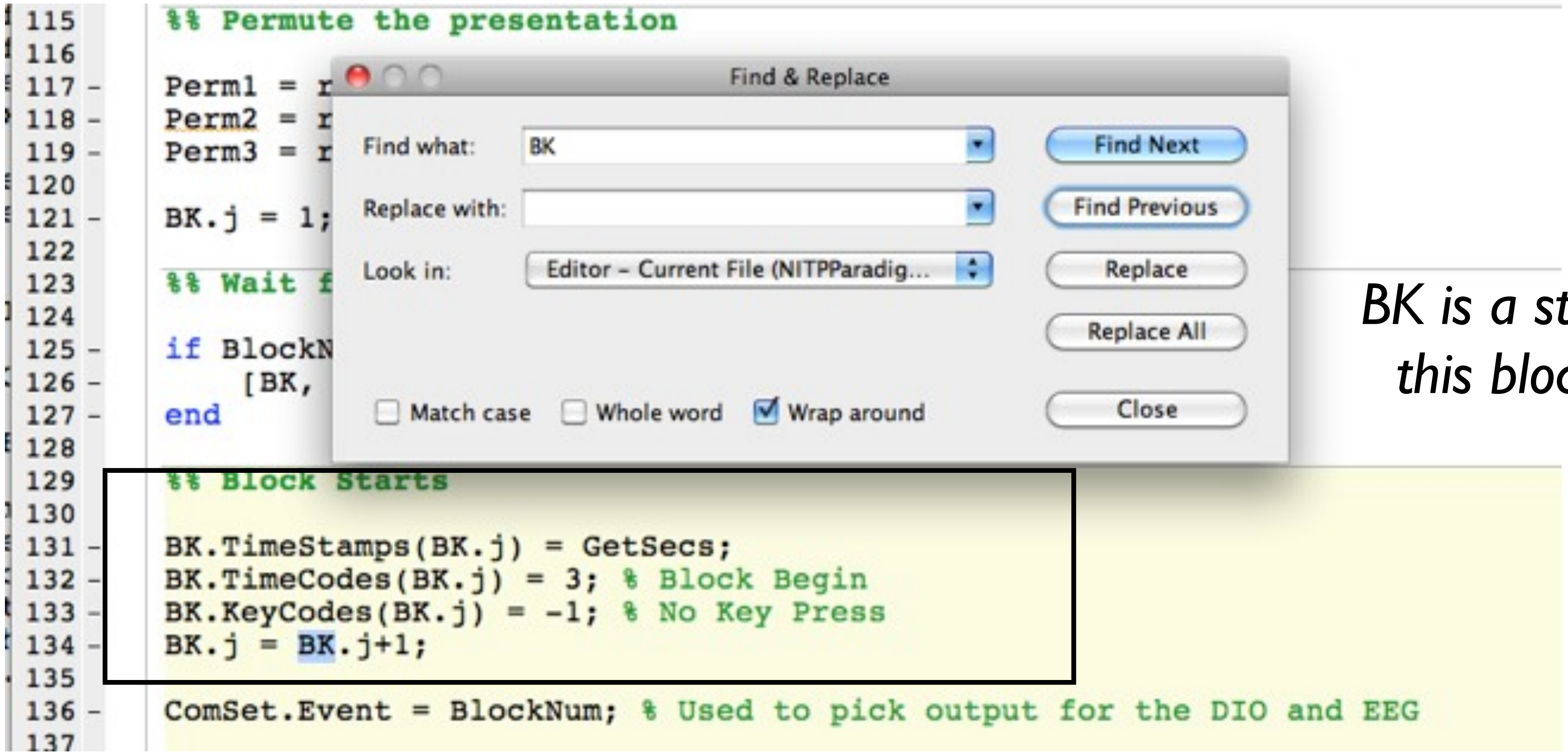


```
EDU>> edit GetKeyPress.m
EDU>>
```



```
function [BK, TimeElapsed, Rbutton] = GetKeyPress(Keys, BK, timeout, ComSet)
% [BK, TimeElapsed, Rbutton] = GetKeyPress(Keys, BK, timeout, ComSet)
%
%*******************************************************************
%
% Written by Cameron Rodriguez
% Last Modified 2012/07/03
%
%*******************************************************************
%
% see also: PTB4NITP, Activate_Screens, NITPParadigm, ...
%           RealTimeAnalysis_NITP, DrawText, DrawImage, ...
%           PlayAudio, WaitForTR
%
%*******************************************************************
%% Set Defualts
%% Unpack Structs

```

What are the inputs? Try to search for these in the calling function (NITPParadigm.m) to find out.



```
%% Permute the presentation

Perm1 = r
Perm2 = r
Perm3 = r

BK.j = 1;

%% Wait f

if BlockN
    [BK,
end

%% Block Starts

BK.TimeStamps(BK.j) = GetSecs;
BK.TimeCodes(BK.j) = 3; % Block Begin
BK.KeyCodes(BK.j) = -1; % No Key Press
BK.j = BK.j+1;

ComSet.Event = BlockNum; % Used to pick output for the DIO and EEG
```

Find & Replace

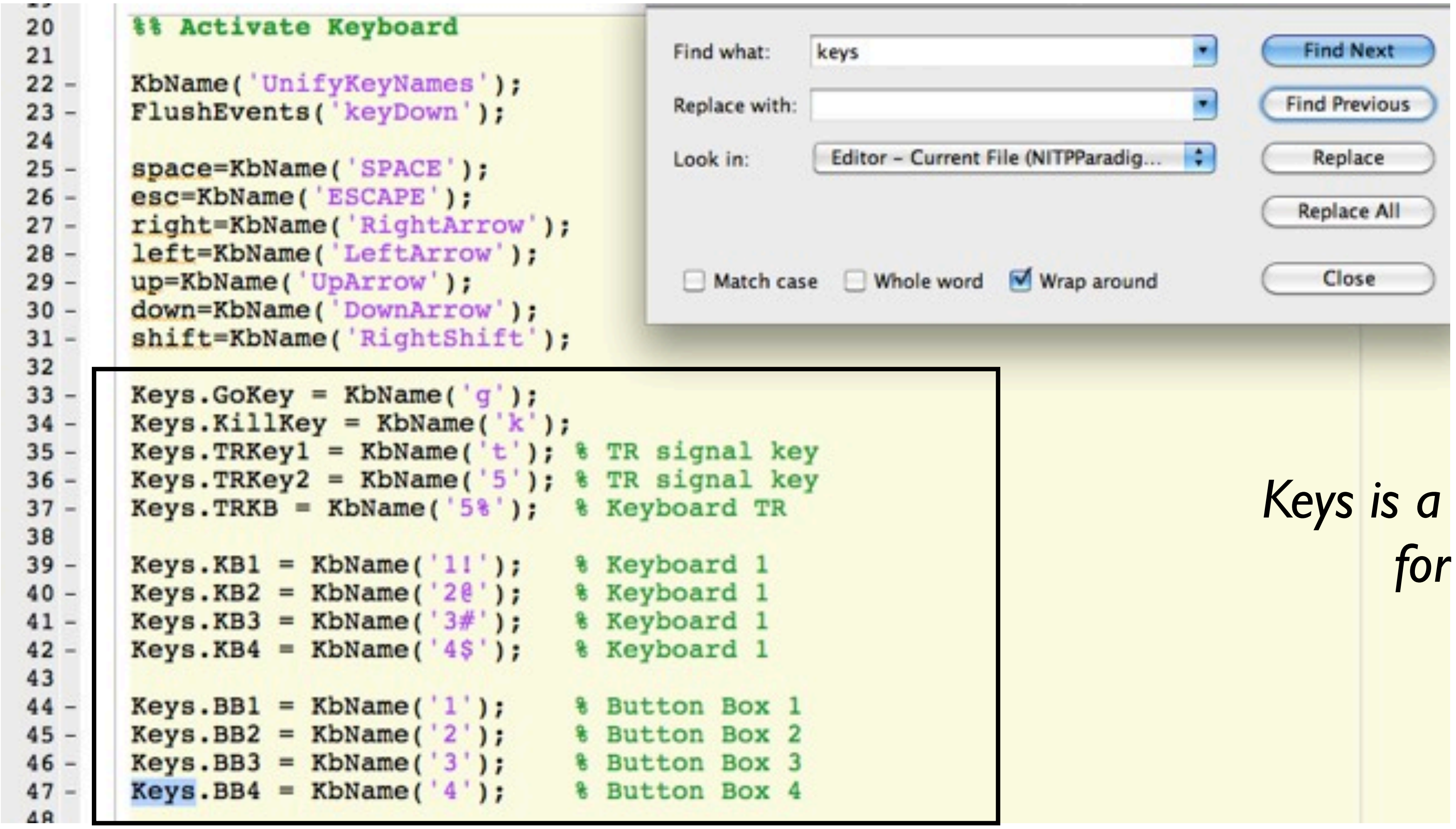
Find what: BK

Replace with:

Look in: Editor - Current File (NITPParadig...

☐ Match case ☐ Whole word ☑ Wrap around

BK is a struct holding some info for this block. We'll get back to this.



```
%% Activate Keyboard

KbName('UnifyKeyNames');
FlushEvents('keyDown');

space=KbName('SPACE');
esc=KbName('ESCAPE');
right=KbName('RightArrow');
left=KbName('LeftArrow');
up=KbName('UpArrow');
down=KbName('DownArrow');
shift=KbName('RightShift');

Keys.GoKey = KbName('g');
Keys.KillKey = KbName('k');
Keys.TRKey1 = KbName('t'); % TR signal key
Keys.TRKey2 = KbName('5'); % TR signal key
Keys.TRKB = KbName('5%');  % Keyboard TR

Keys.KB1 = KbName('1!');   % Keyboard 1
Keys.KB2 = KbName('2@');   % Keyboard 1
Keys.KB3 = KbName('3#');   % Keyboard 1
Keys.KB4 = KbName('4$');   % Keyboard 1

Keys.BB1 = KbName('1');    % Button Box 1
Keys.BB2 = KbName('2');    % Button Box 2
Keys.BB3 = KbName('3');    % Button Box 3
Keys.BB4 = KbName('4');    % Button Box 4
```

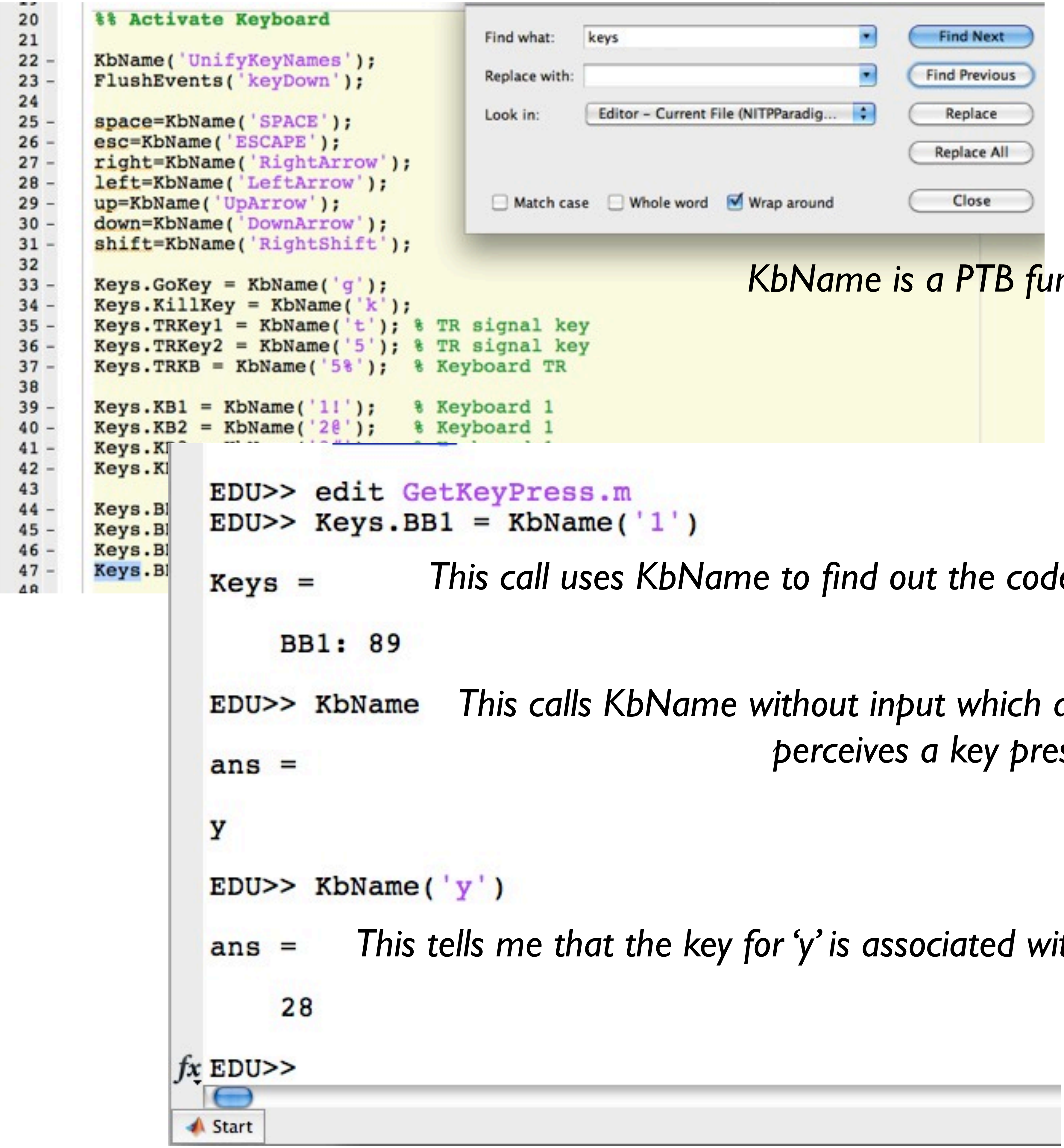
Find what: keys

Replace with:

Look in: Editor - Current File (NITPParadig...

☐ Match case ☐ Whole word ☑ Wrap around

Keys is a struct holding PTB's codes for various keypresses.



```
%% Activate Keyboard

KbName('UnifyKeyNames');
FlushEvents('keyDown');

space=KbName('SPACE');
esc=KbName('ESCAPE');
right=KbName('RightArrow');
left=KbName('LeftArrow');
up=KbName('UpArrow');
down=KbName('DownArrow');
shift=KbName('RightShift');

Keys.GoKey = KbName('g');
Keys.KillKey = KbName('k');
Keys.TRKey1 = KbName('t'); % TR signal key
Keys.TRKey2 = KbName('5'); % TR signal key
Keys.TRKB = KbName('5%');  % Keyboard TR

Keys.KB1 = KbName('1!');   % Keyboard 1
Keys.KB2 = KbName('2@');   % Keyboard 1
Keys.K
Keys.K

Keys.B
Keys.B
Keys.B
Keys.B
```

Find what: keys
Replace with:
Look in: Editor - Current File (NITPParadig...
☐ Match case ☐ Whole word ☑ Wrap around

KbName is a PTB function.

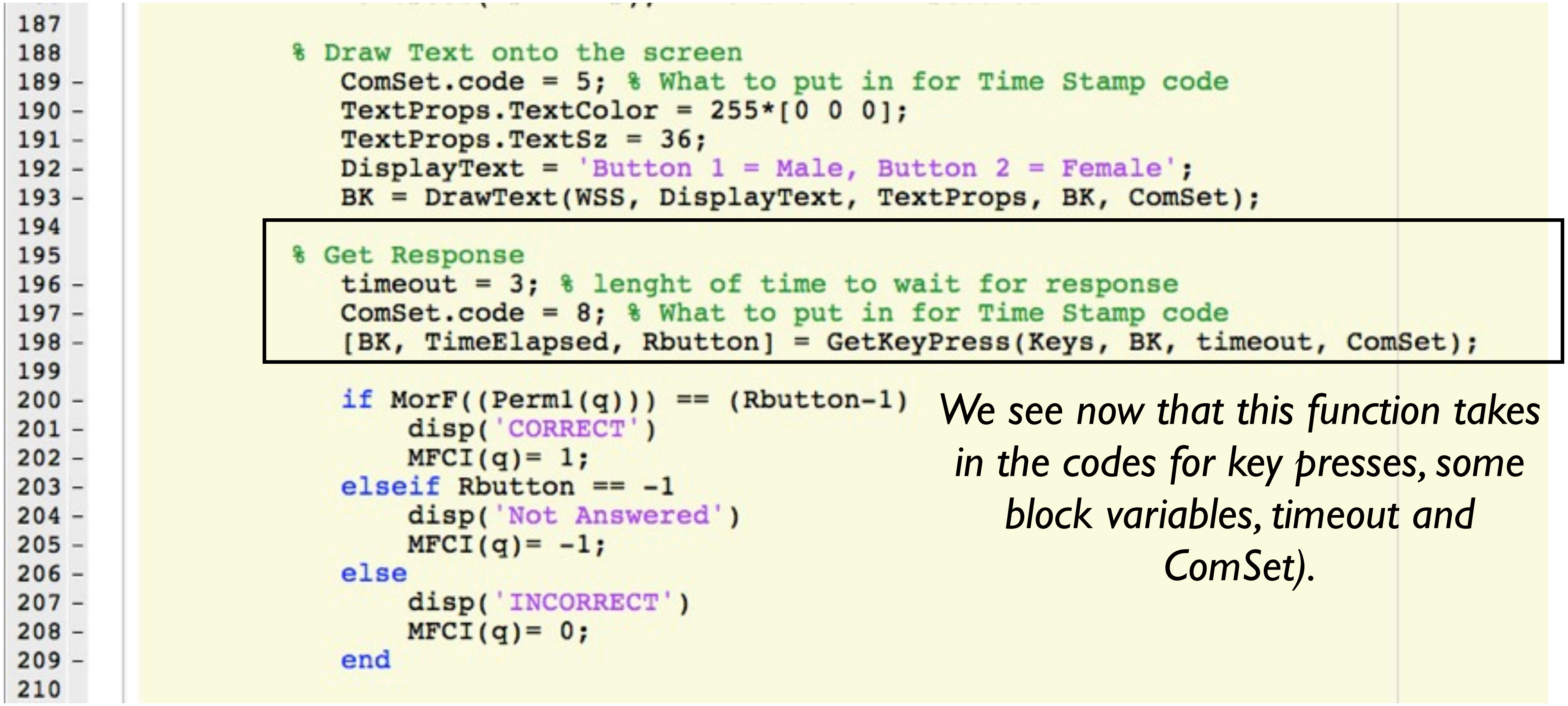
```
EDU>> edit GetKeyPress.m
EDU>> Keys.BB1 = KbName('1')

Keys =

    BB1: 89

EDU>> KbName

ans =

y

EDU>> KbName('y')

ans =

    28

fx EDU>>
```

This call uses KbName to find out the code for the '1' key.

This calls KbName without input which allows you to test how it perceives a key press.

This tells me that the key for 'y' is associated with key code #28.

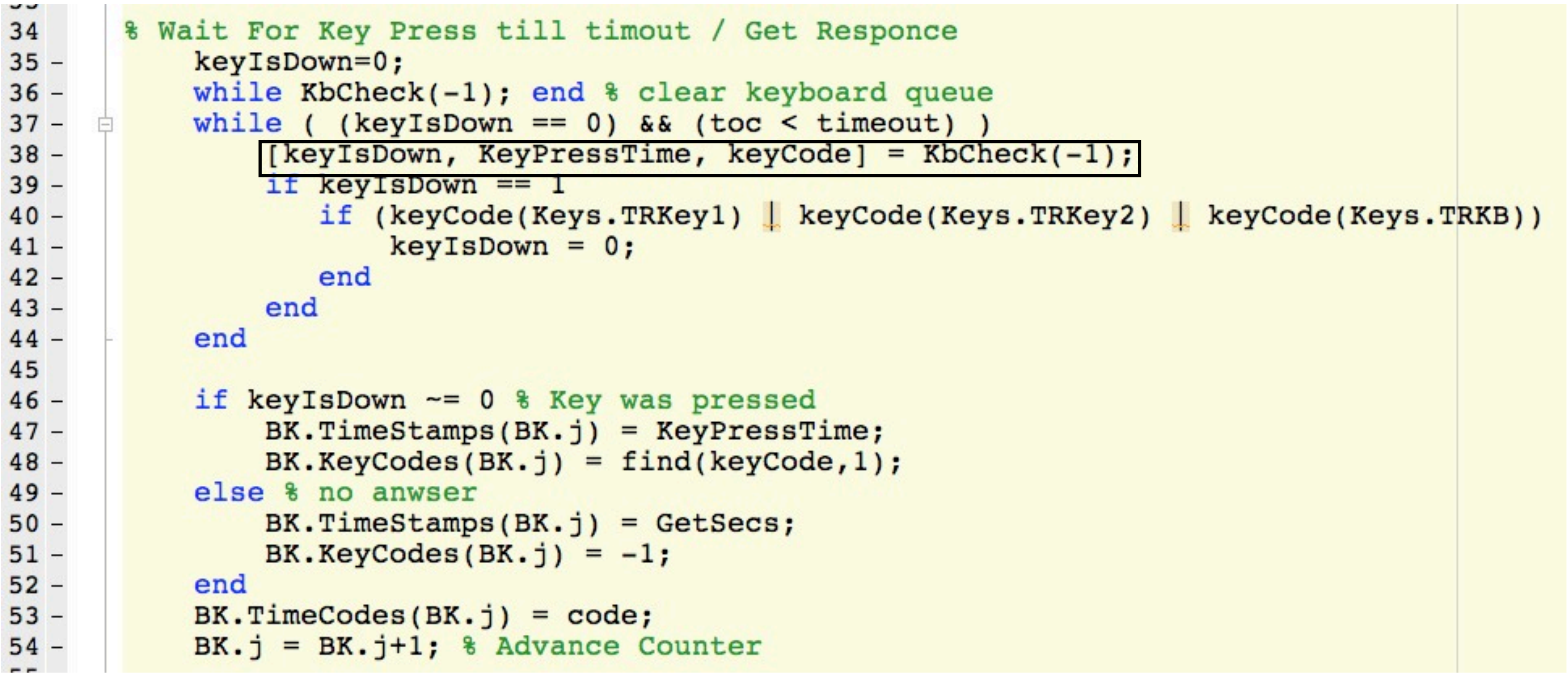
Back to our function.



```
        % Draw Text onto the screen
           ComSet.code = 5; % What to put in for Time Stamp code
           TextProps.TextColor = 255*[0 0 0];
           TextProps.TextSz = 36;
           DisplayText = 'Button 1 = Male, Button 2 = Female';
           BK = DrawText(WSS, DisplayText, TextProps, BK, ComSet);

        % Get Response
           timeout = 3; % lenght of time to wait for response
           ComSet.code = 8; % What to put in for Time Stamp code
           [BK, TimeElapsed, Rbutton] = GetKeyPress(Keys, BK, timeout, ComSet);

           if MorF((Perm1(q))) == (Rbutton-1)
               disp('CORRECT')
               MFCI(q)= 1;
           elseif Rbutton == -1
               disp('Not Answered')
               MFCI(q)= -1;
           else
               disp('INCORRECT')
               MFCI(q)= 0;
           end
```

We see now that this function takes in the codes for key presses, some block variables, timeout and ComSet).

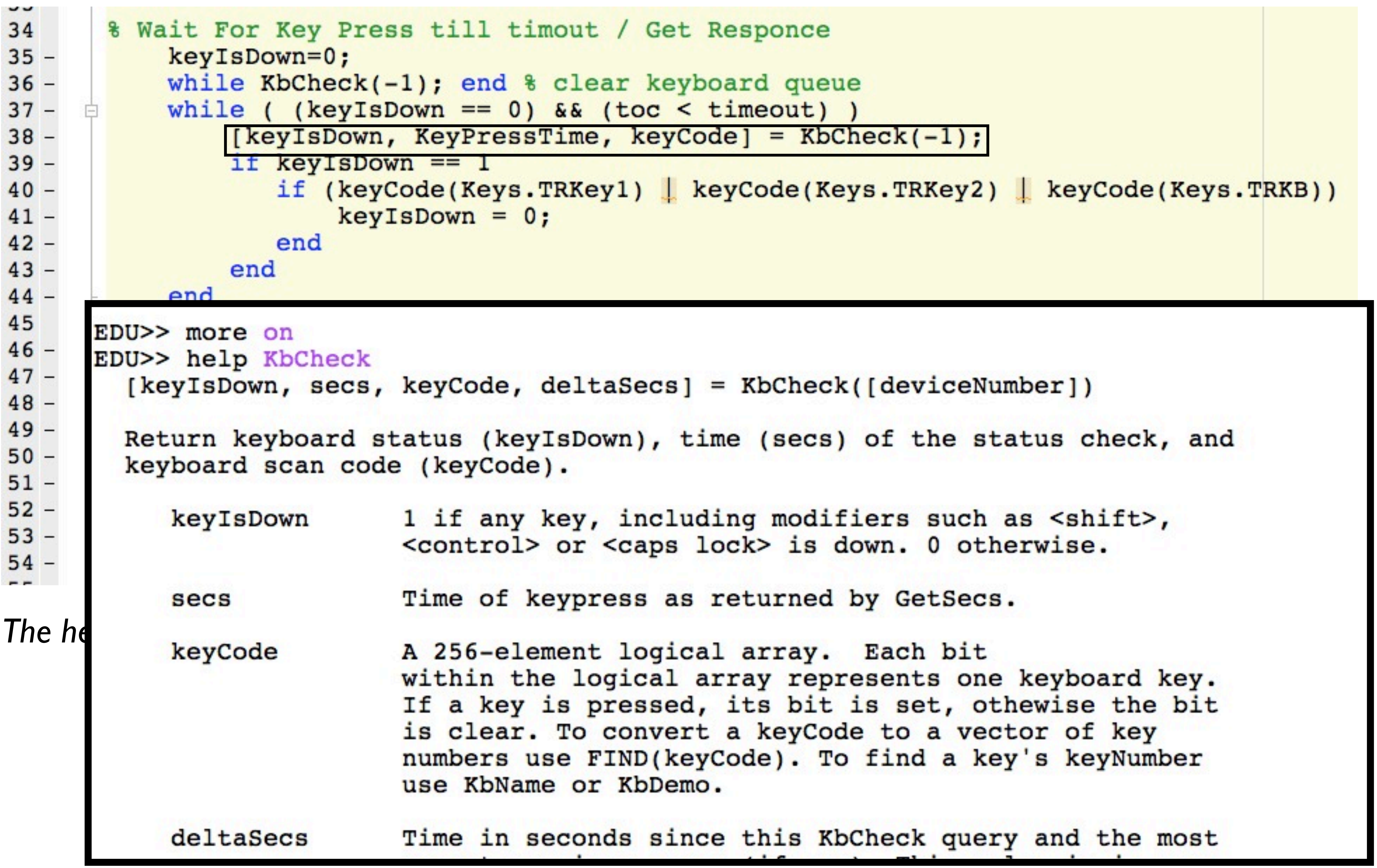
Inside GetKeyPress.m



```
% Wait For Key Press till timout / Get Responce
    keyIsDown=0;
    while KbCheck(-1); end % clear keyboard queue
    while ( (keyIsDown == 0) && (toc < timeout) )
        [keyIsDown, KeyPressTime, keyCode] = KbCheck(-1);
        if keyIsDown == 1
           if (keyCode(Keys.TRKey1) | keyCode(Keys.TRKey2) | keyCode(Keys.TRKB))
               keyIsDown = 0;
           end
        end
    end

    if keyIsDown ~= 0 % Key was pressed
        BK.TimeStamps(BK.j) = KeyPressTime;
        BK.KeyCodes(BK.j) = find(keyCode,1);
    else % no anwser
        BK.TimeStamps(BK.j) = GetSecs;
        BK.KeyCodes(BK.j) = -1;
    end
    BK.TimeCodes(BK.j) = code;
    BK.j = BK.j+1; % Advance Counter
```

The heart of this command sequence (which you're welcome to trust blindly) is the KbCheck command.

Inside GetKeyPress.m

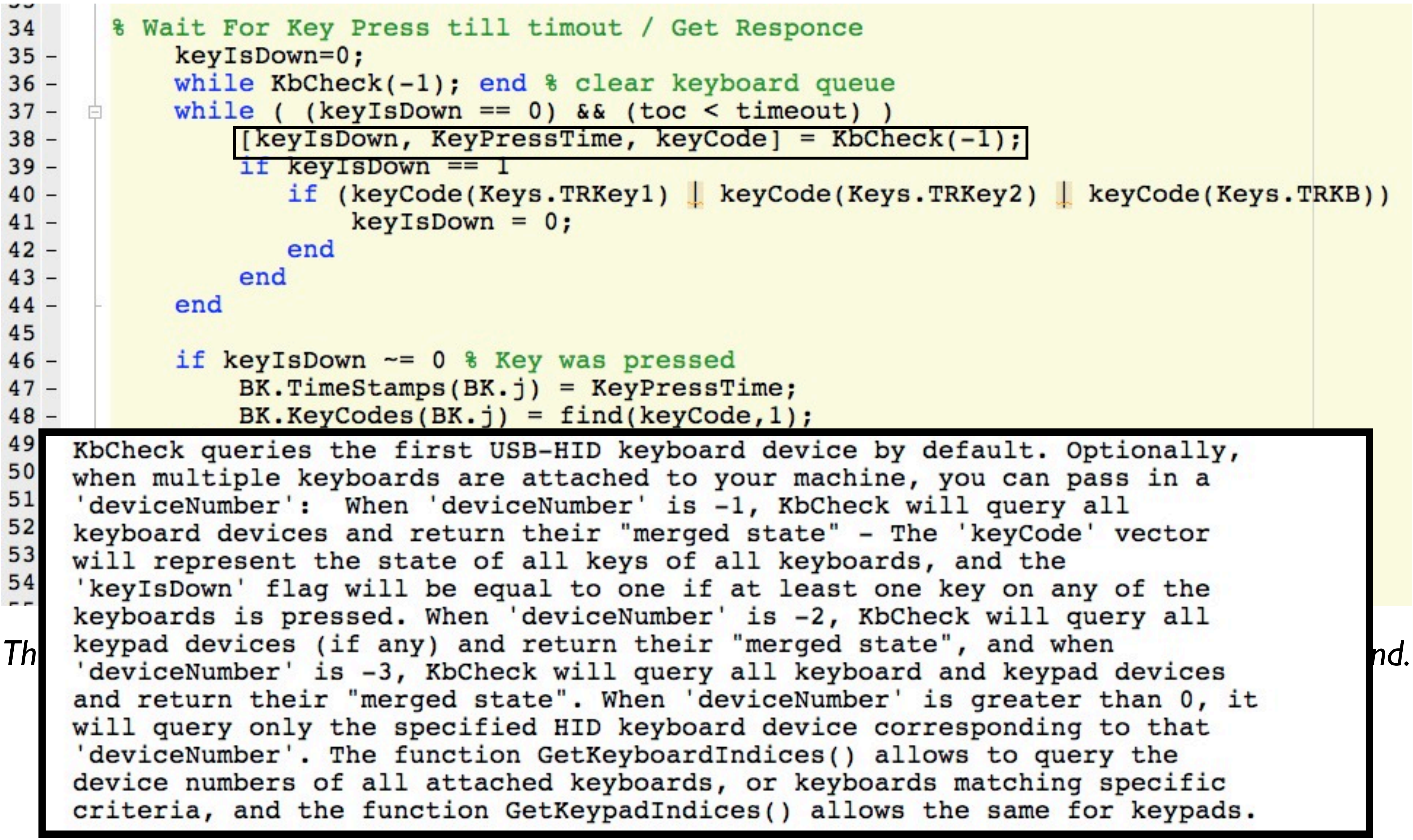


```
% Wait For Key Press till timout / Get Responce
    keyIsDown=0;
    while KbCheck(-1); end % clear keyboard queue
    while ( (keyIsDown == 0) && (toc < timeout) )
        [keyIsDown, KeyPressTime, keyCode] = KbCheck(-1);
        if keyIsDown == 1
            if (keyCode(Keys.TRKey1) | keyCode(Keys.TRKey2) | keyCode(Keys.TRKB))
                keyIsDown = 0;
            end
        end
    end
```

```
EDU>> more on
EDU>> help KbCheck
  [keyIsDown, secs, keyCode, deltaSecs] = KbCheck([deviceNumber])

  Return keyboard status (keyIsDown), time (secs) of the status check, and
  keyboard scan code (keyCode).

     keyIsDown      1 if any key, including modifiers such as <shift>,
                    <control> or <caps lock> is down. 0 otherwise.

     secs           Time of keypress as returned by GetSecs.

     keyCode        A 256-element logical array.  Each bit
                    within the logical array represents one keyboard key.
                    If a key is pressed, its bit is set, othewise the bit
                    is clear. To convert a keyCode to a vector of key
                    numbers use FIND(keyCode). To find a key's keyNumber
                    use KbName or KbDemo.

     deltaSecs      Time in seconds since this KbCheck query and the most
```

The he

Inside GetKeyPress.m



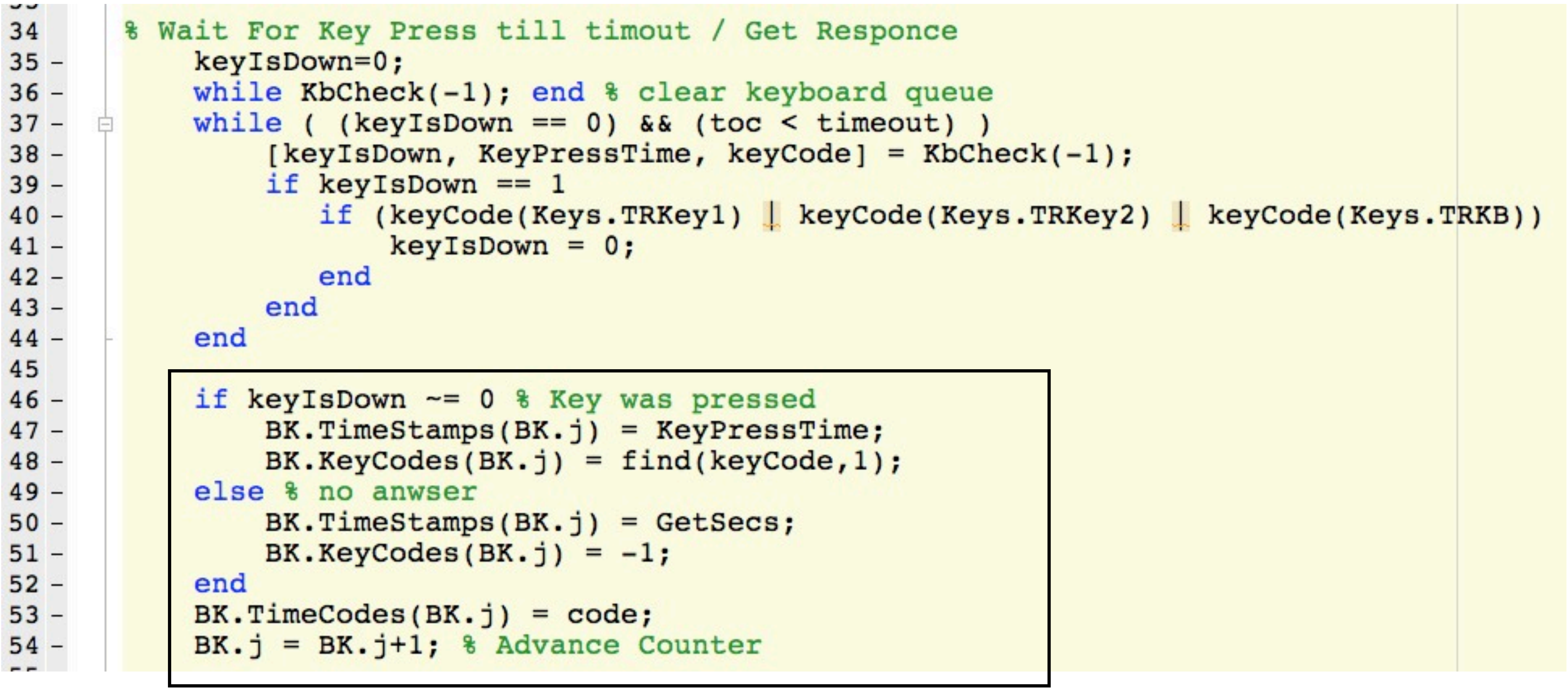
```
% Wait For Key Press till timout / Get Responce
    keyIsDown=0;
    while KbCheck(-1); end % clear keyboard queue
    while ( (keyIsDown == 0) && (toc < timeout) )
        [keyIsDown, KeyPressTime, keyCode] = KbCheck(-1);
        if keyIsDown == 1
           if (keyCode(Keys.TRKey1) | keyCode(Keys.TRKey2) | keyCode(Keys.TRKB))
               keyIsDown = 0;
           end
        end
    end

    if keyIsDown ~= 0 % Key was pressed
        BK.TimeStamps(BK.j) = KeyPressTime;
        BK.KeyCodes(BK.j) = find(keyCode,1);
```

```
KbCheck queries the first USB-HID keyboard device by default. Optionally,
when multiple keyboards are attached to your machine, you can pass in a
'deviceNumber': When 'deviceNumber' is -1, KbCheck will query all
keyboard devices and return their "merged state" - The 'keyCode' vector
will represent the state of all keys of all keyboards, and the
'keyIsDown' flag will be equal to one if at least one key on any of the
keyboards is pressed. When 'deviceNumber' is -2, KbCheck will query all
keypad devices (if any) and return their "merged state", and when
'deviceNumber' is -3, KbCheck will query all keyboard and keypad devices
and return their "merged state". When 'deviceNumber' is greater than 0, it
will query only the specified HID keyboard device corresponding to that
'deviceNumber'. The function GetKeyboardIndices() allows to query the
device numbers of all attached keyboards, or keyboards matching specific
criteria, and the function GetKeypadIndices() allows the same for keypads.
```

Th nd.

HINT ---- Step2 tutorial shows you other fun and smart ways to present collect responses from select devices (rather than querying all) - check out the hid_probe.m function for how this can be done.

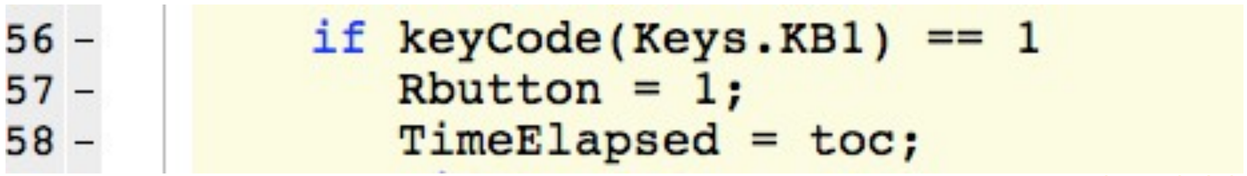
Inside GetKeyPress.m



```
% Wait For Key Press till timout / Get Responce
    keyIsDown=0;
    while KbCheck(-1); end % clear keyboard queue
    while ( (keyIsDown == 0) && (toc < timeout) )
        [keyIsDown, KeyPressTime, keyCode] = KbCheck(-1);
        if keyIsDown == 1
           if (keyCode(Keys.TRKey1) | keyCode(Keys.TRKey2) | keyCode(Keys.TRKB))
               keyIsDown = 0;
           end
        end
    end

    if keyIsDown ~= 0 % Key was pressed
        BK.TimeStamps(BK.j) = KeyPressTime;
        BK.KeyCodes(BK.j) = find(keyCode,1);
    else % no anwser
        BK.TimeStamps(BK.j) = GetSecs;
        BK.KeyCodes(BK.j) = -1;
    end
    BK.TimeCodes(BK.j) = code;
    BK.j = BK.j+1; % Advance Counter
```

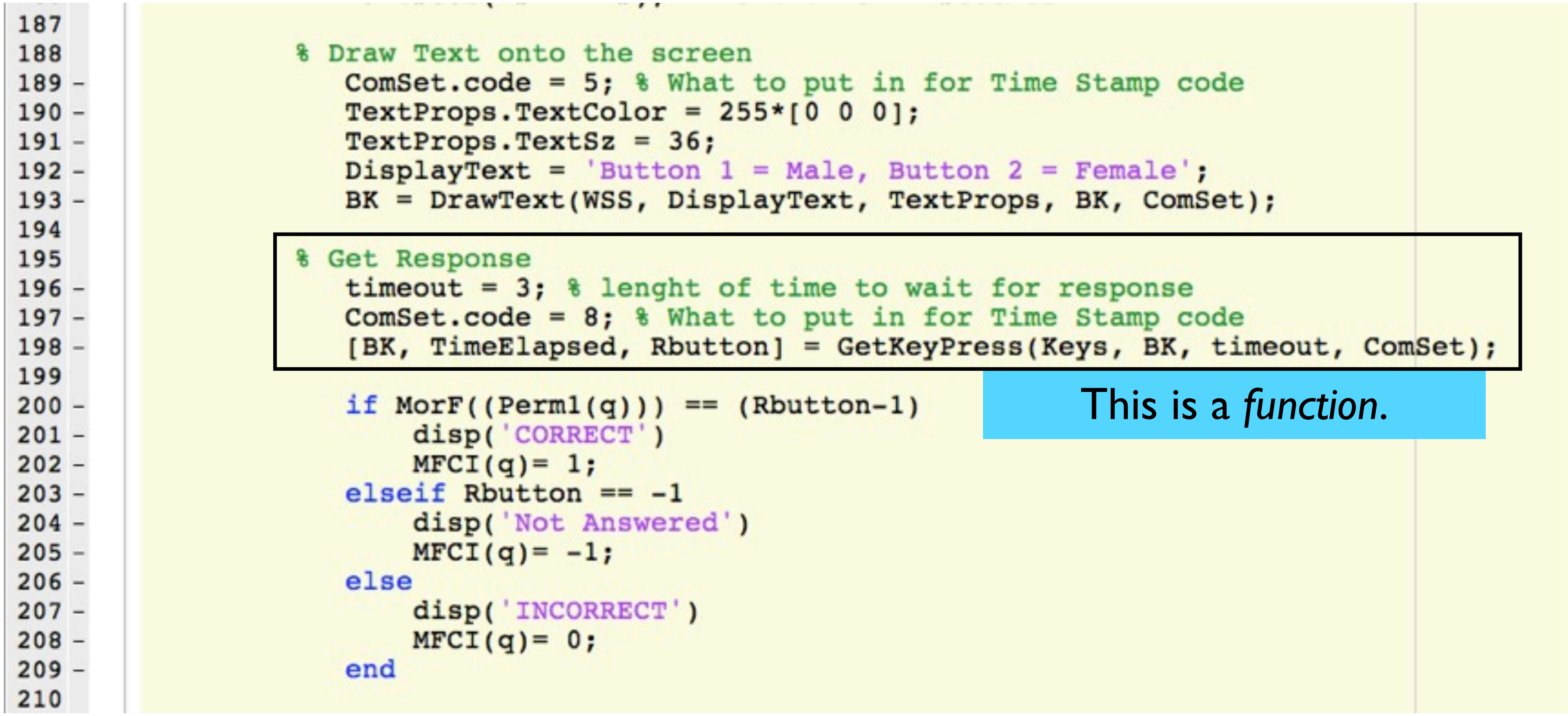
After detecting a key press, this function sets some variables to BK (block struct) - including the response time (KeyPressTime) and which key was pressed (numerical code produced by find(keyCode,1).

Feel free figure the 'code' variable, what GetSecs.m does and what BK.j refers to (hint - it's just a counter of events). And if we look further down in the code we'll see that the rbutton output variables represents which button was actually pressed.



```
    if keyCode(Keys.KB1) == 1
       Rbutton = 1;
       TimeElapsed = toc;
```

So what happens here?

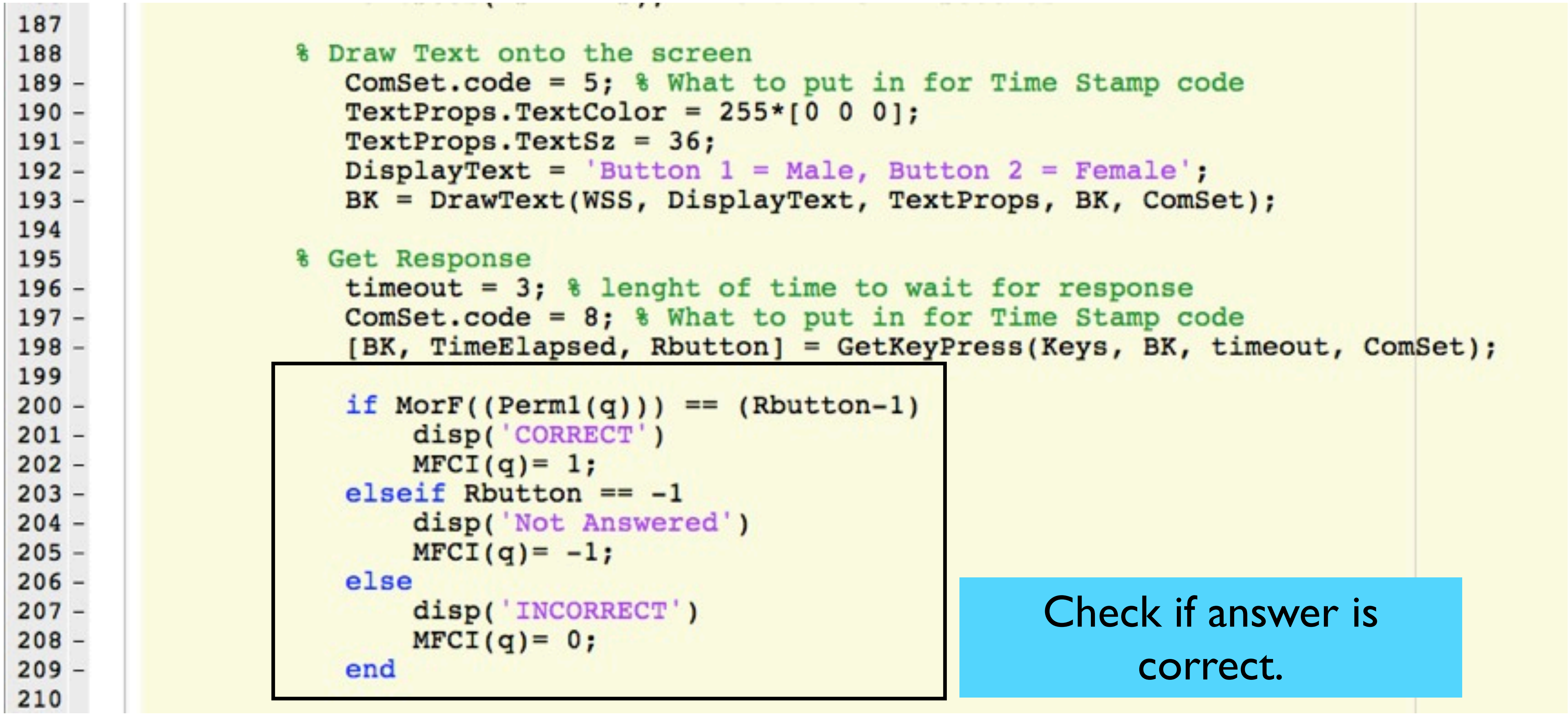


```
    % Draw Text onto the screen
       ComSet.code = 5; % What to put in for Time Stamp code
       TextProps.TextColor = 255*[0 0 0];
       TextProps.TextSz = 36;
       DisplayText = 'Button 1 = Male, Button 2 = Female';
       BK = DrawText(WSS, DisplayText, TextProps, BK, ComSet);

    % Get Response
       timeout = 3; % lenght of time to wait for response
       ComSet.code = 8; % What to put in for Time Stamp code
       [BK, TimeElapsed, Rbutton] = GetKeyPress(Keys, BK, timeout, ComSet);

       if MorF((Perm1(q))) == (Rbutton-1)
           disp('CORRECT')
           MFCI(q)= 1;
       elseif Rbutton == -1
           disp('Not Answered')
           MFCI(q)= -1;
       else
           disp('INCORRECT')
           MFCI(q)= 0;
       end
```

This is a *function.*

GetKeyPress wait for 'timeout' seconds for a key press and then spits out BK (info), RT (TimeElapsed) and key press (Rbutton).

So what happens here?



```
% Draw Text onto the screen
   ComSet.code = 5; % What to put in for Time Stamp code
   TextProps.TextColor = 255*[0 0 0];
   TextProps.TextSz = 36;
   DisplayText = 'Button 1 = Male, Button 2 = Female';
   BK = DrawText(WSS, DisplayText, TextProps, BK, ComSet);

% Get Response
   timeout = 3; % lenght of time to wait for response
   ComSet.code = 8; % What to put in for Time Stamp code
   [BK, TimeElapsed, Rbutton] = GetKeyPress(Keys, BK, timeout, ComSet);

   if MorF((Perm1(q))) == (Rbutton-1)
       disp('CORRECT')
       MFCI(q)= 1;
   elseif Rbutton == -1
       disp('Not Answered')
       MFCI(q)= -1;
   else
       disp('INCORRECT')
       MFCI(q)= 0;
   end
```

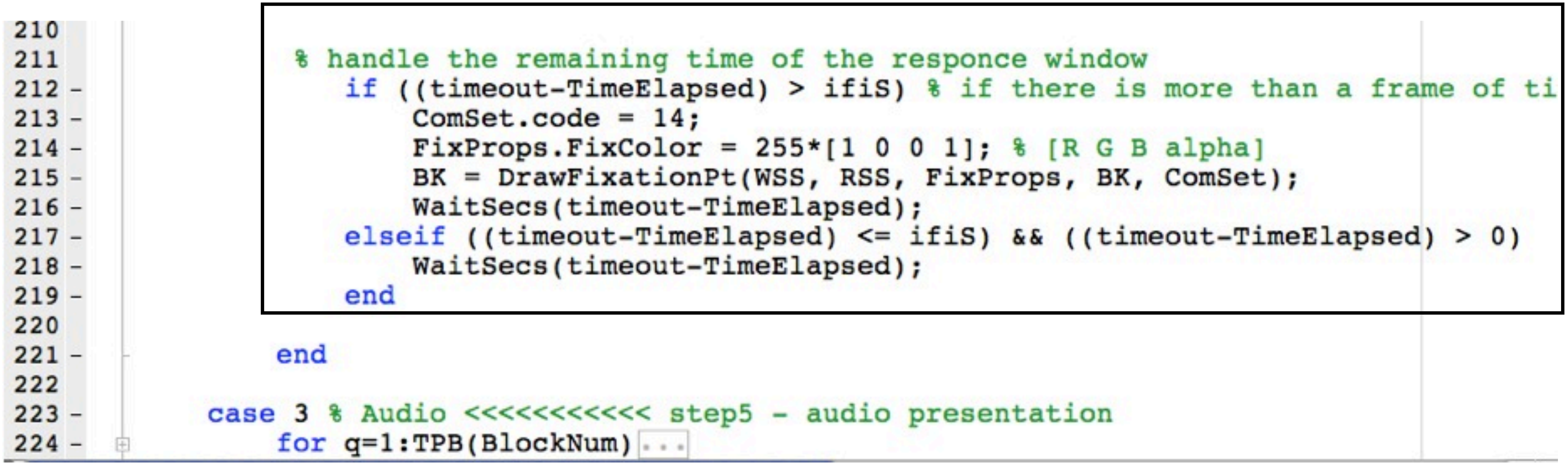
Check if answer is correct.

Your task - figure out what MorF is (hint - it holds info about correct buttons for the MorF image. Look up if (help if) to understand if statements.

HINT ---- Step2 tutorial shows you other fun and smart ways to collect responses and test your response speed.

step 7 <<<< save data

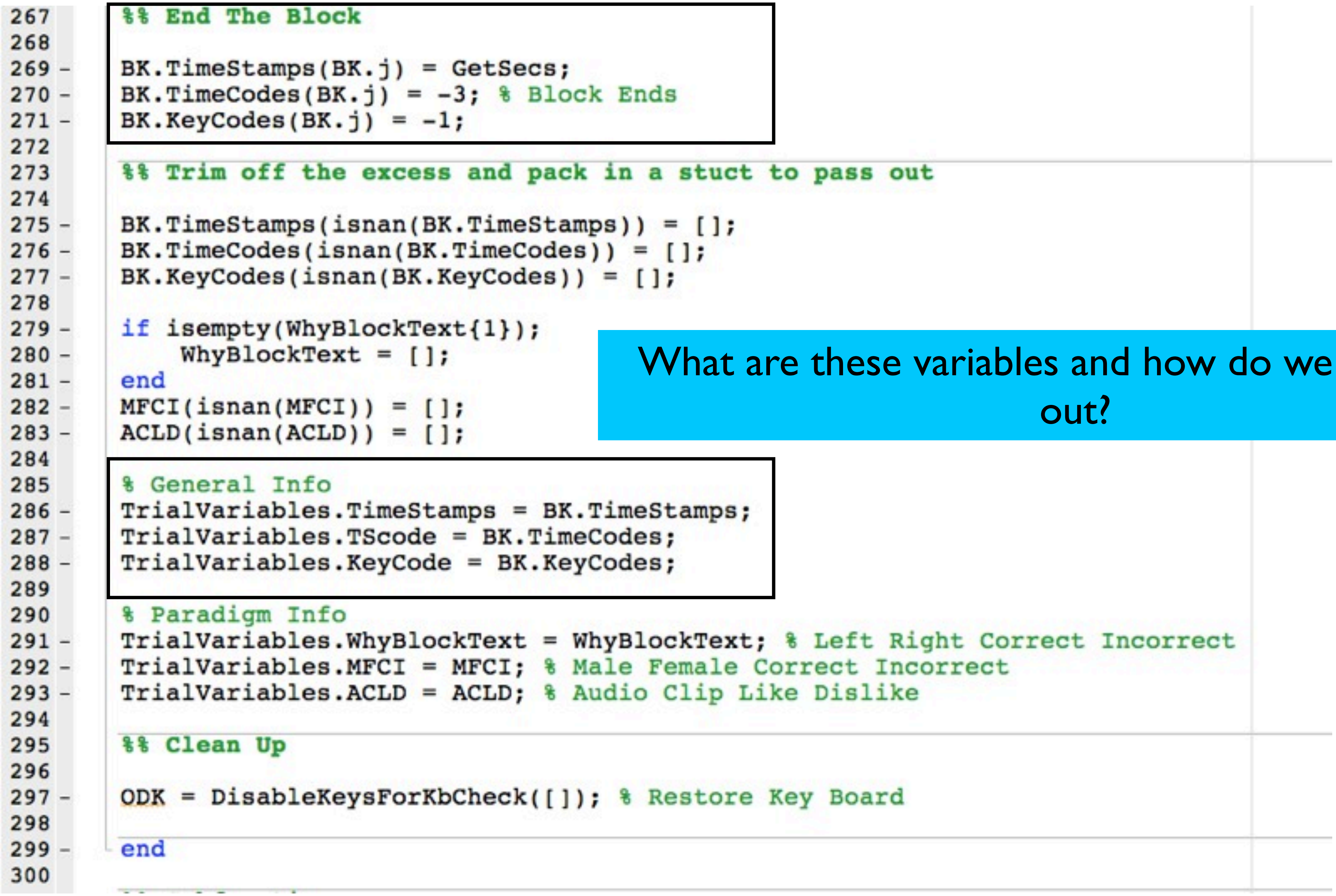
The remainder of NITPparadigm. waits out some dead time...



```
      % handle the remaining time of the responce window
        if ((timeout-TimeElapsed) > ifiS) % if there is more than a frame of ti
          ComSet.code = 14;
          FixProps.FixColor = 255*[1 0 0 1]; % [R G B alpha]
          BK = DrawFixationPt(WSS, RSS, FixProps, BK, ComSet);
          WaitSecs(timeout-TimeElapsed);
        elseif ((timeout-TimeElapsed) <= ifiS) && ((timeout-TimeElapsed) > 0)
          WaitSecs(timeout-TimeElapsed);
        end

      end

    case 3 % Audio <<<<<<<<<<< step5 - audio presentation
      for q=1:TPB(BlockNum)...
```

and then...

Assigns a few variables to the TrialVariables and Block structures...

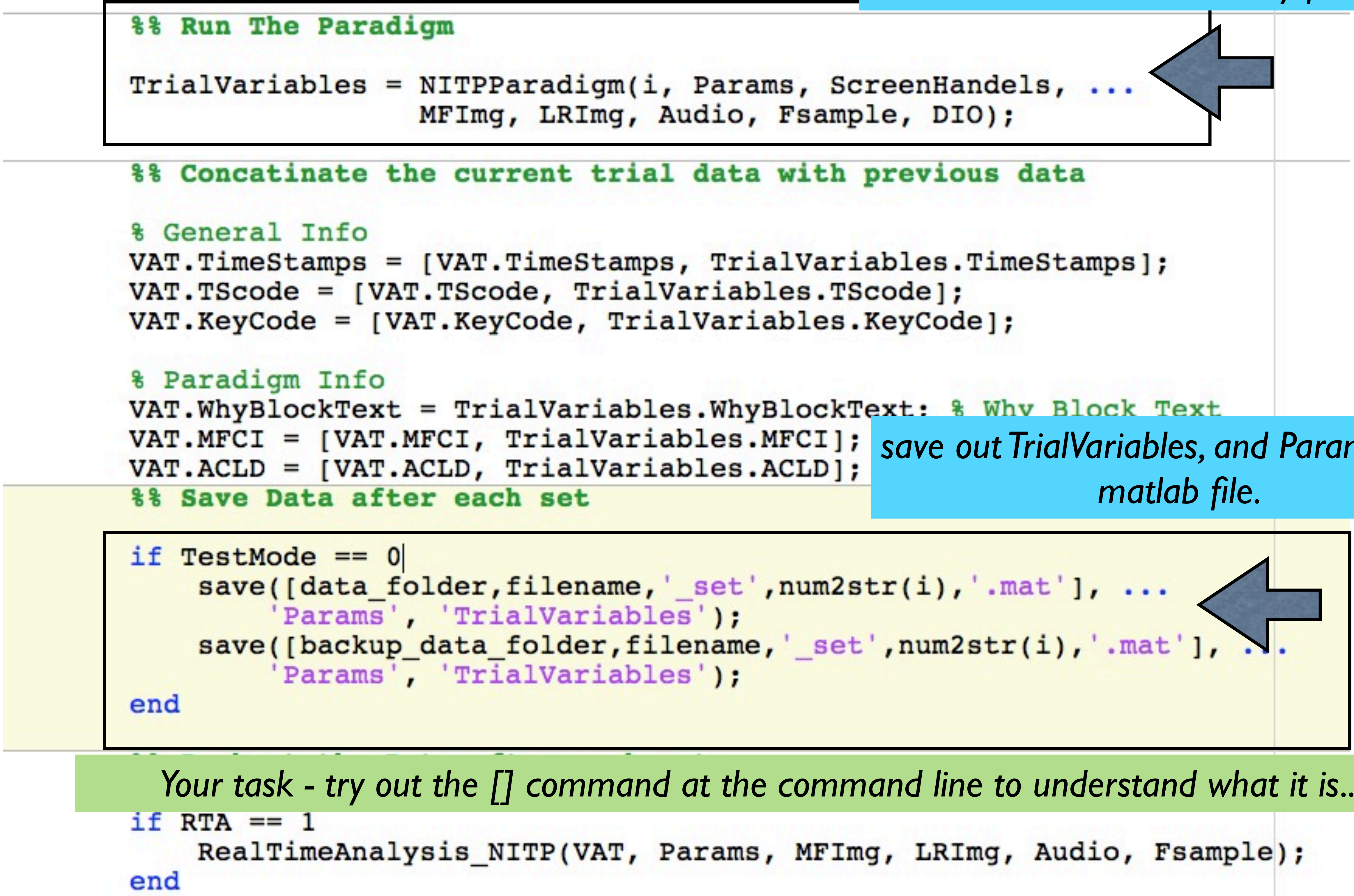


```
%% End The Block

BK.TimeStamps(BK.j) = GetSecs;
BK.TimeCodes(BK.j) = -3; % Block Ends
BK.KeyCodes(BK.j) = -1;

%% Trim off the excess and pack in a stuct to pass out

BK.TimeStamps(isnan(BK.TimeStamps)) = [];
BK.TimeCodes(isnan(BK.TimeCodes)) = [];
BK.KeyCodes(isnan(BK.KeyCodes)) = [];

if isempty(WhyBlockText{1});
   WhyBlockText = [];
end
MFCI(isnan(MFCI)) = [];
ACLD(isnan(ACLD)) = [];

% General Info
TrialVariables.TimeStamps = BK.TimeStamps;
TrialVariables.TScode = BK.TimeCodes;
TrialVariables.KeyCode = BK.KeyCodes;

% Paradigm Info
TrialVariables.WhyBlockText = WhyBlockText; % Left Right Correct Incorrect
TrialVariables.MFCI = MFCI; % Male Female Correct Incorrect
TrialVariables.ACLD = ACLD; % Audio Clip Like Dislike

%% Clean Up

ODK = DisableKeysForKbCheck([]); % Restore Key Board

end
```

What are these variables and how do we save them out?

Let’s go back runMYEXPMT.m and
see how to do this…

Spit out TrialVariables which hold info about the event and key press.



```
%% Run The Paradigm

TrialVariables = NITPParadigm(i, Params, ScreenHandels, ...
                 MFImg, LRImg, Audio, Fsample, DIO);

%% Concatinate the current trial data with previous data

% General Info
VAT.TimeStamps = [VAT.TimeStamps, TrialVariables.TimeStamps];
VAT.TScode = [VAT.TScode, TrialVariables.TScode];
VAT.KeyCode = [VAT.KeyCode, TrialVariables.KeyCode];

% Paradigm Info
VAT.WhyBlockText = TrialVariables.WhyBlockText; % Why Block Text
VAT.MFCI = [VAT.MFCI, TrialVariables.MFCI];
VAT.ACLD = [VAT.ACLD, TrialVariables.ACLD];
%% Save Data after each set

if TestMode == 0
    save([data_folder,filename,'_set',num2str(i),'.mat'], ...
        'Params', 'TrialVariables');
    save([backup_data_folder,filename,'_set',num2str(i),'.mat'], ...
        'Params', 'TrialVariables');
end

if RTA == 1
    RealTimeAnalysis_NITP(VAT, Params, MFImg, LRImg, Audio, Fsample);
end
```

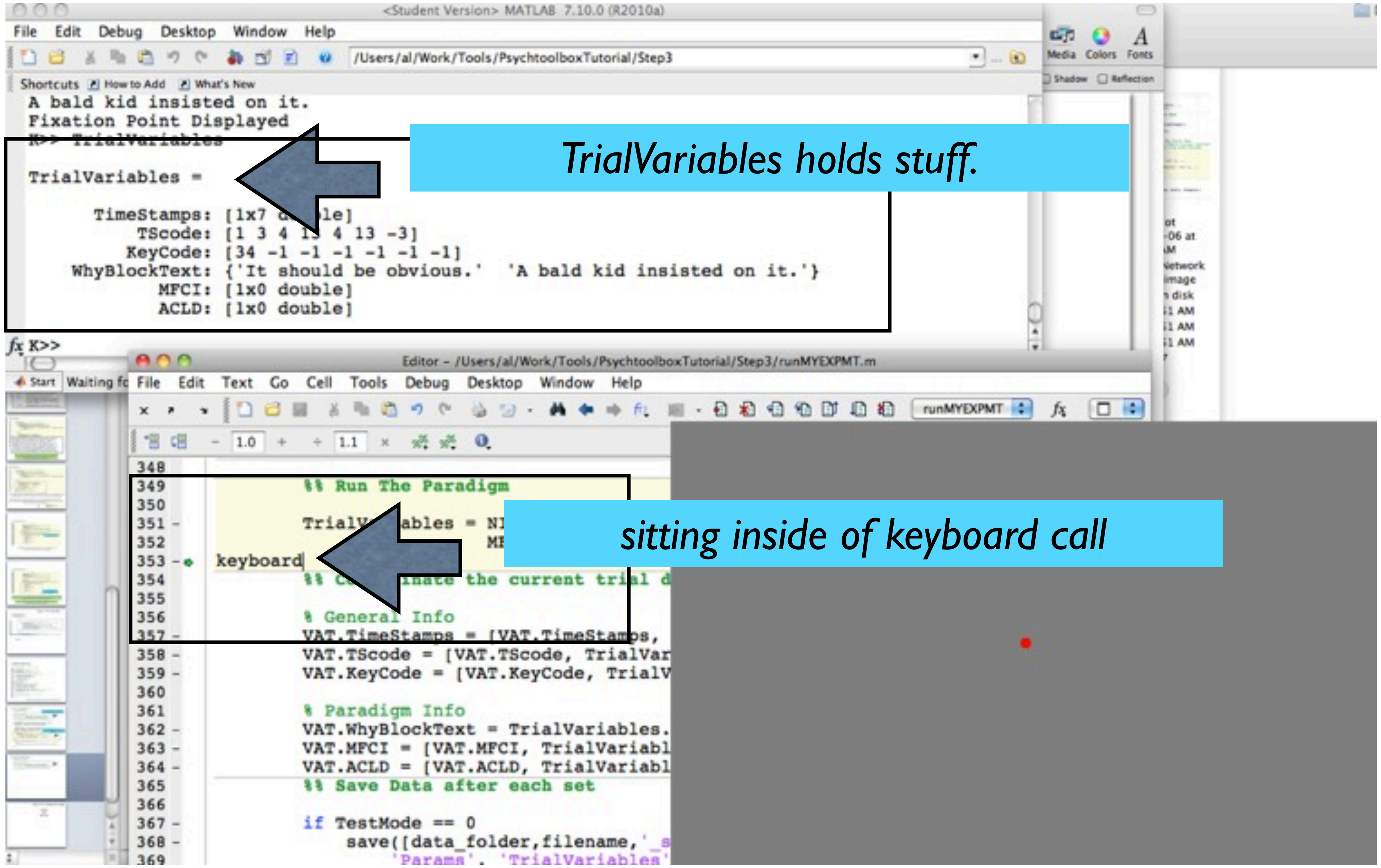
save out TrialVariables, and Params into a matlab file.

Your task - try out the [] command at the command line to understand what it is…

Let's peak into TrialVariables (put a keyboard after this call)...

```
%% Run The Paradigm

TrialVariables = NITPParadigm(i, Params, ScreenHandels, ...
                 MFImg, LRImg, Audio, Fsample, DIO);
```

```
%% Concatinate the current trial data with previous data
```


<Student Version> MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/Users/al/Work/Tools/PsychtoolboxTutorial/Step3
Shortcuts How to Add What's New
Editor - /Users/al/Work/Tools/PsychtoolboxTutorial/Step3/runMYEXPMT.m
File Edit Text Go Cell Tools Debug Desktop Window Help
runMYEXPMT

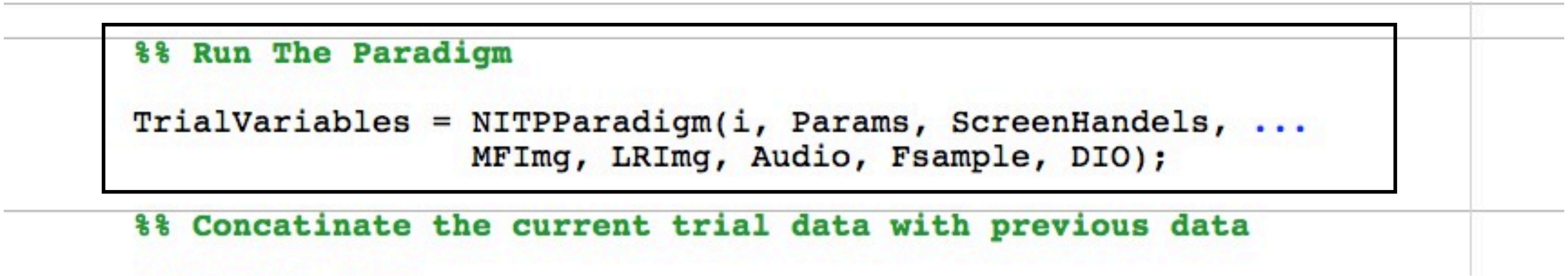
```
A bald kid insisted on it.
Fixation Point Displayed
K>> TrialVariables

TrialVariables =

       TimeStamps: [1x7 double]
           TScode: [1 3 4 13 4 13 -3]
          KeyCode: [34 -1 -1 -1 -1 -1 -1]
    WhyBlockText: {'It should be obvious.'  'A bald kid insisted on it.'}
             MFCI: [1x0 double]
             ACLD: [1x0 double]

K>>
```

TrialVariables holds stuff.

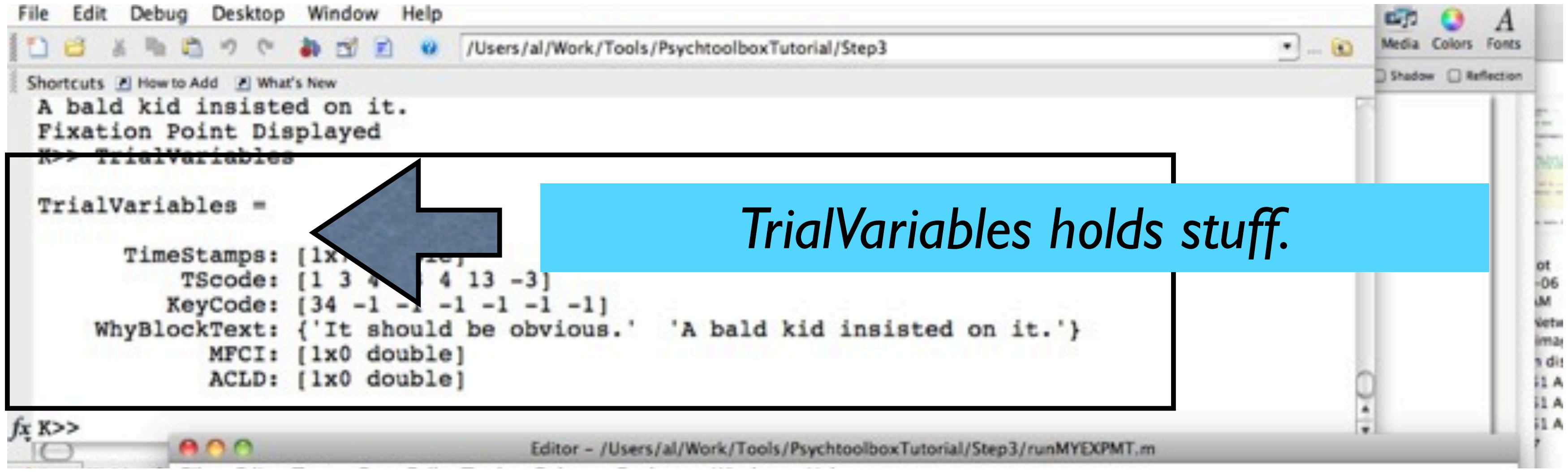
```
            %% Run The Paradigm

            TrialVariables = NI
                             MF
keyboard
            %% Concatinate the current trial d

            % General Info
            VAT.TimeStamps = [VAT.TimeStamps,
            VAT.TScode = [VAT.TScode, TrialVar
            VAT.KeyCode = [VAT.KeyCode, TrialV

            % Paradigm Info
            VAT.WhyBlockText = TrialVariables.
            VAT.MFCI = [VAT.MFCI, TrialVariabl
            VAT.ACLD = [VAT.ACLD, TrialVariabl
            %% Save Data after each set

            if TestMode == 0
                save([data_folder,filename,'_s
                    'Params', 'TrialVariables'
```

sitting inside of keyboard call

Let's peak into TrialVariables (put a keyboard after this call)...

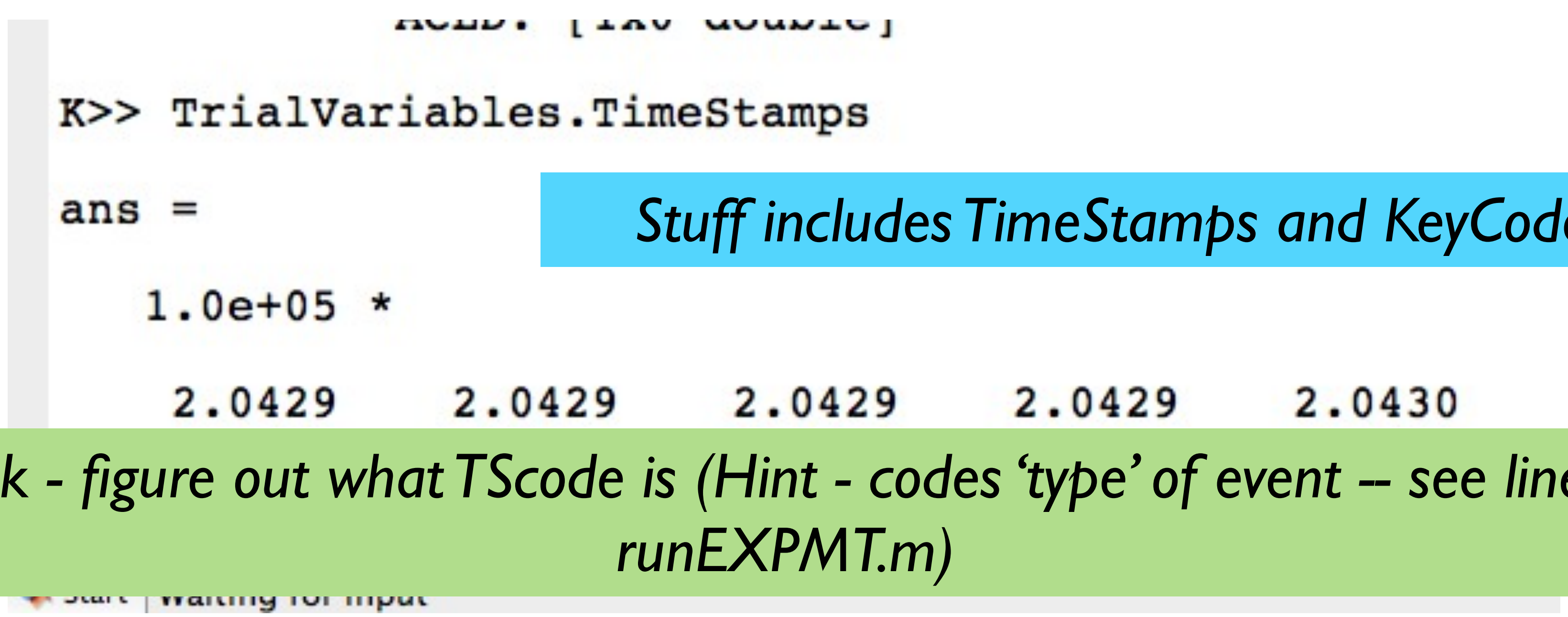


```
%% Run The Paradigm

TrialVariables = NITPParadigm(i, Params, ScreenHandels, ...
                 MFImg, LRImg, Audio, Fsample, DIO);

%% Concatinate the current trial data with previous data
```


File Edit Debug Desktop Window Help
/Users/al/Work/Tools/PsychtoolboxTutorial/Step3
Shortcuts How to Add What's New
Media Colors Fonts
Shadow Reflection
Editor – /Users/al/Work/Tools/PsychtoolboxTutorial/Step3/runMYEXPMT.m

```
A bald kid insisted on it.
Fixation Point Displayed
K>> TrialVariables

TrialVariables = 

     TimeStamps: [1x
         TScode: [1 3 4     4 13 -3]
        KeyCode: [34 -1 -1 -1 -1 -1 -1]
  WhyBlockText: {'It should be obvious.'  'A bald kid insisted on it.'}
           MFCI: [1x0 double]
           ACLD: [1x0 double]

fx K>>
```

TrialVariables holds stuff.

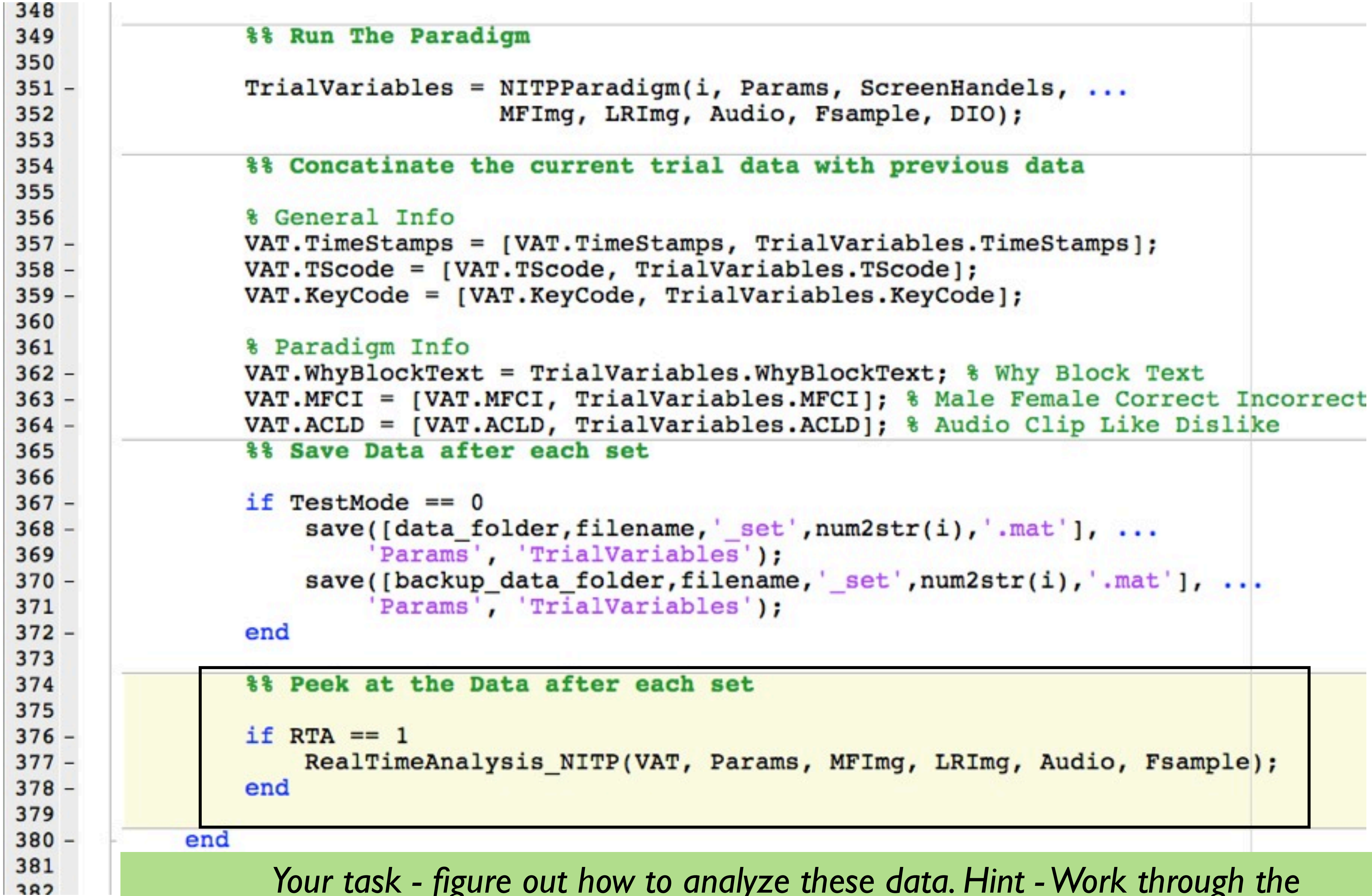


```
K>> TrialVariables.TimeStamps

ans =

   1.0e+05 *

    2.0429    2.0429    2.0429    2.0429    2.0430
```

Stuff includes TimeStamps and KeyCodes

Your task - figure out what TScode is (Hint - codes 'type' of event -- see line 135 in runEXPMT.m)

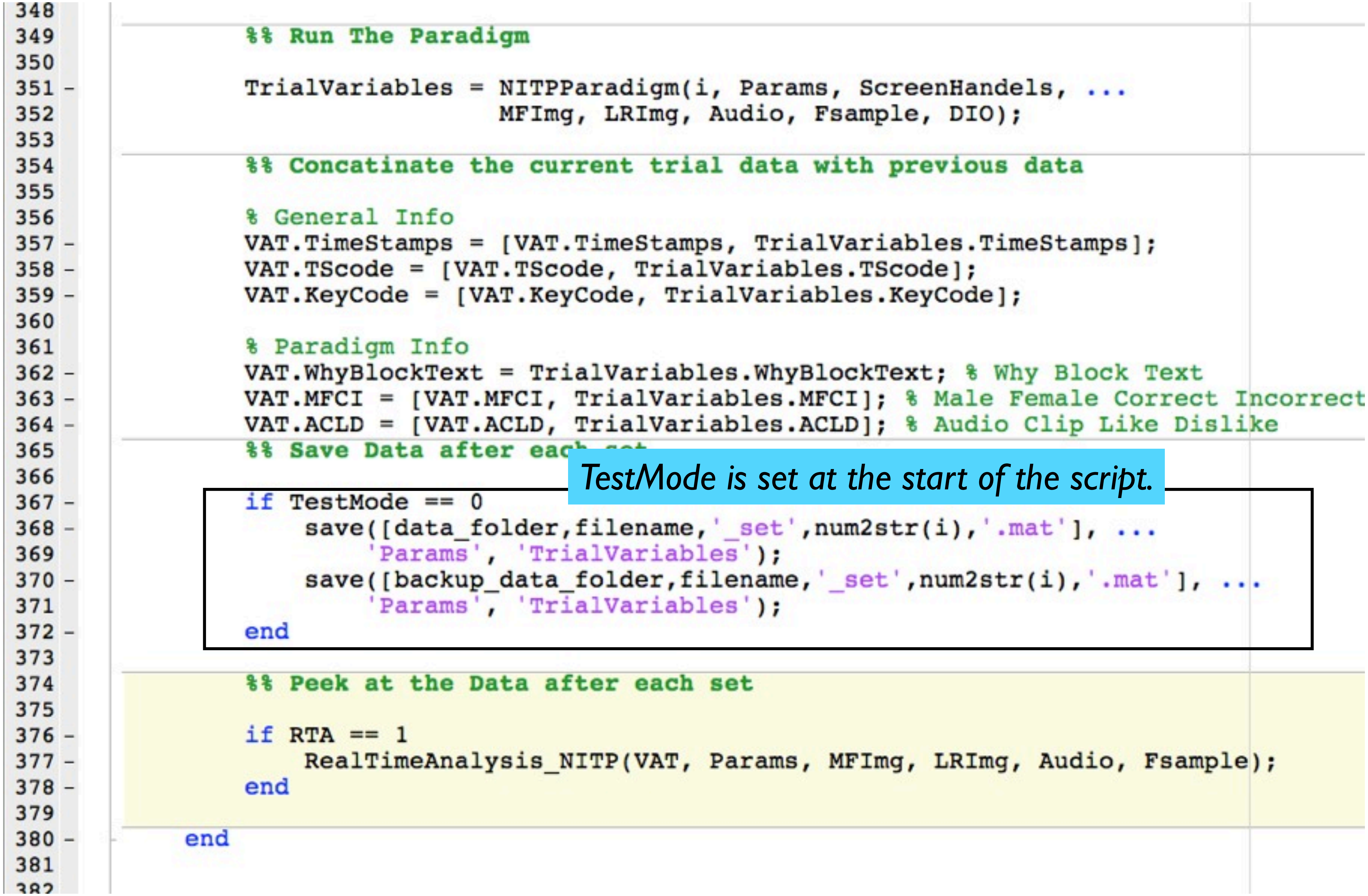
1. You have to decode these codes.



```
        %% Run The Paradigm

        TrialVariables = NITPParadigm(i, Params, ScreenHandels, ...
                         MFImg, LRImg, Audio, Fsample, DIO);

        %% Concatinate the current trial data with previous data

        % General Info
        VAT.TimeStamps = [VAT.TimeStamps, TrialVariables.TimeStamps];
        VAT.TScode = [VAT.TScode, TrialVariables.TScode];
        VAT.KeyCode = [VAT.KeyCode, TrialVariables.KeyCode];

        % Paradigm Info
        VAT.WhyBlockText = TrialVariables.WhyBlockText; % Why Block Text
        VAT.MFCI = [VAT.MFCI, TrialVariables.MFCI]; % Male Female Correct Incorrect
        VAT.ACLD = [VAT.ACLD, TrialVariables.ACLD]; % Audio Clip Like Dislike
        %% Save Data after each set

        if TestMode == 0
            save([data_folder,filename,'_set',num2str(i),'.mat'], ...
                'Params', 'TrialVariables');
            save([backup_data_folder,filename,'_set',num2str(i),'.mat'], ...
                'Params', 'TrialVariables');
        end

        %% Peek at the Data after each set

        if RTA == 1
            RealTimeAnalysis_NITP(VAT, Params, MFImg, LRImg, Audio, Fsample);
        end

    end
```

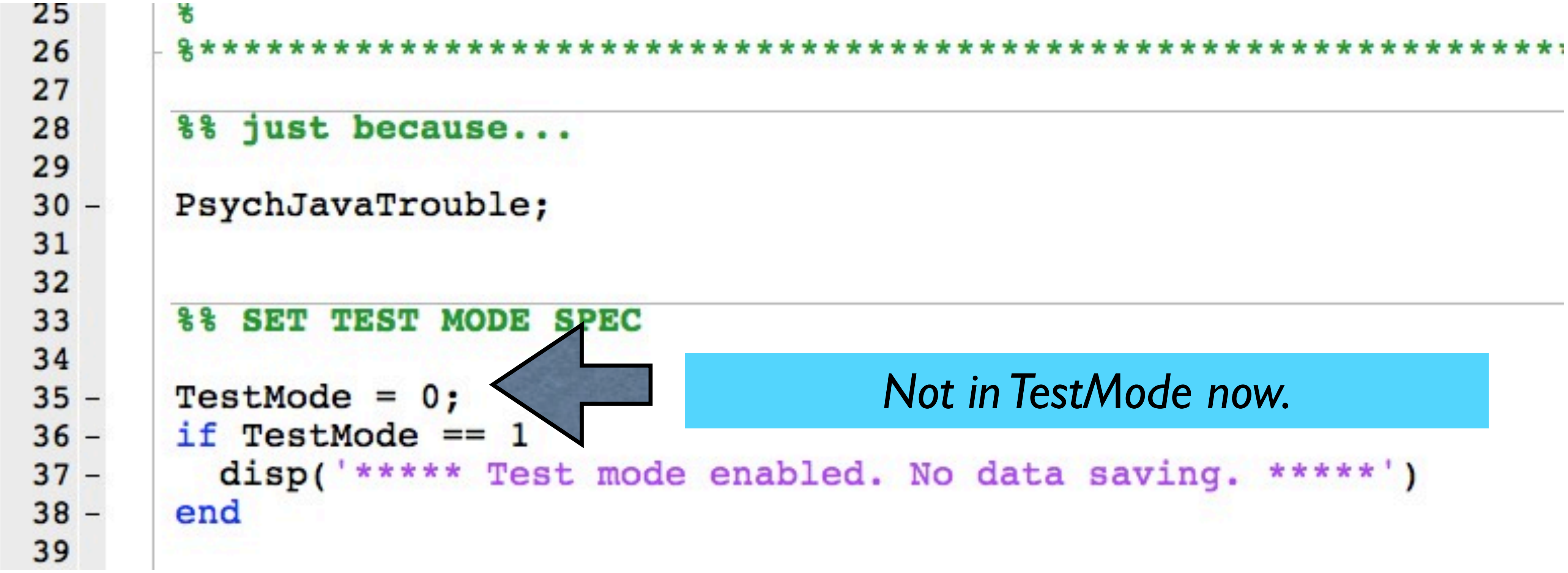
Your task - figure out how to analyze these data. Hint - Work through the RealTimeAnalysis_NITP.m script to see how Cameron does it.

2. If you don't want to save output change TestMode to 1.



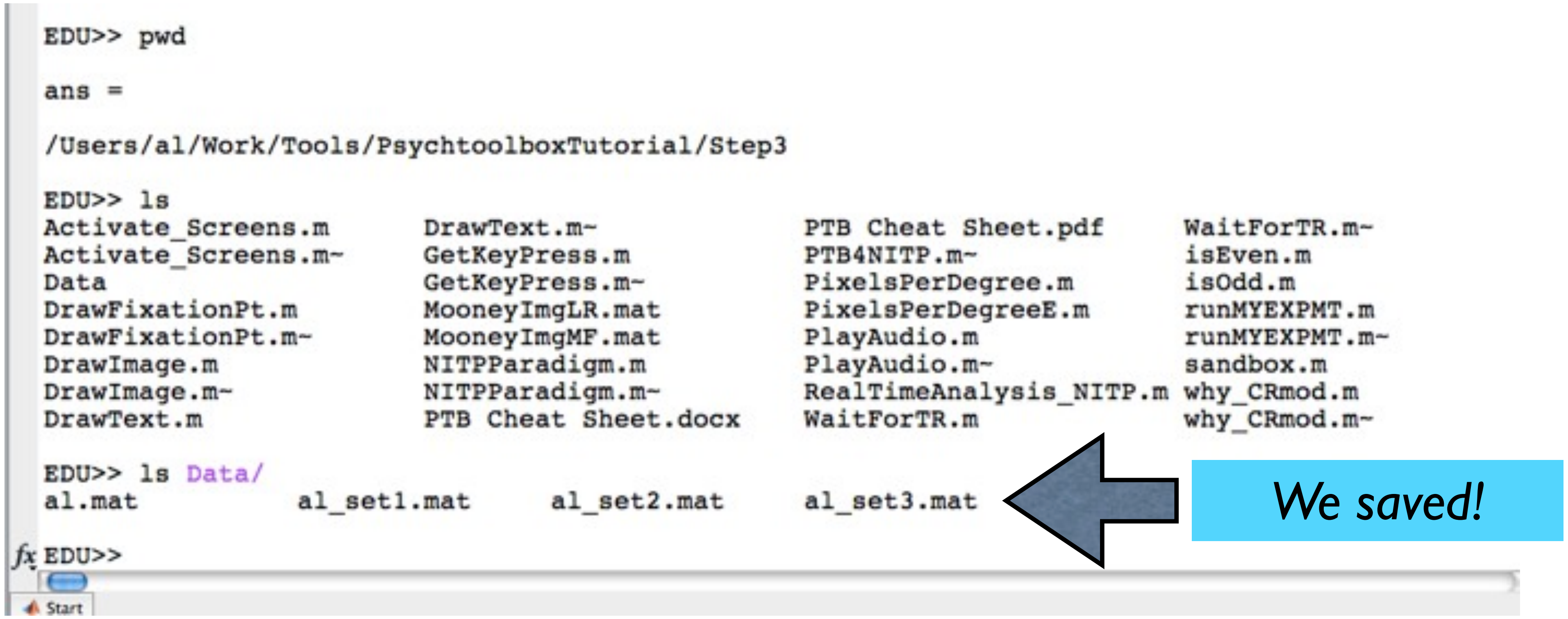
```
        %% Run The Paradigm

        TrialVariables = NITPParadigm(i, Params, ScreenHandels, ...
                         MFImg, LRImg, Audio, Fsample, DIO);

        %% Concatinate the current trial data with previous data

        % General Info
        VAT.TimeStamps = [VAT.TimeStamps, TrialVariables.TimeStamps];
        VAT.TScode = [VAT.TScode, TrialVariables.TScode];
        VAT.KeyCode = [VAT.KeyCode, TrialVariables.KeyCode];

        % Paradigm Info
        VAT.WhyBlockText = TrialVariables.WhyBlockText; % Why Block Text
        VAT.MFCI = [VAT.MFCI, TrialVariables.MFCI]; % Male Female Correct Incorrect
        VAT.ACLD = [VAT.ACLD, TrialVariables.ACLD]; % Audio Clip Like Dislike
        %% Save Data after each set

        if TestMode == 0
            save([data_folder,filename,'_set',num2str(i),'.mat'], ...
                'Params', 'TrialVariables');
            save([backup_data_folder,filename,'_set',num2str(i),'.mat'], ...
                'Params', 'TrialVariables');
        end

        %% Peek at the Data after each set

        if RTA == 1
            RealTimeAnalysis_NITP(VAT, Params, MFImg, LRImg, Audio, Fsample);
        end

    end
```

TestMode is set at the start of the script.



```
%
%*********************************************************

%% just because...

PsychJavaTrouble;


%% SET TEST MODE SPEC

TestMode = 0;
if TestMode == 1
  disp('***** Test mode enabled. No data saving. *****')
end

```

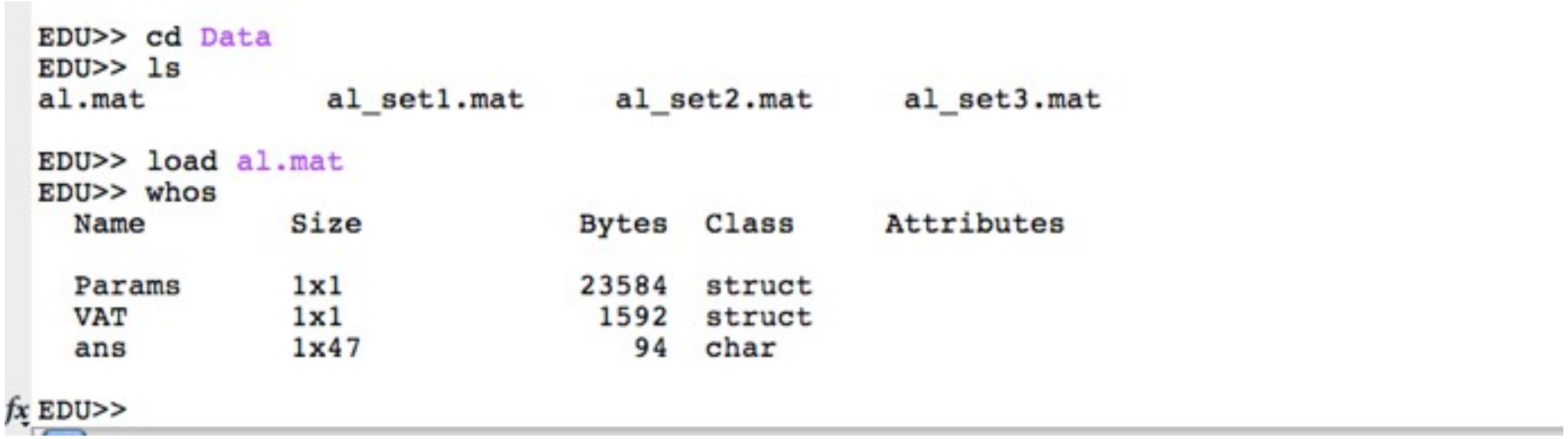
Not in TestMode now.



```
EDU>> pwd

ans =

/Users/al/Work/Tools/PsychtoolboxTutorial/Step3

EDU>> ls
Activate_Screens.m      DrawText.m~             PTB Cheat Sheet.pdf     WaitForTR.m~
Activate_Screens.m~     GetKeyPress.m           PTB4NITP.m~             isEven.m
Data                    GetKeyPress.m~          PixelsPerDegree.m       isOdd.m
DrawFixationPt.m        MooneyImgLR.mat         PixelsPerDegreeE.m      runMYEXPMT.m
DrawFixationPt.m~       MooneyImgMF.mat         PlayAudio.m             runMYEXPMT.m~
DrawImage.m             NITPParadigm.m          PlayAudio.m~            sandbox.m
DrawImage.m~            NITPParadigm.m~         RealTimeAnalysis_NITP.m why_CRmod.m
DrawText.m              PTB Cheat Sheet.docx    WaitForTR.m             why_CRmod.m~

EDU>> ls Data/
al.mat          al_set1.mat     al_set2.mat     al_set3.mat

EDU>>
```

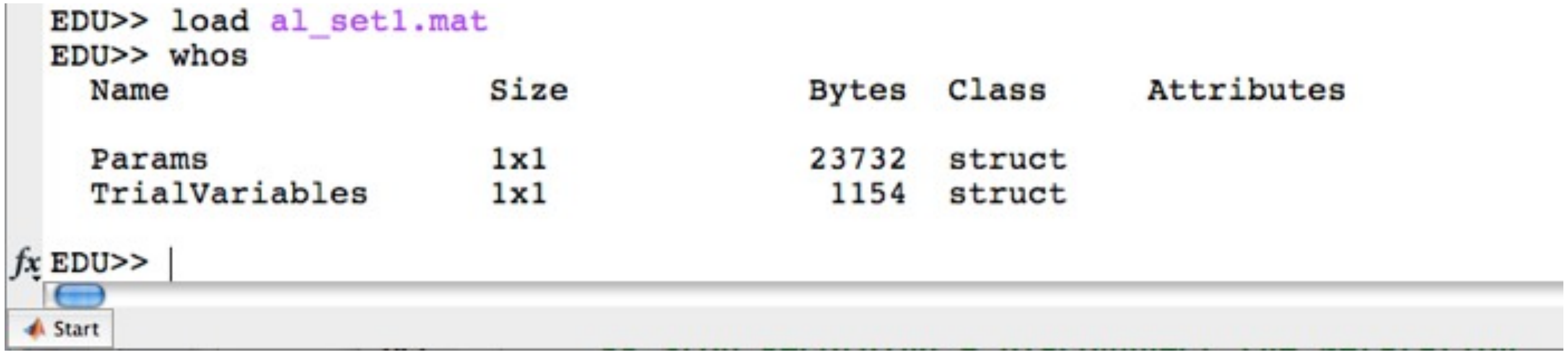
We saved!

Let’s look inside these files



```
EDU>> cd Data
EDU>> ls
al.mat              al_set1.mat         al_set2.mat         al_set3.mat

EDU>> load al.mat
EDU>> whos
  Name            Size                Bytes  Class     Attributes

  Params          1x1                 23584  struct
  VAT             1x1                  1592  struct
  ans             1x47                   94  char

EDU>>
```

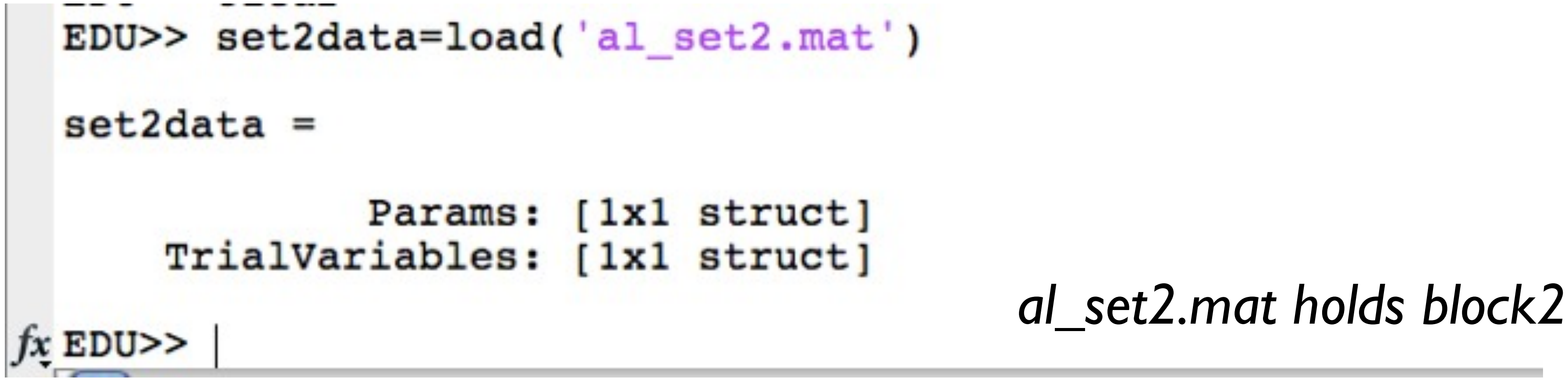
al.mat holds block variables - VAT gets passed into RealTime. You can ignore it for the most part.



```
EDU>> load al_set1.mat
EDU>> whos
  Name                    Size                Bytes  Class     Attributes

  Params                  1x1                 23732  struct
  TrialVariables          1x1                  1154  struct

EDU>> |
```

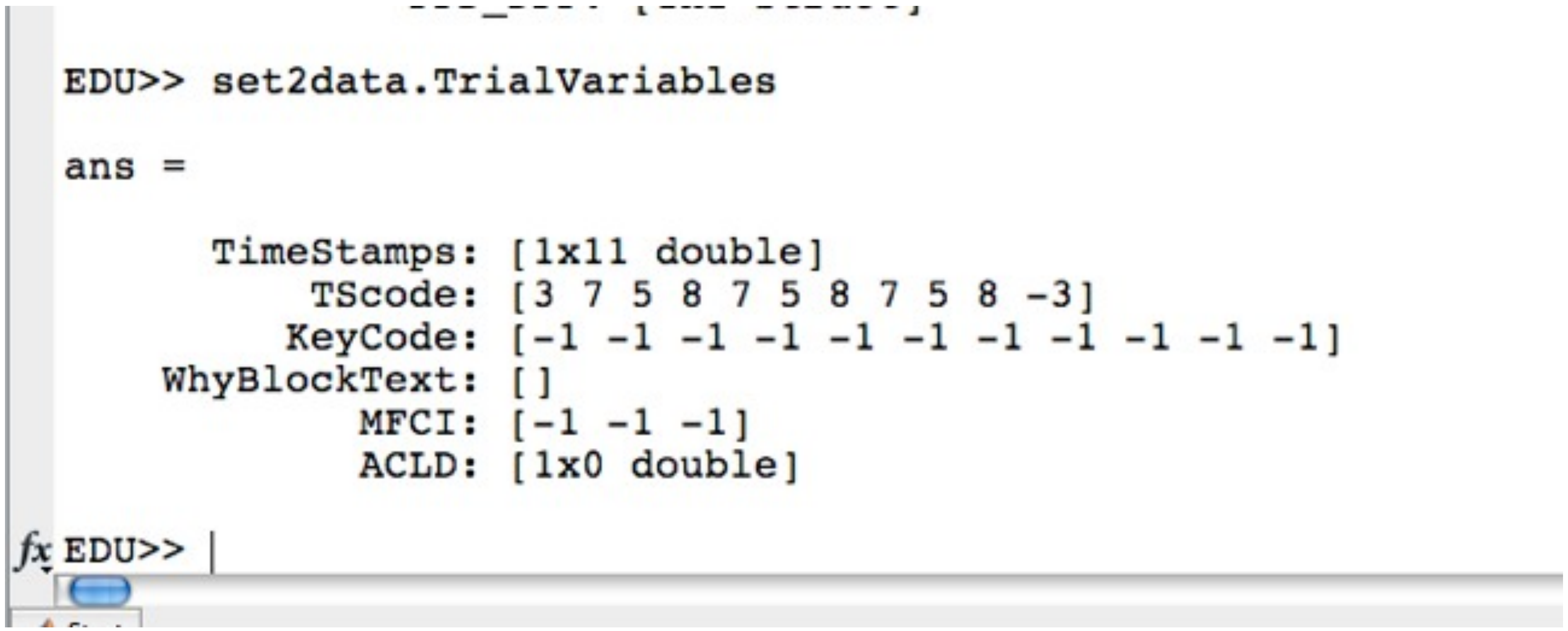
al_set1.mat holds block1 variables.

Let’s look inside these files



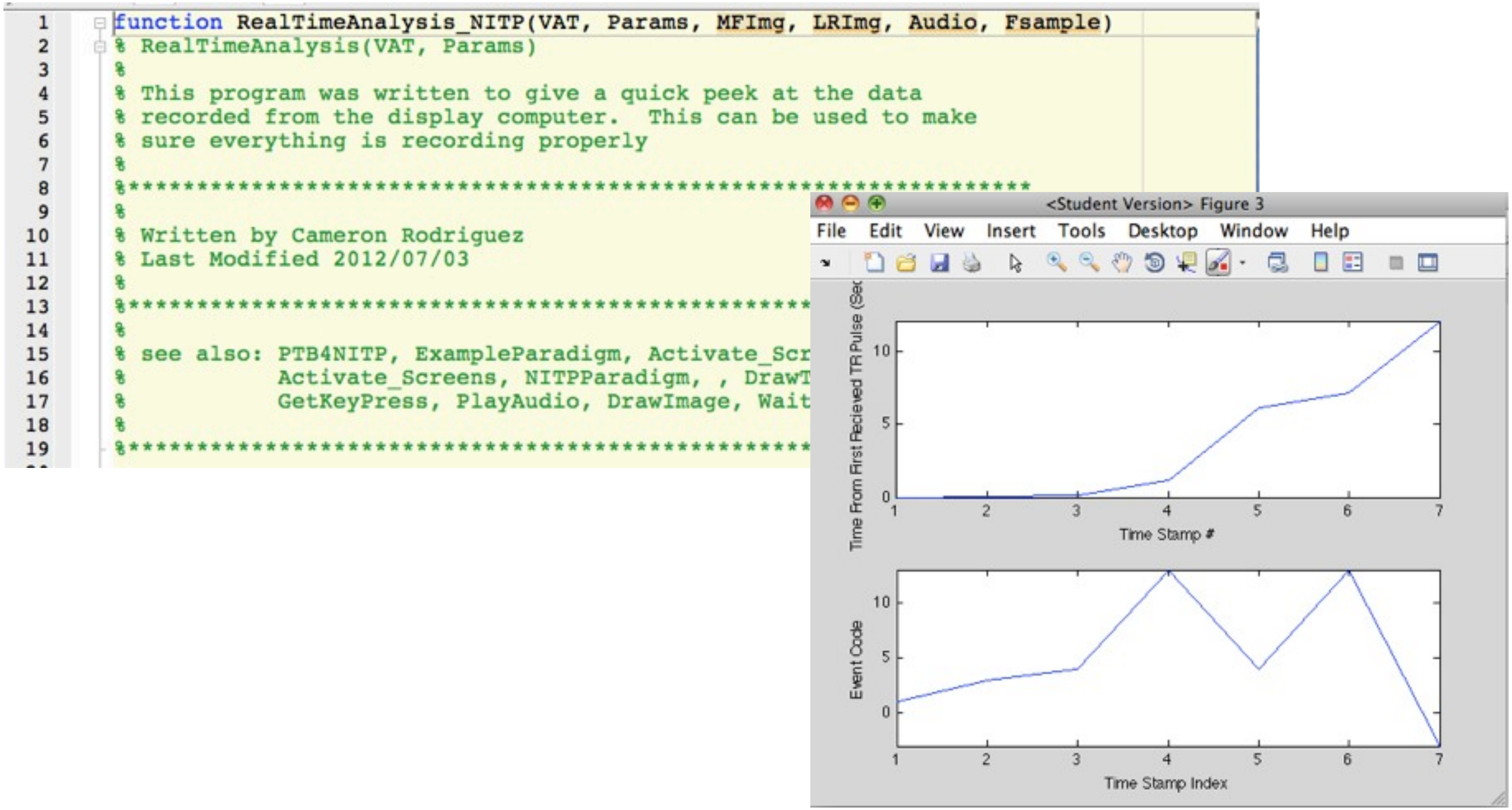
```
EDU>> set2data=load('al_set2.mat')

set2data = 

           Params: [1x1 struct]
   TrialVariables: [1x1 struct]

EDU>> 
```

al_set2.mat holds block2 variables.



```
EDU>> set2data.TrialVariables

ans = 

      TimeStamps: [1x11 double]
          TScode: [3 7 5 8 7 5 8 7 5 8 -3]
         KeyCode: [-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]
    WhyBlockText: []
            MFCI: [-1 -1 -1]
            ACLD: [1x0 double]

EDU>> 
```

Again - to figure out how to analyze such variables you can look into RealTimeAnalysis*.mat



```
function RealTimeAnalysis_NITP(VAT, Params, MFImg, LRImg, Audio, Fsample)
% RealTimeAnalysis(VAT, Params)
%
% This program was written to give a quick peek at the data
% recorded from the display computer.  This can be used to make
% sure everything is recording properly
%
%*******************************************************************
%
% Written by Cameron Rodriguez
% Last Modified 2012/07/03
%
%****************************************************
%
% see also: PTB4NITP, ExampleParadigm, Activate_Scr
%           Activate_Screens, NITPParadigm, , DrawT
%           GetKeyPress, PlayAudio, DrawImage, Wait
%
%****************************************************
```

There are other ways of saving data (e.g., into a text file) - see Step2 tutorial.

step 8 <<<< change your design

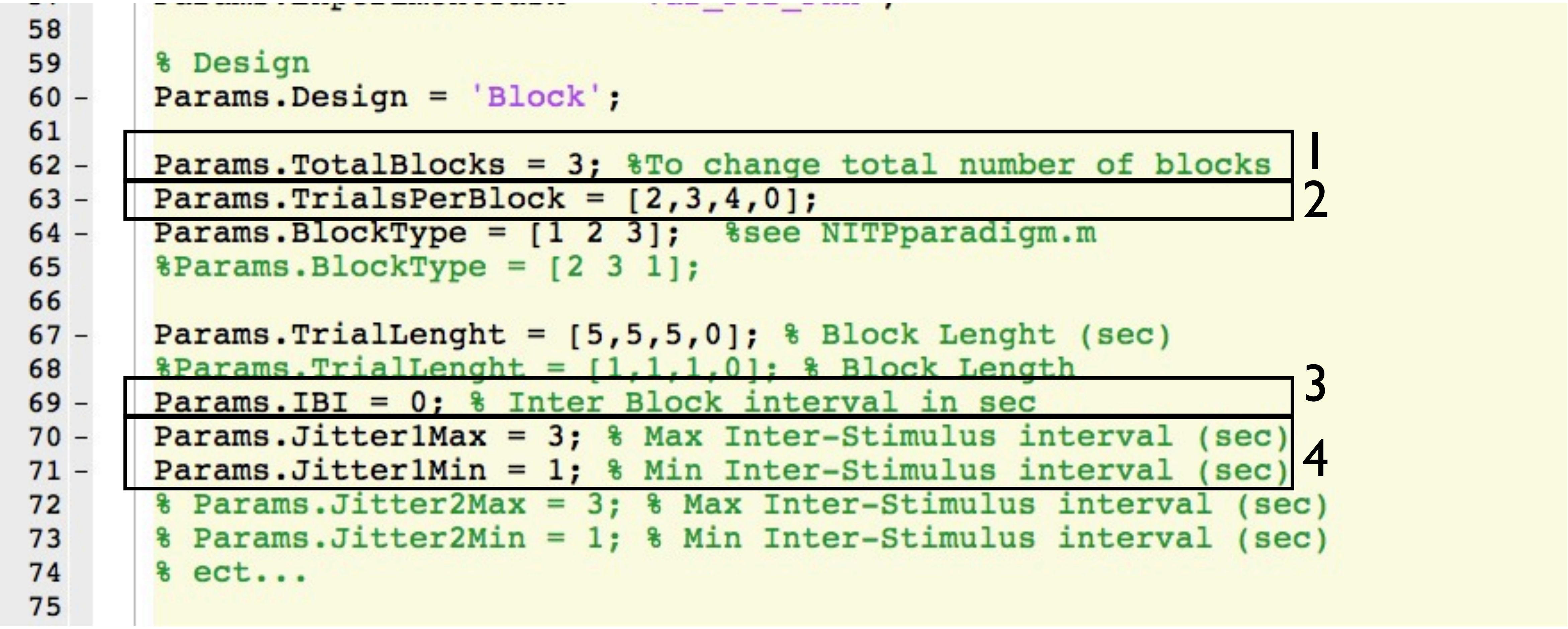
You may want to change how many mini-blocks to show in a block, how many trials, timing.

1 RUN of CODE = x 'mini-blocks' (BLOCKS)

~ one option is to set BLOCKS to 1 and have many many trials

~ another option is to have mini-blocks within this RUN (as we did here)

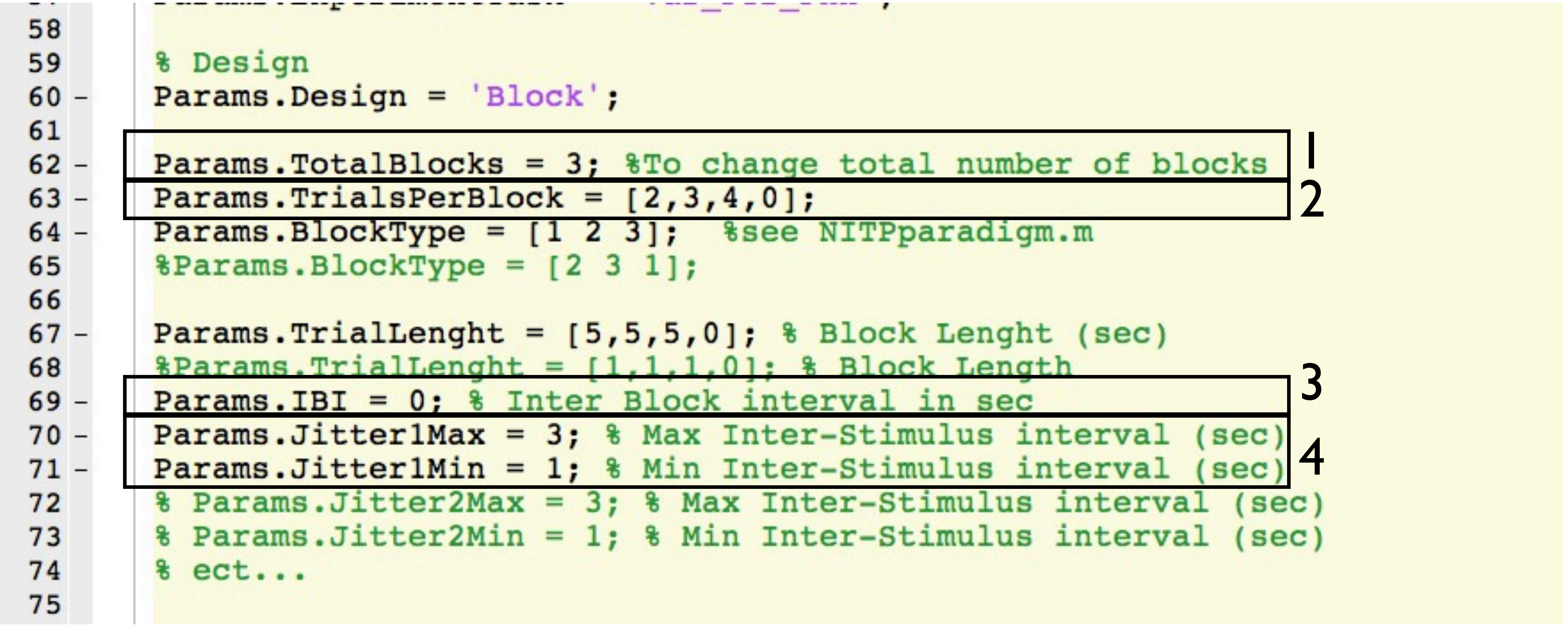
This is set in the initial lines of code of runEXPMT.m



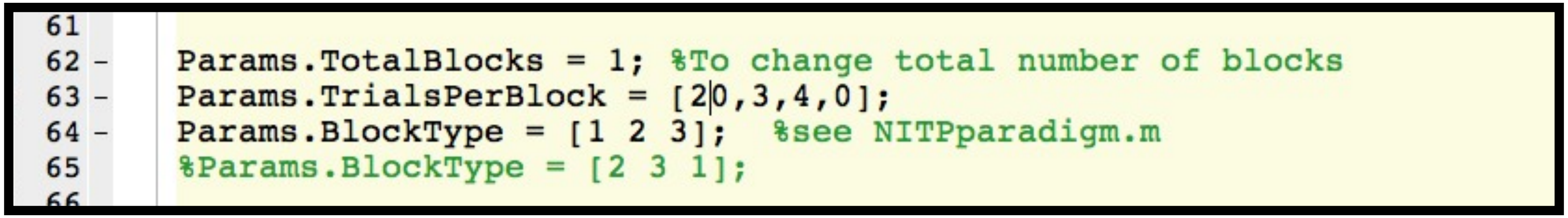
```
% Design
Params.Design = 'Block';

Params.TotalBlocks = 3; %To change total number of blocks
Params.TrialsPerBlock = [2,3,4,0];
Params.BlockType = [1 2 3];  %see NITPparadigm.m
%Params.BlockType = [2 3 1];

Params.TrialLenght = [5,5,5,0]; % Block Lenght (sec)
%Params.TrialLenght = [1,1,1,0]; % Block Length
Params.IBI = 0; % Inter Block interval in sec
Params.Jitter1Max = 3; % Max Inter-Stimulus interval (sec)
Params.Jitter1Min = 1; % Min Inter-Stimulus interval (sec)
% Params.Jitter2Max = 3; % Max Inter-Stimulus interval (sec)
% Params.Jitter2Min = 1; % Min Inter-Stimulus interval (sec)
% ect...
```

1. Set how many ‘mini’ blocks to present in run (currently 3).
2. Set how many trials in each block.
3. Set rest time to fit in between blocks.
4. Jitter to place b/w trials (you figure it out).

This is set in the initial lines of code of runEXPMT.m



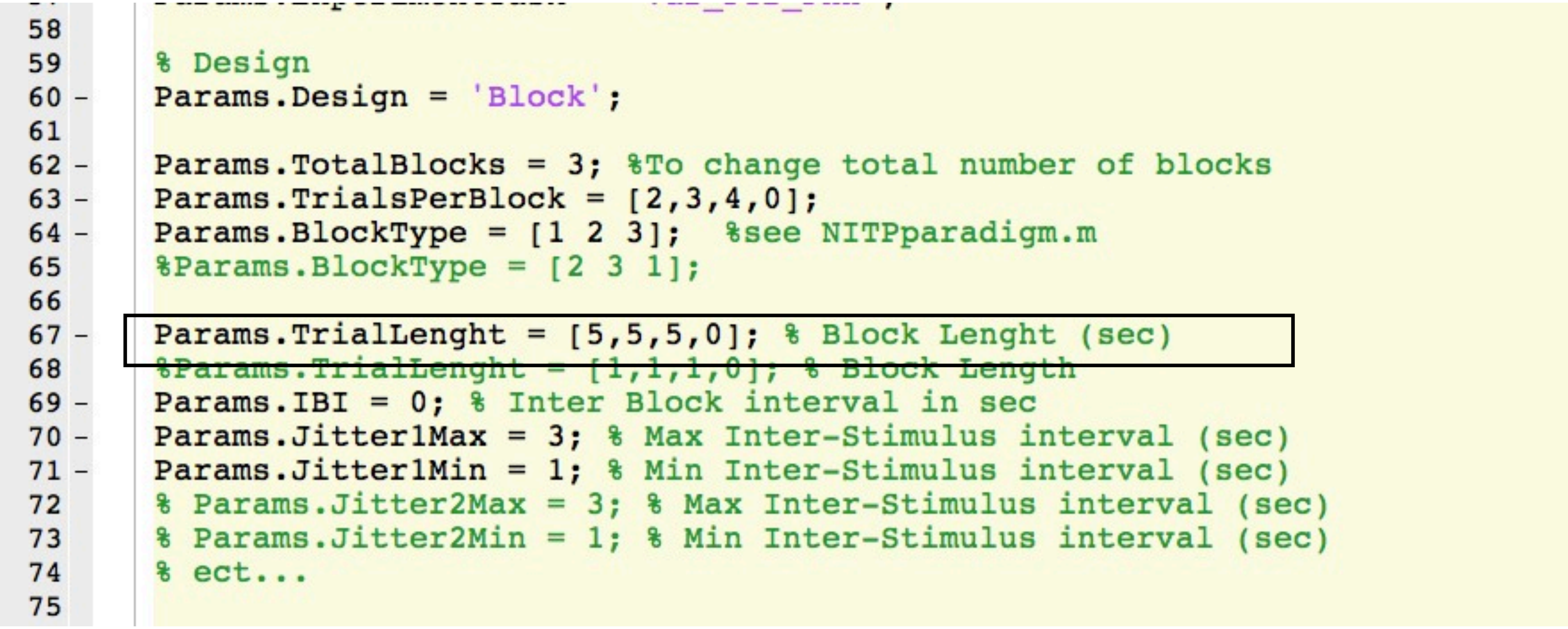
```

% Design
Params.Design = 'Block';

Params.TotalBlocks = 3; %To change total number of blocks
Params.TrialsPerBlock = [2,3,4,0];
Params.BlockType = [1 2 3];  %see NITPparadigm.m
%Params.BlockType = [2 3 1];

Params.TrialLenght = [5,5,5,0]; % Block Lenght (sec)
%Params.TrialLenght = [1,1,1,0]; % Block Length
Params.IBI = 0; % Inter Block interval in sec
Params.Jitter1Max = 3; % Max Inter-Stimulus interval (sec)
Params.Jitter1Min = 1; % Min Inter-Stimulus interval (sec)
% Params.Jitter2Max = 3; % Max Inter-Stimulus interval (sec)
% Params.Jitter2Min = 1; % Min Inter-Stimulus interval (sec)
% ect...

```



```

Params.TotalBlocks = 1; %To change total number of blocks
Params.TrialsPerBlock = [20,3,4,0];
Params.BlockType = [1 2 3];  %see NITPparadigm.m
%Params.BlockType = [2 3 1];
```

1 block with 20 trials

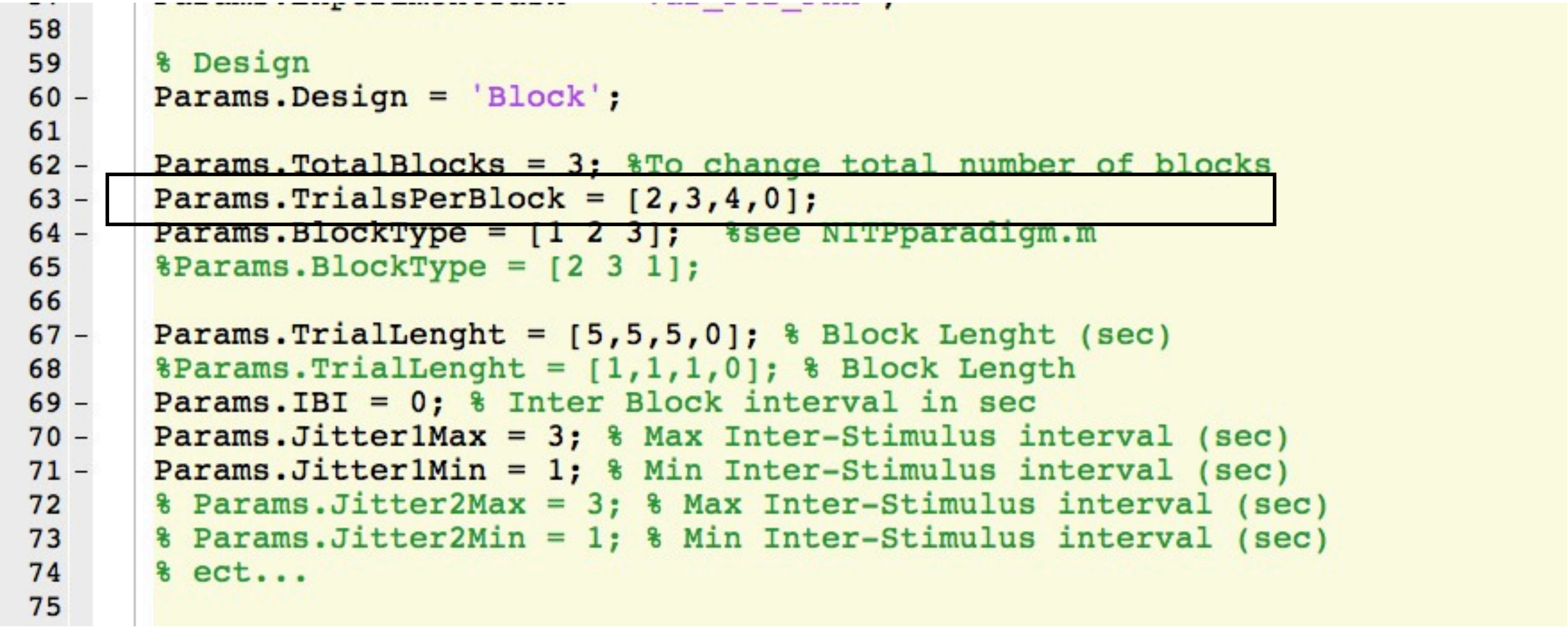
What about number of trials....



```
% Design
Params.Design = 'Block';

Params.TotalBlocks = 3; %To change total number of blocks
Params.TrialsPerBlock = [2,3,4,0];
Params.BlockType = [1 2 3];  %see NITPparadigm.m
%Params.BlockType = [2 3 1];

Params.TrialLenght = [5,5,5,0]; % Block Lenght (sec)
%Params.TrialLenght = [1,1,1,0]; % Block Length
Params.IBI = 0; % Inter Block interval in sec
Params.Jitter1Max = 3; % Max Inter-Stimulus interval (sec)
Params.Jitter1Min = 1; % Min Inter-Stimulus interval (sec)
% Params.Jitter2Max = 3; % Max Inter-Stimulus interval (sec)
% Params.Jitter2Min = 1; % Min Inter-Stimulus interval (sec)
% ect...
```

trial length we spoke about but that’s different than #trials

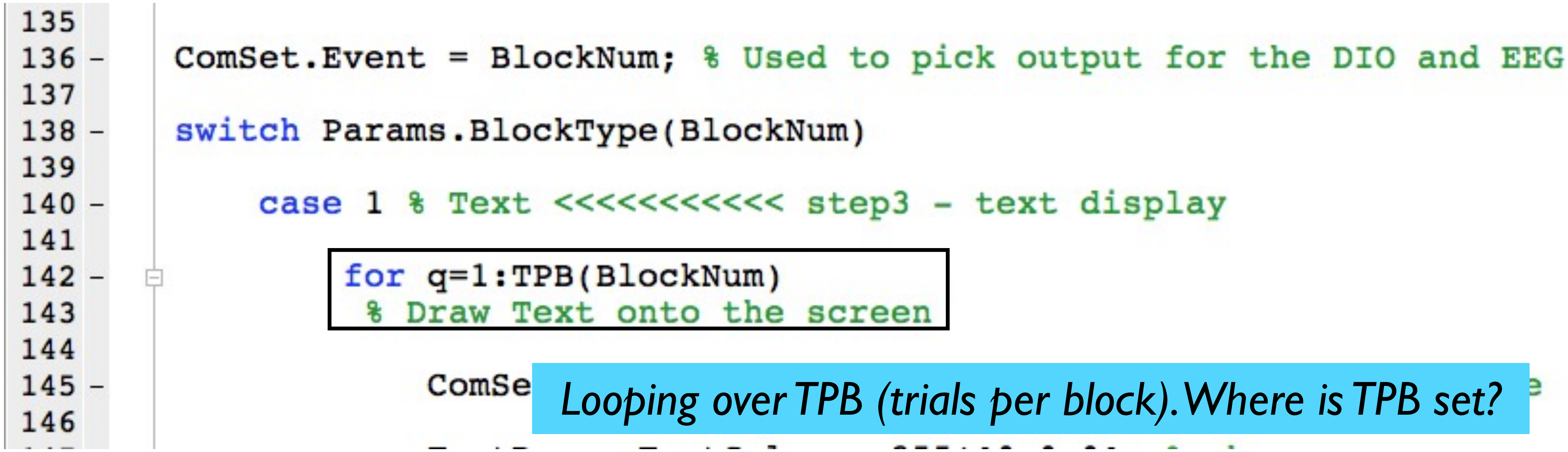
What about number of trials....



```
% Design
Params.Design = 'Block';

Params.TotalBlocks = 3; %To change total number of blocks
Params.TrialsPerBlock = [2,3,4,0];
Params.BlockType = [1 2 3];  %see NITPparadigm.m
%Params.BlockType = [2 3 1];

Params.TrialLenght = [5,5,5,0]; % Block Lenght (sec)
%Params.TrialLenght = [1,1,1,0]; % Block Length
Params.IBI = 0; % Inter Block interval in sec
Params.Jitter1Max = 3; % Max Inter-Stimulus interval (sec)
Params.Jitter1Min = 1; % Min Inter-Stimulus interval (sec)
% Params.Jitter2Max = 3; % Max Inter-Stimulus interval (sec)
% Params.Jitter2Min = 1; % Min Inter-Stimulus interval (sec)
% ect...
```

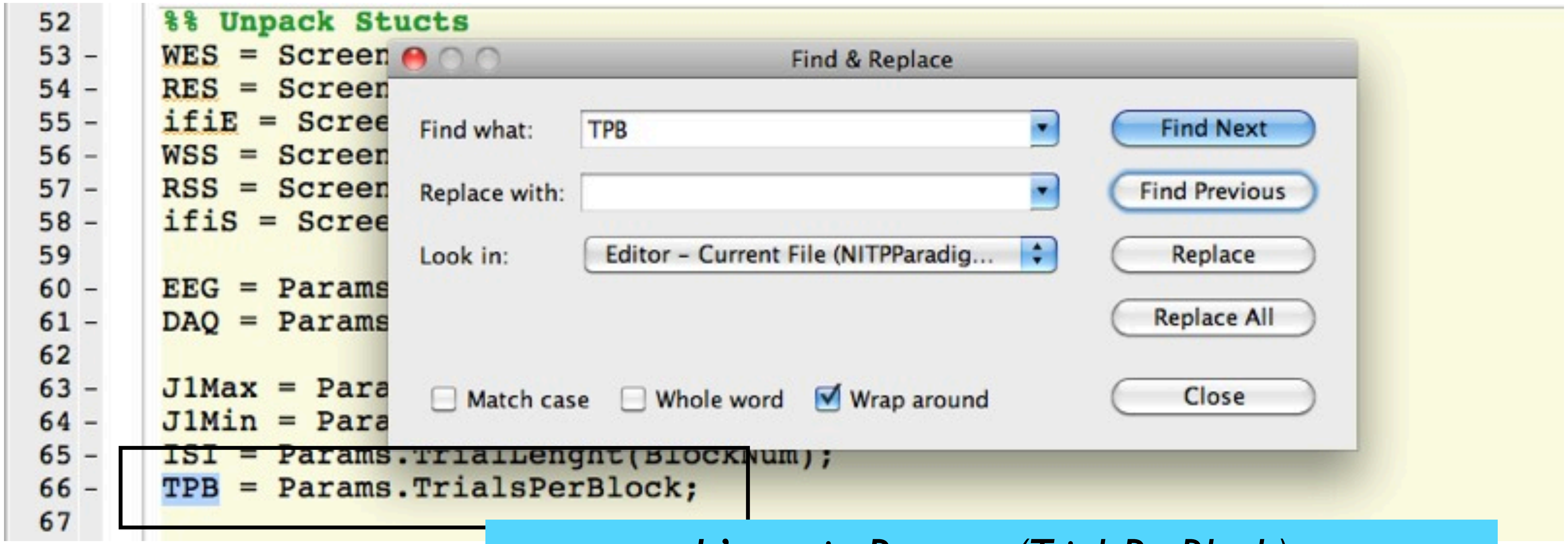
Specify #trials for each block.

Trials are looped over in NITPparadigm.m



```
ComSet.Event = BlockNum; % Used to pick output for the DIO and EEG

switch Params.BlockType(BlockNum)

    case 1 % Text <<<<<<<<<<< step3 - text display

        for q=1:TPB(BlockNum)
         % Draw Text onto the screen

            ComSe
```

Looping over TPB (trials per block). Where is TPB set?



```
%% Unpack Stucts
WES = Screen
RES = Screen
ifiE = Scree
WSS = Screen
RSS = Screen
ifiS = Scree

EEG = Params
DAQ = Params

J1Max = Para
J1Min = Para
ISI = Params.TrialLenght(BlockNum);
TPB = Params.TrialsPerBlock;
```

Find & Replace

Find what: TPB

Replace with:

Look in: Editor - Current File (NITPParadig...

☐ Match case ☐ Whole word ☑ Wrap around

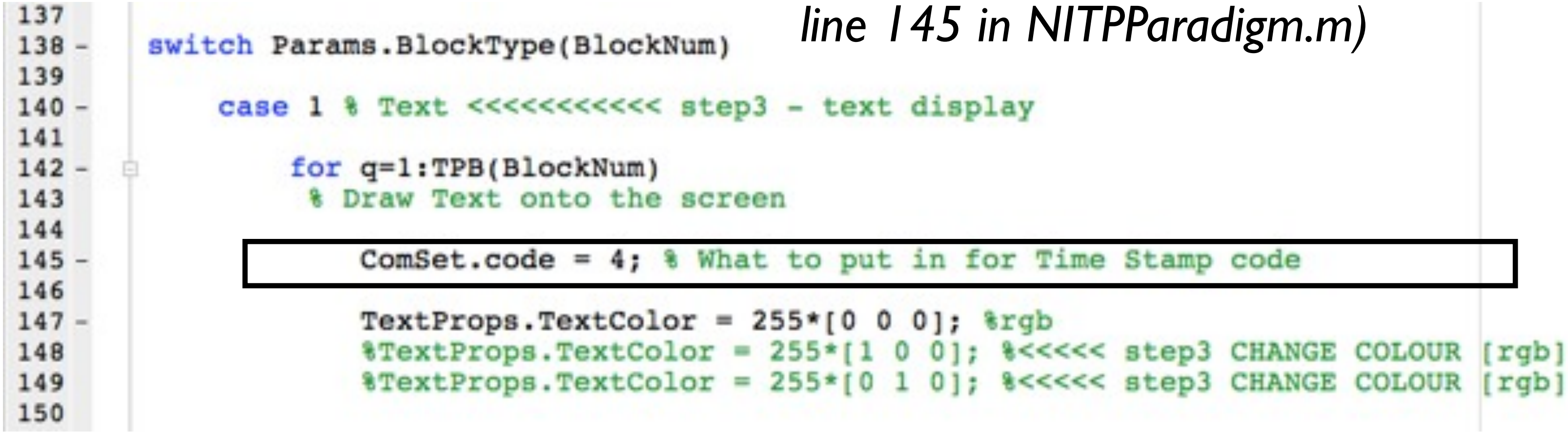
It's set in Params (TrialsPerBlock)

Additional tasks you might want to try:

1. Where is the trigger screen created and how does it work? (hint - line 125 in the NITPparadigm.m code)

2. What are all these other parameters (PARAMS.xx) that you can modify? What other parameter might you add? Can you get some of these from user input? e.g., How would you change between goggles/screen? (hint - line 35 or so in runMYEXPMT.m)

3. What is the meaning of all the event codes? *(Hint - codes 'type' of event – see line 135 in runEXPMT.m & line 145 in NITPParadigm.m)*



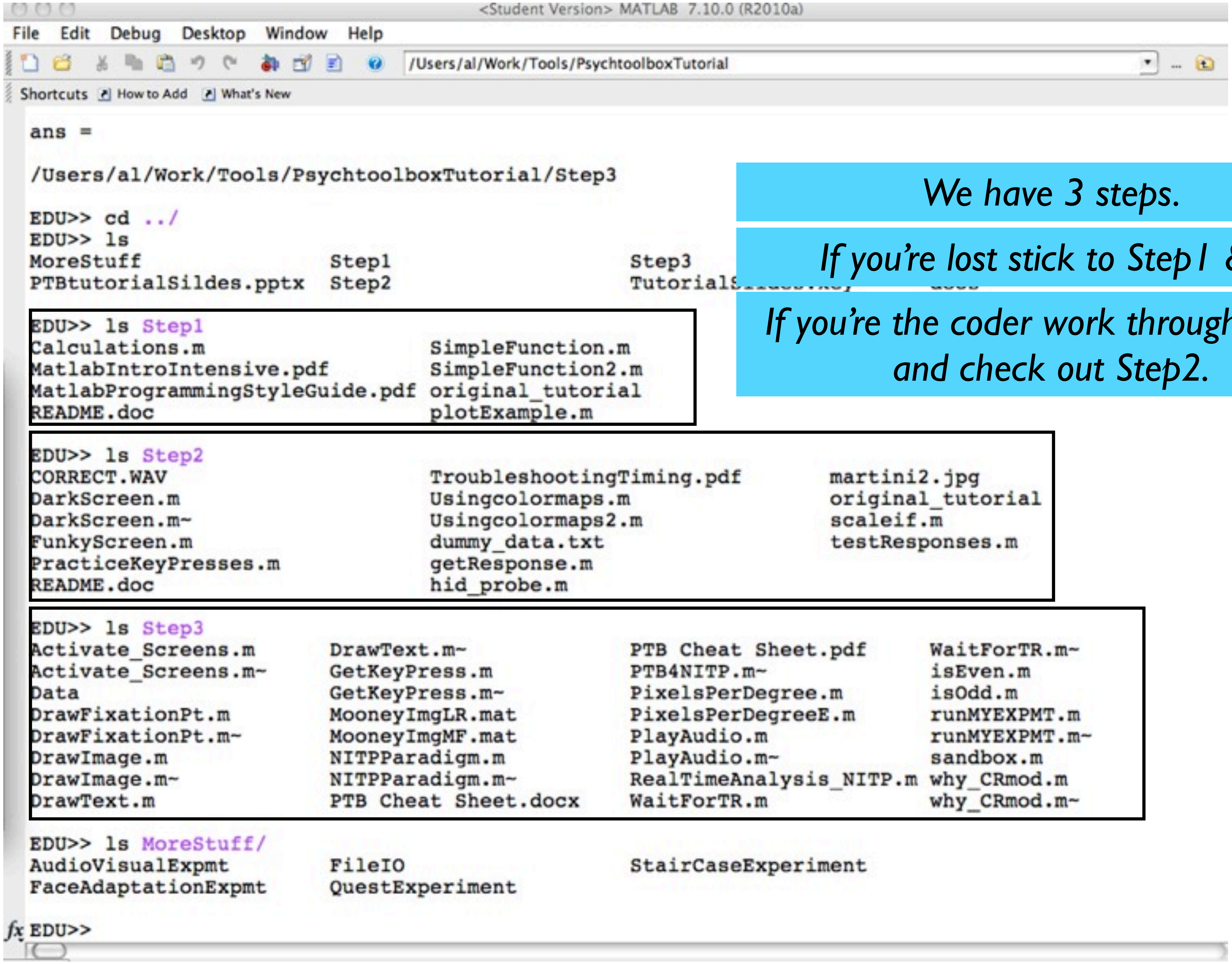
```
137
138 -  switch Params.BlockType(BlockNum)
139
140 -      case 1 % Text <<<<<<<<<<< step3 - text display
141
142 -          for q=1:TPB(BlockNum)
143             % Draw Text onto the screen
144
145 -              ComSet.code = 4; % What to put in for Time Stamp code
146
147 -              TextProps.TextColor = 255*[0 0 0]; %rgb
148                %TextProps.TextColor = 255*[1 0 0]; %<<<<< step3 CHANGE COLOUR [rgb]
149                %TextProps.TextColor = 255*[0 1 0]; %<<<<< step3 CHANGE COLOUR [rgb]
150
```

4. Find the command (in runMYEXPMT.m) that hides and restores the cursor.

5. Find where the mini-analysis script gets called (~line 327 in runMYEXPMT.m), look inside to how Cameron analyzes the outputs to find the timing of events.

6. Get the 'help' for 'try' (i.e., help try) in the Matlab command window. What is the purpose of this control statement? Trace out where it's implemented in runMYEXPMT.m (check out Step2 for more info on this safety mechanism)

GUIDELINES


<Student Version> MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/Users/al/Work/Tools/PsychtoolboxTutorial
Shortcuts
How to Add
What's New

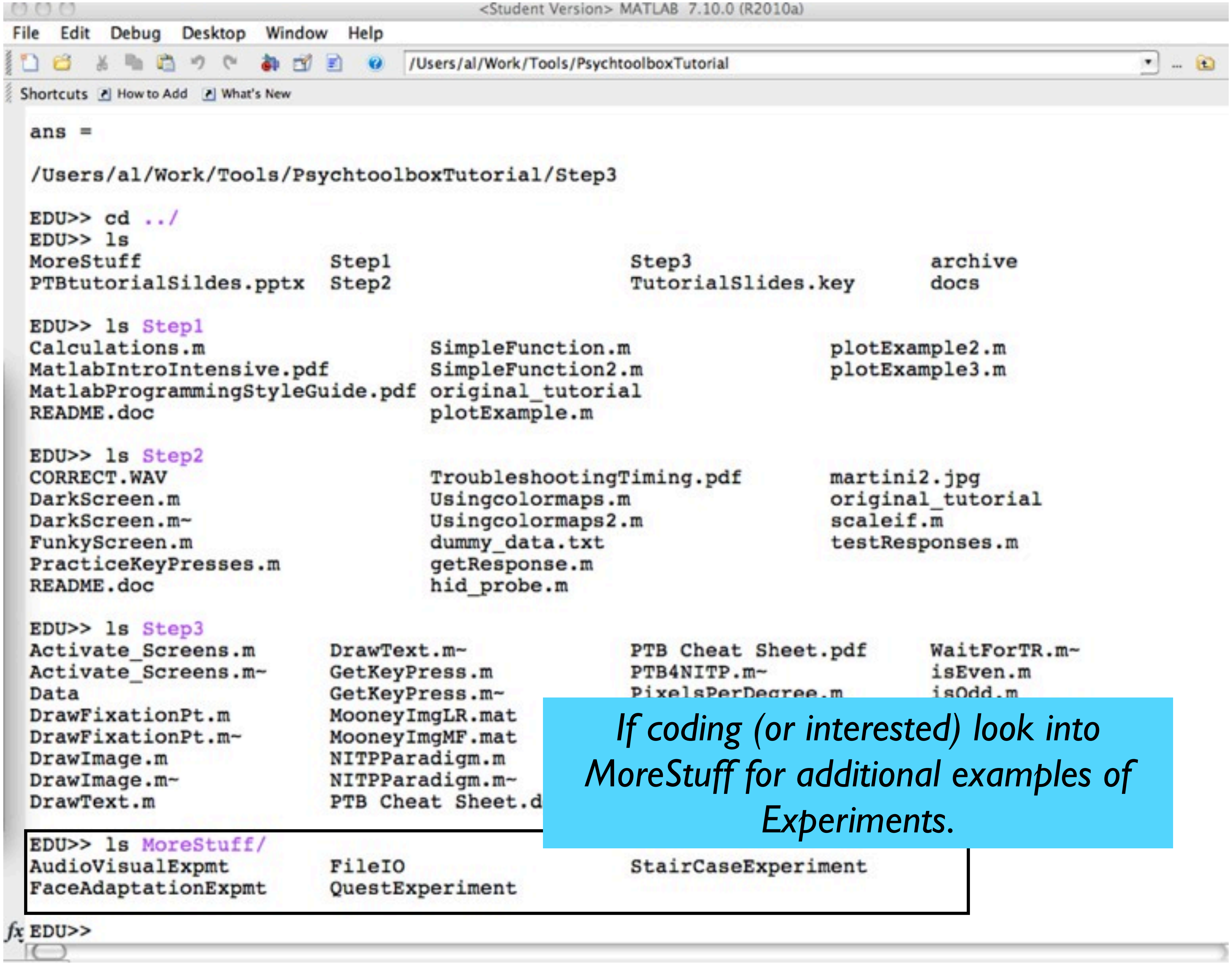
```
ans =

/Users/al/Work/Tools/PsychtoolboxTutorial/Step3

EDU>> cd ../
EDU>> ls
MoreStuff               Step1                   Step3
PTBtutorialSildes.pptx  Step2                   TutorialS

EDU>> ls Step1
Calculations.m                  SimpleFunction.m
MatlabIntroIntensive.pdf        SimpleFunction2.m
MatlabProgrammingStyleGuide.pdf original_tutorial
README.doc                      plotExample.m

EDU>> ls Step2
CORRECT.WAV                     TroubleshootingTiming.pdf       martini2.jpg
DarkScreen.m                    Usingcolormaps.m                original_tutorial
DarkScreen.m~                   Usingcolormaps2.m               scaleif.m
FunkyScreen.m                   dummy_data.txt                  testResponses.m
PracticeKeyPresses.m            getResponse.m
README.doc                      hid_probe.m

EDU>> ls Step3
Activate_Screens.m      DrawText.m~             PTB Cheat Sheet.pdf     WaitForTR.m~
Activate_Screens.m~     GetKeyPress.m           PTB4NITP.m~             isEven.m
Data                    GetKeyPress.m~          PixelsPerDegree.m       isOdd.m
DrawFixationPt.m        MooneyImgLR.mat         PixelsPerDegreeE.m      runMYEXPMT.m
DrawFixationPt.m~       MooneyImgMF.mat         PlayAudio.m             runMYEXPMT.m~
DrawImage.m             NITPParadigm.m          PlayAudio.m~            sandbox.m
DrawImage.m~            NITPParadigm.m~         RealTimeAnalysis_NITP.m why_CRmod.m
DrawText.m              PTB Cheat Sheet.docx    WaitForTR.m             why_CRmod.m~

EDU>> ls MoreStuff/
AudioVisualExpmt        FileIO                  StairCaseExperiment
FaceAdaptationExpmt     QuestExperiment

fx EDU>>
```

We have 3 steps.

If you're lost stick to Step1 & 2

If you're the coder work through Step3 and check out Step2.

GUIDELINES

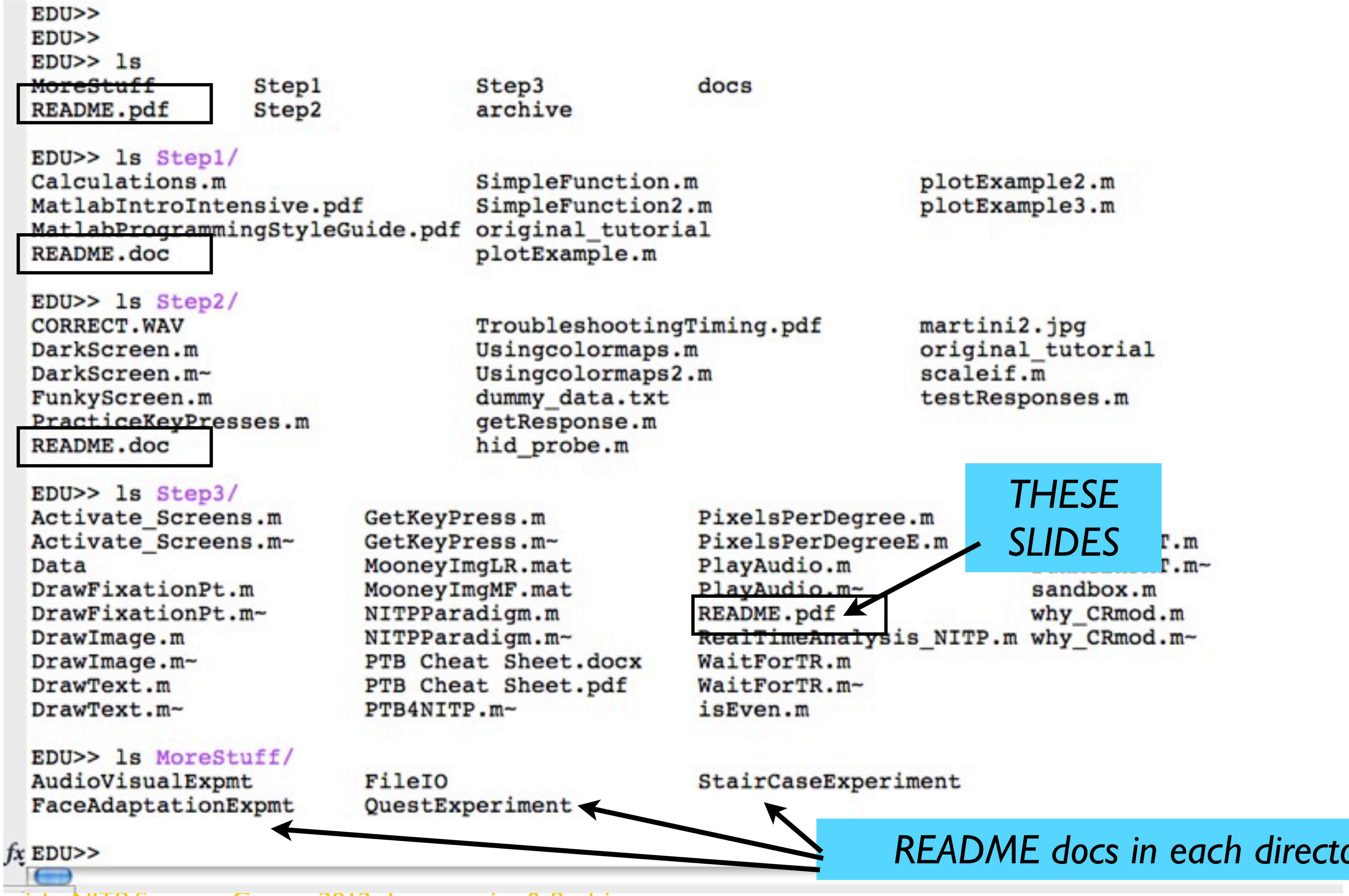

<Student Version> MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/Users/al/Work/Tools/PsychtoolboxTutorial
Shortcuts How to Add What's New

```
ans =

/Users/al/Work/Tools/PsychtoolboxTutorial/Step3

EDU>> cd ../
EDU>> ls
MoreStuff                Step1                    Step3                    archive
PTBtutorialSildes.pptx   Step2                    TutorialSlides.key       docs

EDU>> ls Step1
Calculations.m                  SimpleFunction.m                plotExample2.m
MatlabIntroIntensive.pdf        SimpleFunction2.m               plotExample3.m
MatlabProgrammingStyleGuide.pdf original_tutorial
README.doc                      plotExample.m

EDU>> ls Step2
CORRECT.WAV                     TroubleshootingTiming.pdf       martini2.jpg
DarkScreen.m                    Usingcolormaps.m                original_tutorial
DarkScreen.m~                   Usingcolormaps2.m               scaleif.m
FunkyScreen.m                   dummy_data.txt                  testResponses.m
PracticeKeyPresses.m            getResponse.m
README.doc                      hid_probe.m

EDU>> ls Step3
Activate_Screens.m       DrawText.m~              PTB Cheat Sheet.pdf      WaitForTR.m~
Activate_Screens.m~      GetKeyPress.m            PTB4NITP.m~              isEven.m
Data                     GetKeyPress.m~           PixelsPerDegree.m        isOdd.m
DrawFixationPt.m         MooneyImgLR.mat
DrawFixationPt.m~        MooneyImgMF.mat
DrawImage.m              NITPParadigm.m
DrawImage.m~             NITPParadigm.m~
DrawText.m               PTB Cheat Sheet.d

EDU>> ls MoreStuff/
AudioVisualExpmt         FileIO                   StairCaseExperiment
FaceAdaptationExpmt      QuestExperiment

EDU>>
```

If coding (or interested) look into MoreStuff for additional examples of Experiments.

GUIDELINES

WHERE ARE MY INSTRUCTIONS?



```
EDU>>
EDU>>
EDU>> ls
MoreStuff       Step1           Step3           docs
README.pdf      Step2           archive

EDU>> ls Step1/
Calculations.m                  SimpleFunction.m                plotExample2.m
MatlabIntroIntensive.pdf        SimpleFunction2.m               plotExample3.m
MatlabProgrammingStyleGuide.pdf original_tutorial
README.doc                      plotExample.m

EDU>> ls Step2/
CORRECT.WAV                     TroubleshootingTiming.pdf       martini2.jpg
DarkScreen.m                    Usingcolormaps.m                original_tutorial
DarkScreen.m~                   Usingcolormaps2.m               scaleif.m
FunkyScreen.m                   dummy_data.txt                  testResponses.m
PracticeKeyPresses.m            getResponse.m
README.doc                      hid_probe.m

EDU>> ls Step3/
Activate_Screens.m      GetKeyPress.m           PixelsPerDegree.m
Activate_Screens.m~     GetKeyPress.m~          PixelsPerDegreeE.m               T.m
Data                    MooneyImgLR.mat         PlayAudio.m                      T.m~
DrawFixationPt.m        MooneyImgMF.mat         PlayAudio.m~            sandbox.m
DrawFixationPt.m~       NITPParadigm.m          README.pdf              why_CRmod.m
DrawImage.m             NITPParadigm.m~         RealTimeAnalysis_NITP.m why_CRmod.m~
DrawImage.m~            PTB Cheat Sheet.docx    WaitForTR.m
DrawText.m              PTB Cheat Sheet.pdf     WaitForTR.m~
DrawText.m~             PTB4NITP.m~             isEven.m

EDU>> ls MoreStuff/
AudioVisualExpmt        FileIO                  StairCaseExperiment
FaceAdaptationExpmt     QuestExperiment

fx EDU>>
```

THESE SLIDES

README docs in each directory.

GUIDELINES

Please - do not use your own old code and your own software to program your task.

1. It's annoying (we can't help you and more often then not there is some trivial but painful incompatibility with some listening device in the scanner, e.g., projecting your stimuli).

2. Our code works and is configured to work with the devices in the scanner.

3. Cameron spent hours (well 2) slaving over this code.

4. It makes it easier for us b/c there are more people who can help you.

It's no problem if you want add snippets of your own code, call your own functions, change how you save data and so on - the request is to simply work within the skeleton we provided you with (it'll accommodate pretty much every imaginable paradigm).

Happy Experimenting