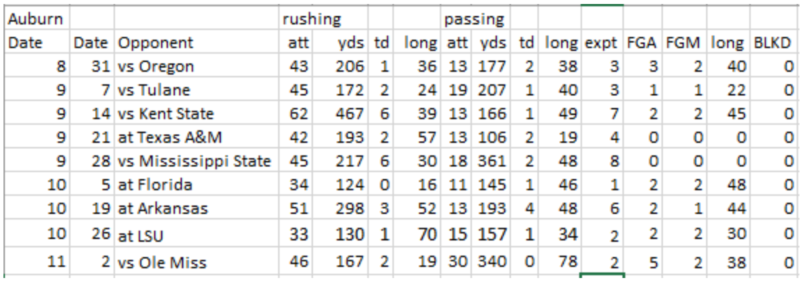
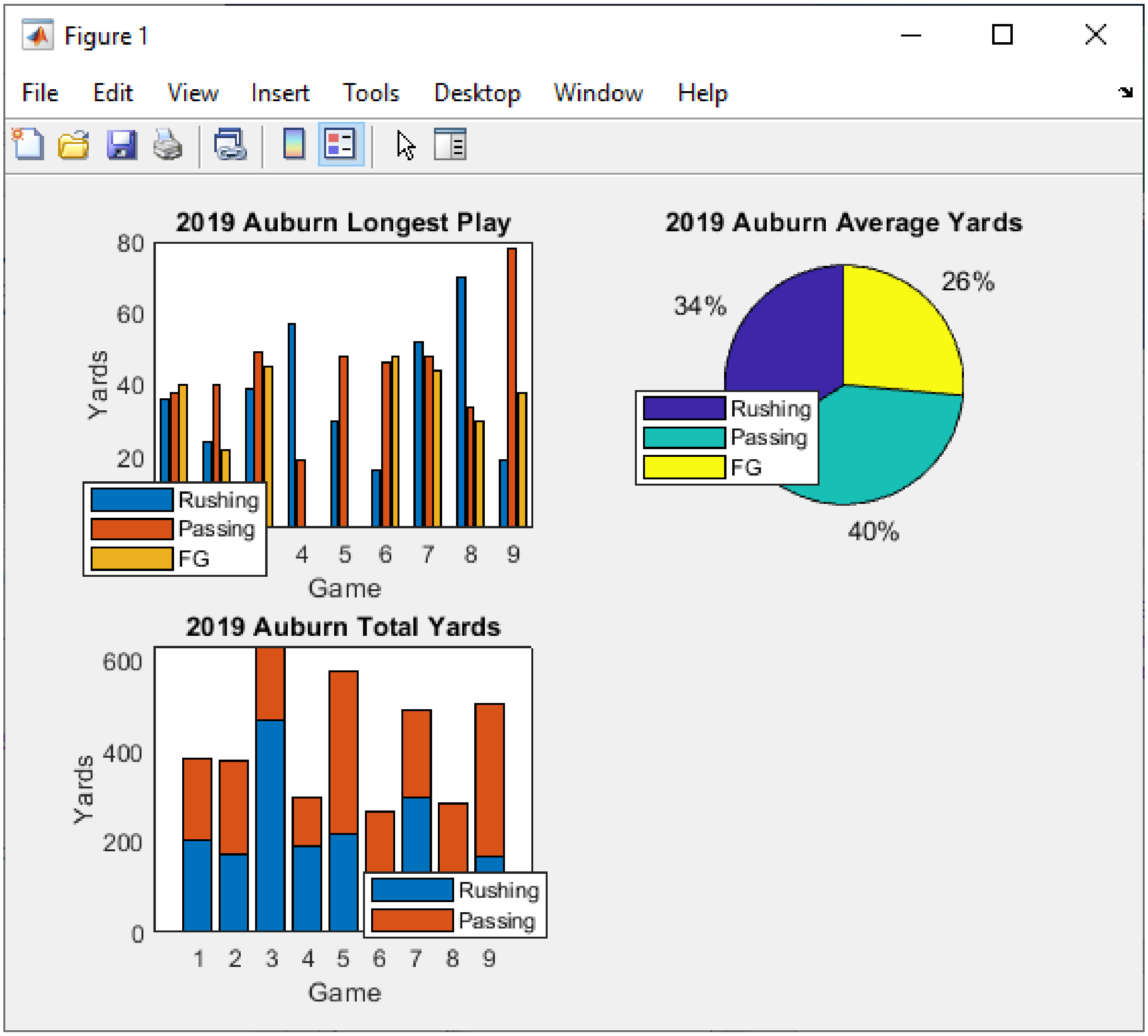
```
% J Hundley
% assign09.m
% % read Auburn season football scoring stats from spreadsheet
% create a report and graph

clc, clear all
format compact
%***** CONSTANTS *****
AU_STATS = 'AU_stats09.xls';

%***** INPUT *****
if ~exist( AU_STATS, 'file' )
    disp( 'file not available' )
else
    % file available continue with problem solution
    % read auburn statistics
    [ stats, labels ] = xlsread( AU_STATS );
    [ nGames, nCols ] = size( stats );

    %***** COMPUTE *****
    % compute auburn points scored for each game
    % no loop using colon notation for columns
    points = ( stats(:,6)+stats(:,10) )*6 + stats(:,12) + stats(:,14)*3;
    % combine TDs, extra pts, FGs and game points into a matrix
    printMatrix = [ stats(:,1:2),stats(:,6),stats(:,10),stats(:,12),stats(:,14),points ];

    %***** OUTPUT *****
    % print stat report and draw graph
    % print with title that includes last game date and column headings
    fprintf( '2019 Auburn Football as of %d/%d \n', stats(nGames,1), stats(nGames,2) )
    fprintf( '      Rush  Pass Extra Field Total\n' )
    fprintf( 'Date  TDs   TDs   Pts  Goals Points\n' )
    % print tabel of dates and stats using one fprintf and matrix
    fprintf( '%02d/%02d  %2d    %2d    %2d   %2d     %2d\n', printMatrix' )

    % draw graphs of stats
    % grouped bar chart of longest yards for rushing, passing, and FG
    subplot(2,2,1)
    bar( [ stats(:,7), stats(:,11), stats(:,15) ] )
    legend( 'Rushing', 'Passing', 'FG' )
    title( '2019 Auburn Longest Play' )
    xlabel( 'Game' )
    ylabel( 'Yards' )

    % pie chart of average play yards for rushing, passing, and FG
    subplot(2,2,2)
    pie( [ mean(stats(:,7)), mean(stats(:,11)), mean(stats(:,15)) ] )
    legend( 'Rushing', 'Passing', 'FG' )
    title( '2019 Auburn Average Yards' )

    % stacked bar chart of total rushing and passing yards
    subplot(2,2,3)
    bar( [ stats(:,5), stats(:,9) ], 'stacked' )
    legend( 'Rushing', 'Passing' )
    title( '2019 Auburn Total Yards' )
    xlabel( 'Game' )
    ylabel( 'Yards' )
end
```

Results used for grading. 11/02 game is added.

```
2019 Auburn Football as of 11/2
      Rush  Pass Extra Field Total
Date  TDs   TDs   Pts  Goals Points
08/31   1     2     3    2      27
09/07   2     1     3    1      24
09/14   6     1     7    2      55
09/21   2     2     4    0      28
09/28   6     2     8    0      56
10/05   0     1     1    2      13
10/19   3     4     6    1      51
10/26   1     1     2    2      20
11/02   2     0     2    2      20
```



Auburn			rushing				passing								
Date	Date	Opponent	att	yds	td	long	att	yds	td	long	expt	FGA	FGM	long	BLKD
8	31	vs Oregon	43	206	1	36	13	177	2	38	3	3	2	40	0
9	7	vs Tulane	45	172	2	24	19	207	1	40	3	1	1	22	0
9	14	vs Kent State	62	467	6	39	13	166	1	49	7	2	2	45	0
9	21	at Texas A&M	42	193	2	57	13	106	2	19	4	0	0	0	0
9	28	vs Mississippi State	45	217	6	30	18	361	2	48	8	0	0	0	0
10	5	at Florida	34	124	0	16	11	145	1	46	1	2	2	48	0
10	19	at Arkansas	51	298	3	52	13	193	4	48	6	2	1	44	0
10	26	at LSU	33	130	1	70	15	157	1	34	2	2	2	30	0
11	2	vs Ole Miss	46	167	2	19	30	340	0	78	2	5	2	38	0



Read all instructions before beginning your work.

COMP1200-MatLab - assign 09
Due 4:45 pm – Friday – November 15, 2019
Submit `assign09.m` via Canvas

NOTE:
Your submitted file(s) MUST be spelled and cased as instructed.
[-5 points for not doing so.]

Before you start writing your program:

Read the complete instructions.

Program**: assign09.m**

Scoring statistics are available for Auburn and the opponents for each football game. Read the statistics from files and print a report and graph of information as instructed.

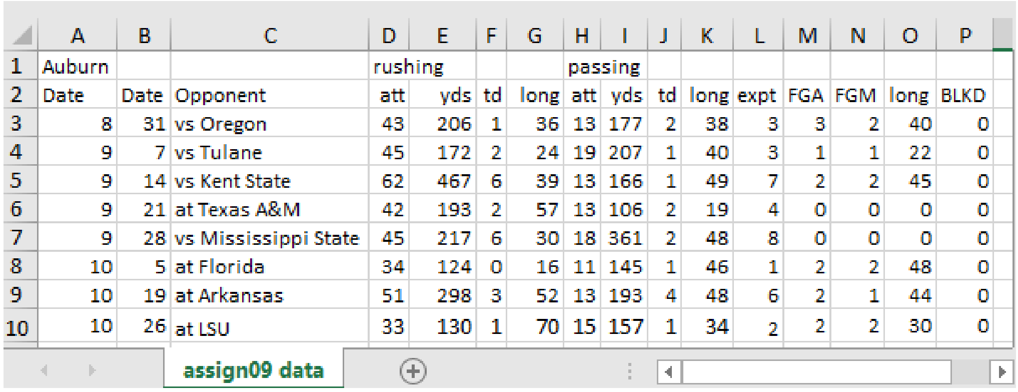
Problem Constants:

AU file name `'AU_stats09.xls'` **<< NOT .xlsx**

Problem Inputs:

See spreadsheet. **Note, you do NOT know the number of games.**

You may observe the matrix results of the `xlsread()` by temporarily removing the semicolon.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Auburn			rushing				passing								
2	Date	Date	Opponent	att	yds	td	long	att	yds	td	long	expt	FGA	FGM	long	BLKD
3	8	31	vs Oregon	43	206	1	36	13	177	2	38	3	3	2	40	0
4	9	7	vs Tulane	45	172	2	24	19	207	1	40	3	1	1	22	0
5	9	14	vs Kent State	62	467	6	39	13	166	1	49	7	2	2	45	0
6	9	21	at Texas A&M	42	193	2	57	13	106	2	19	4	0	0	0	0
7	9	28	vs Mississippi State	45	217	6	30	18	361	2	48	8	0	0	0	0
8	10	5	at Florida	34	124	0	16	11	145	1	46	1	2	2	48	0
9	10	19	at Arkansas	51	298	3	52	13	193	4	48	6	2	1	44	0
10	10	26	at LSU	33	130	1	70	15	157	1	34	2	2	2	30	0

Problem Outputs:

Report and graphs. See output.

Other variables:

One matrix with the six columns of input and the total points as needed

Equations:

Compute the total points for each game.

TD = 6 points. Extra point = 1 point. Field goal = 3 points.

Output:

Four graphs in one figure,

Group bar chart of longest yards for rushing, passing, and FG
Pie chart of average play yards for rushing, passing, and FG
Stacked bar chart of total rushing and passing yards
One additional chart of your choosing

Report:

Date of last game in spreadsheet
You do not know the number of games.

```
2019 Auburn Football as of 10/26
      Rush  Pass Extra Field Total
Date  TDs   TDs   Pts  Goals Points
08/31   1     2     3    2     27
09/07   2     1     3    1     24
09/14   6     1     7    2     55
09/21   2     2     4    0     28
09/28   6     2     8    0     56
10/05   0     1     1    2     13
10/19   3     4     6    1     51
10/26   1     1     2    2     20
```

Using one matrix in one `fprintf()`

Do not use commands and statements beyond what has been taught on class.

New commands:

Continue ONLY if file exist.

`xlsread()`

Use colon notation when using a column numbers.

NO LOOPS when computing or printing numbers.

Use data to determine number of games.

`subplot()`

`pie()`, grouped, stacked `bar()` charts with `title()`, `legend()`, `xlabel()`, `ylabel()` four graphs on one figure

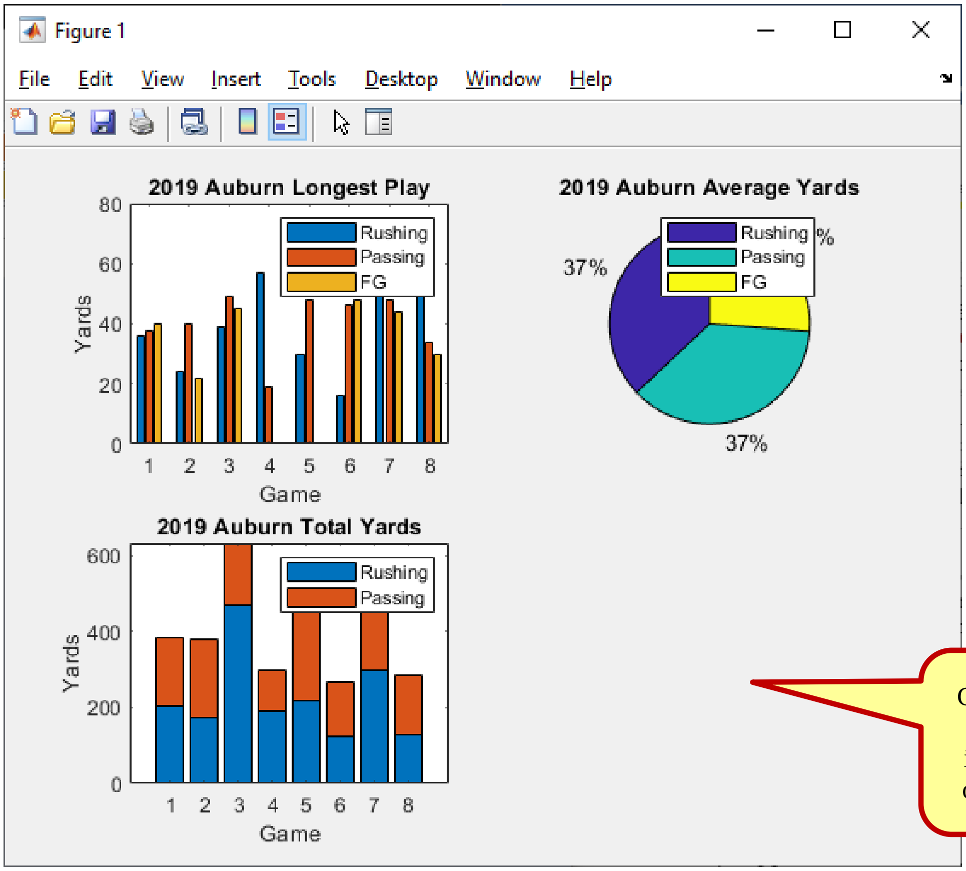
Continue to use:

Zero fill dates

Set decimals as shown with columns right aligned.

Use descriptive variables.

Remove statements not needed for current requirements



Create a new chart of your choice, i.e. do not reuse one of the original three.

ALL script files

Instructions for all assignment scripts:

- □ See Standards for Documentation of MATLAB Programs on the Canvas Resources page.
- □ Insert comments at the top and throughout each file.
  - o Include the follow comments at the beginning of this (and ALL) files.
    - % submitter's name, **GROUP # or "none"**
    - % other group members' names or "none"
    - % `program file name`, ex. assign02a.m
    - % due date of the assignment
    - % **statement about collaboration REQUIRED.**
    - % a short narrative about what the file does
  - o Use the algorithm given as comments throughout your program.
- □ Observe the instructor's rule for naming variables.
  - o Use ALL CAPS for constants variable names.
  - o Start other variables with lower case.
  - o Use descriptive variable names.
- □ Use Sample Input/Output as a guide.
- □ Code clarity:
  - o Indent blocks as needed. **Use Smart Indent.**
  - o Divide your solution program code into sections as noted in the algorithm. Use blank lines as needed to group statements.
  - o Use section comments as well as the algorithm step comments.
  - o Remove statements from previous assignments that do not apply to the current requirements.
- □ Use comments to show units.
- □ **Use the CONSTANT and variable names, not numbers. Exceptions are incrementers (or counters) and numbers without identity.**
- □ No extra output, i.e. use semicolons

GRADE OF ZERO **for a file if submitter name not part of Canvas group.**

(-3pts) No CURRENT GROUP# or "none".
(-3pts) For your own protection, type "none" for other group members if submitting alone.
(-5pts) Five point penalty for not joining your Canvas group.

(-5pts) Zero points for comments if no collaboration statement.

NOTE:
Your submitted file(s) MUST be spelled and cased as instructed.
[-5 points for not doing so.]

Submit via Canvas:

`assign09.m` MATLAB script file