UNM - ECE 131
Fall 2017
Final Project
Design Doc Due - **Nov. 30th** by 11:59 PM
Project Due - **Friday, Dec. 8th** by 11:59 PM

Note: Start early! - this project will be challenging and take some time and effort. Seek out help during office hours/by email if you need help. Also there is tutoring available.

Ethics Reminder:
- Everyone needs to submit their own work!
- In no circumstances is it O.K. to look at someone else's code, or look for code in Internet, or copy any portion of it. THIS IS CHEATING!
- Looking online for answers is also cheating. If you can find it, I promise you, we will find that solution too, and YOU WILL GET A ZERO.
- You may discuss algorithms at a high level of how you solved a problem, but you may not share code - including verbally.
- Ask the TAs or Professor if you have any questions or concerns with regards to the ethics policy, which is also clearly stated in the syllabus.

Overview:
You will implement a tic-tac-toe game using the *gfx* C Graphics Library developed at the University of Notre Dame. You will draw the game board, accept mouse clicks to determine which cell was selected, and draw either a square or circle depending on which player's turn it is. You will also get some practice writing to a file, and presenting your work to a TA.

1. Design Document
   You should read this statement in its entirety before you begin the project. After doing so, you should spend time to carefully plan how you will implement the game. Things you should include in your design document are: what functions you will need and how you will break up your code, clear definitions of what those functions will do, and who will call those functions. You should also include some pseudocode discussing the flow of your program. You are free to draw diagrams, write a paragraph, or use any other techniques as long as your submission is clear.

2. Download the *gfx* Graphics Library files from
   https://www3.nd.edu/~dthain/courses/cse20211/fall2013/gfx/

You should download the three files from the website, and follow instructions for compiling an example program. YOU SHOULD NOT MODIFY THESE FILES

Note that if you are using a Mac, you will need to use ‘ssh -Y name@linux.unm.edu’ instead of the regular ssh command. This will allow you to forward a graphics window from the server to your local display.

MobaXTerm has this functionality built-in on Windows, so you should be able to do everything without change.

If you are using a different terminal emulator, and run into issues getting X11 forwarding working, contact a TA or come to office hours.

3. Once you have run and compiled the program, you can rename the example.c to be <netid>_tic_tac_toe.c to use it as the basis of your own program.

4. Using a Makefile:
    a. Makefiles are a way to repeat a sequence of commands for building your project. Because the gcc command for compiling the project is fairly long, we suggest using a Makefile to make your life easier, although it is not a requirement.
    b. To create a Makefile do the following:
        i. $ vi Makefile   // in the same directory as your project (case sensitive, the M is capital)
        ii. Add the following lines to the Makefile

```
all:
        gcc example.c gfx.c -o example -lX11 -lm
```

        iii. You will need to change “example.c” to your new file name, and “example” to the desired output filename.
        iv. To use the makefile, now just type ‘make’ at the command-line. The invocation to make, should be done in the same directory where the Makefile is. This should execute the gcc command for you, so you don’t have to remember the entire gcc command every time.

5. The Project.
    a. First, you should become familiar with the different functions available within the *gfx* Library. We suggest playing around with them for a bit first before doing the project. Try drawing lines at different locations in different colors. Also try changing your window size.
    b. Next you will create functions to draw an X and an O. You are free to implement these as you like. One technique is to pass the center coordinates of the X or O as a parameter along with the height and width of the cell. From this information, you can calculate where to draw your X or O. You will later call these functions to draw either the X or O to the selected cell.

6. The main program:
    a. First, the main program should prompt a user to enter the dimension(s) for the tic-tac-toe board (you may choose to allow rectangular boards, but only allowing square boards is perfectly acceptable).
    b. The program should then ask who goes first. The user should enter either ‘x’ or ‘o’.
    c. The main program should then draw the board (this would be an appropriate place to create a separate drawBoard function)
    d. The game should then begin with the selected player’s turn.

7. The Game:
    a. The game should alternate between player 1 and 2.
    b. It should prompt the user to select a cell with the mouse.
    c. It should wait for a mouse click within one of the cells. If the click is not within the boundaries of the grid, or it is within an already occupied cell, you should prompt the same player to click again.
    d. You should then draw the appropriate symbol X or O at the selected square. You should use a different color for Xs and Os.
    e. The game should end when there are no blank cells left to select.
    f. It should then prompt the user at the terminal to play again. The user can select ‘y’ or ‘n’ accordingly.

8. File logging:
    a. Your program should log all of the moves to a file called “tic_tac_toe.out”
    b. On each turn, you should append a line to the file with the following information
    “Player ??? played a (X or O) at tile (x, y)”
    Where x,y are the coordinates for the cell as shown here:

0,0	0,1	0,2
1,0	1,1	1,2
2,0	2,1	2,2

9. Extra Credit!
    a. You can include code to check for a winner for extra credit points. This code should check at the end of every turn to see if there is a winner, and if so, end the game, and display a message.

10. Submission
    a. For submitting this assignment, as usual, you will submit your .c file to Learn. You should not submit either of the gfx library files. WE WILL USE THE DEFAULT FILES TO GRADE.
    b. In addition to the online submission, you are expected to schedule a time (more details to come later) with a TA where you can provide a short demo of your code. The TA’s will ask you some questions about your code and the project as a whole during this time.
    c. For online students, you may choose to either come to campus and meet with a TA in person, or if that is not an option, you may make a short video to demonstrate your project. You are then expected to answer questions about your project as well, via email. NOTE: this is not an option for the on-campus students except in extenuating circumstances, in which case contact the TAs.
    d. In order to receive partial credit, your code must at least compile.

Grading Rubric:

● Design Document:	10
● Compiles and opens a graphics window:	5
● Accepts user input for game board and draws board correctly:	10
● Prompts for who goes first:	5
● Accepts user mouse click:	10
● Correctly draws X or O in proper cell:	20
● Game plays properly, ends when all cells selected, play again:	20
● File logging is correct:	10
● Demo & Questions:	10

Total: 100