Lab 2. Introduction to C (Spring 2023)

All students do Lab 2 by themselves (no partner for Lab 2)

Preparation

Read all of ebook Chapter 2 or Sections 1.12, 2.1, 2.2, 2.3, 2.5, and 2.7 of the textbook
Find the starter project **Lab2_EE319K** (EE319K installer, or Canvas directions).

Purpose

The general purpose of this laboratory is to familiarize you with the software development in the C programming language. We choose a problem that exercises problem-solving skills you acquired in EE306 that allow you to devise a solution (algorithm). However, you will code the solution in C instead of assembly.

Requirements

The objective of this lab is to write three C functions in **Lab2.c** that are called by grader code in **Lab2Grader.o**. Your task is to write these three functions. Lab 10 involves a hand-held video game. These three functions will be used later in Lab 10.

The exact requirements for your lab will be revealed in the UART window when you enter your EID into the project and then build and run the system.

Consider two rectangles as shown in the following figure. Each rectangle is defined by the (**x,y**) position of the lower left corner, a height **h**, and a length **w**. The x position varies from 0 to 127, and the **y** position varies from 0 to 159. Location (0,0) is the upper left and (127,159) is the lower right.

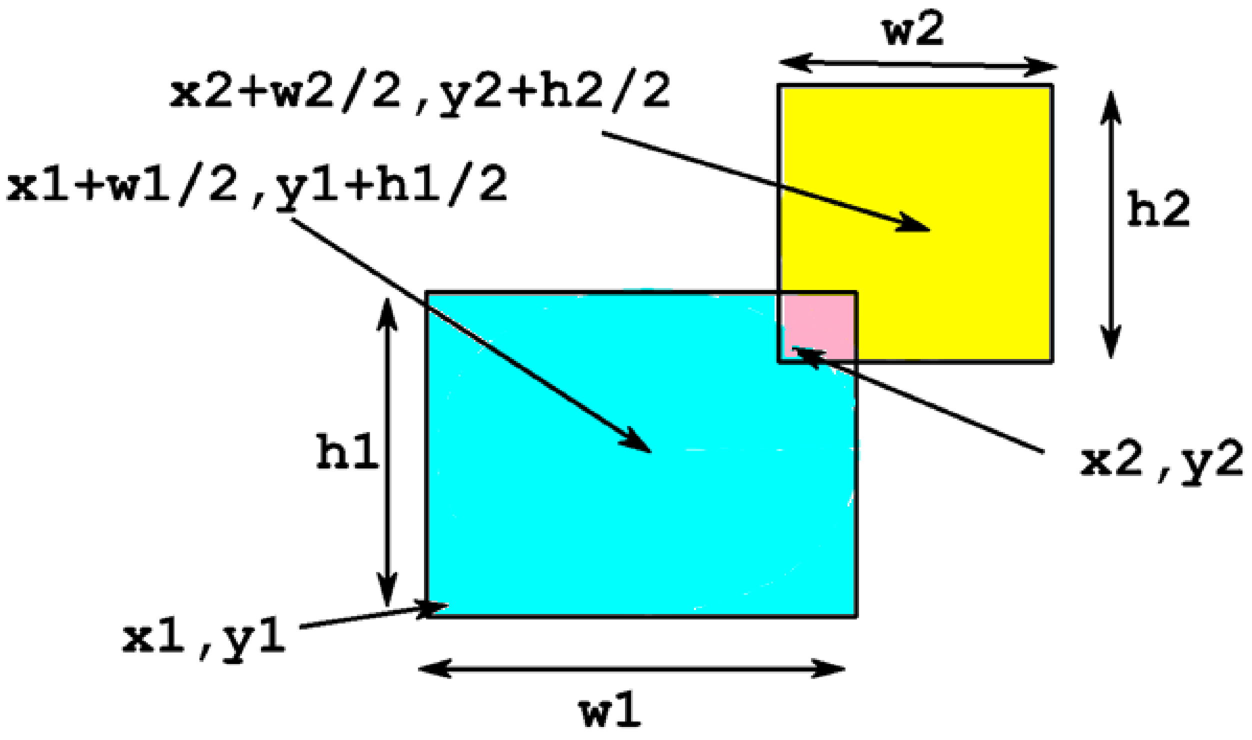


Figure 1. Two rectangles on the graphics screen, with overlap in pink

If the lower left corner of the rectangle is (**x1**,**y1**) then the upper right corner is at (**x1**+**w1**-1,**y1**-**h1**+1). In Figure 2, the yellow rectangle's lower left (**x1**,**y1**) is (4,9), **w1** is 10, and **h1** is 4. The upper right is (13,6).

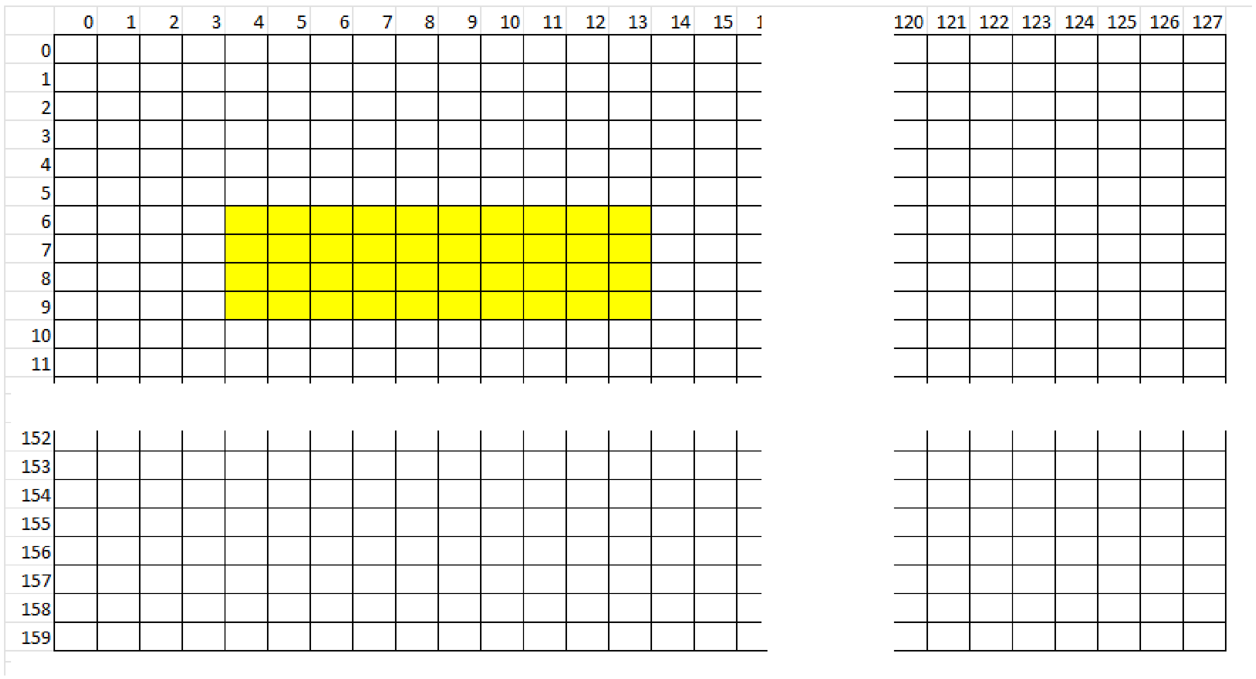


Figure 2. Each pixel on the LCD screen has an x position from 0 to 127, and a y position from 0 to 157.

The first function, called **Distance**, will calculate the distance from (**x1**,**y1**) to (**x2**,**y2**). See the starter code for details of how the function should operate. You will implement one of these three possible distance functions

Square distance = $(x1-x2)^2 + (y1-y2)^2$

Manhattan distance = $|x1-x2| + |y1-y2|$

ECE319K distance = $\max\{ |x1-x2| , |y1-y2| \}$

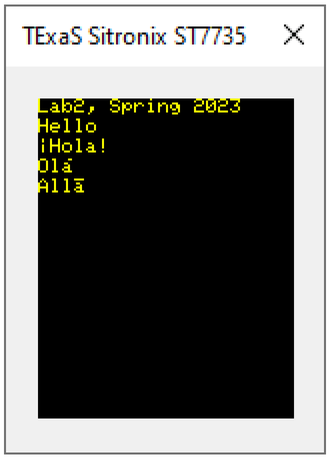
Note: early versions of the Lab2 autograder mistakenly expected

ECE319K distance = $\min\{ |x1-x2| , |y1-y2| \}$

So if your grader expects minimum, just accept my apologies, solve minimum and go on

The second function, called **OverLap**, is to determine if the two rectangles overlap. See the starter code for details of how the function should operate. You may assume both rectangles completely on the LCD screen.

The third function, called **SayHello**, will be used to output messages in different languages. Your **SayHello** function will call my **LCD_OutChar** function seven times to display seven characters on the LCD. You will see the output on the LCD, as shown in the next figure. You will learn the low-level details of the LCD later in Lab 7.



```
Lab2, Spring 2023
Hello
¡Hola!
Olá
Allã
```

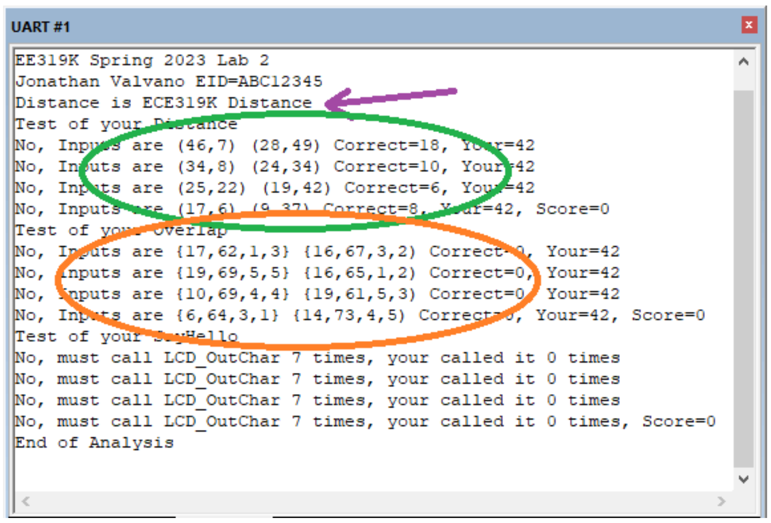
Figure 3. The grader calls your SayHello four times.

If you do not see the LCD window, execute **Peripherals->TExaS Sitronix** when in the debugger.

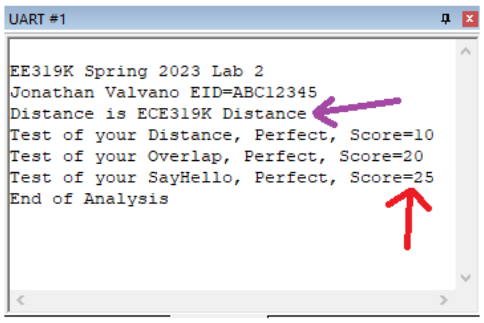
M1 Macintosh students using CCS: you can do Lab2 with or without an actual ST7735 LCD. The grader will observe your calls to my **LCD_OutChar** function to make sure the correct number of calls are made with the correct characters. The CCS Lab2 autograder also expects minimum for ECE319K distance, so if you get this version, solve for minimum instead of maximum.

Procedure

The starter project provided (Lab2_EE319K) has one assembly file Startup.s and your **Lab2.c** C file, and multiple object files (Lab2grader.o, pll.o, ST7735.o, and uart.o). All your tasks are performed by writing code for the three subroutines (called functions in C) whose blank stubs are provided in Lab2.c. To test whether your implementations of these functions are correct, you can run the project in the simulator. The input parameters and expected output will be shown in the UART window as the grader runs. Figure 4 shows a typical result in the UART window (you will need to enter your name and EID in Lab2.c). The maximum score is 25. Purple arrow shows which Distance formula to implement. Green oval shows some input/output parameters for Distance. Distance will have 30 randomly generated test cases Orange oval shows some input/output parameters for Overlap. Overlap will have 40 randomly generated test cases.



```
EE319K Spring 2023 Lab 2
Jonathan Valvano EID=ABC12345
Distance is ECE319K Distance
Test of your Distance
No, Inputs are (46,7) (28,49) Correct=18, Your=42
No, Inputs are (34,8) (24,34) Correct=10, Your=42
No, Inputs are (25,22) (19,42) Correct=6, Your=42
No, Inputs are (17,6) (9,37) Correct=8, Your=42, Score=0
Test of your Overlap
No, Inputs are {17,62,1,3} {16,67,3,2) Correct=0, Your=42
No, Inputs are {19,69,5,5} {16,65,1,2) Correct=0, Your=42
No, Inputs are {10,69,4,4} {19,61,5,3) Correct=0, Your=42
No, Inputs are {6,64,3,1} {14,73,4,5) Correct=0, Your=42, Score=0
Test of your SayHello
No, must call LCD_OutChar 7 times, your called it 0 times
No, must call LCD_OutChar 7 times, your called it 0 times
No, must call LCD_OutChar 7 times, your called it 0 times
No, must call LCD_OutChar 7 times, your called it 0 times, Score=0
End of Analysis
```



```
EE319K Spring 2023 Lab 2
Jonathan Valvano EID=ABC12345
Distance is ECE319K Distance
Test of your Distance, Perfect, Score=10
Test of your Overlap, Perfect, Score=20
Test of your SayHello, Perfect, Score=25
End of Analysis
```

Figure 4. UART window showing full score of 25. It will show Spring 2023.

If you do not see the UART window, execute **View->SerialWindows->UART1** when in the debugger.

Demonstration

There are grading sheets for every lab so you know exactly how you will be evaluated. During the demonstration, you will be asked to run your program to verify proper operation. You should be able to single step your program and explain what your program is doing and why. You need to know how to set and clear breakpoints, watch variables like the Readings array and any local variables you declare in your subroutines.
Please make note of which TA checked you out. The name of the TA will greatly help you when resolving any grading issues later.

Do all these well in advance of your checkout

1. **Signup for a time with a TA. All students do Lab 2 by themselves**
2. **Upload your software to canvas, make sure your name is on your software**
3. **Upload your one pdf with deliverables to Canvas**

Do all these during TA checkout meeting

1. **Have your one pdf with deliverables open on your computer**
2. **Have Keil Lab 2 open so TA can ask about your code**
3. **Start promptly, because we are on a schedule.**
4. **Demonstrate lab to TA**
5. **Answer questions from TA to determine your understanding**
6. **TA tells you your score (later the TA will upload scores to Canvas)**

Deliverables

Upload your Lab2.c file to Canvas. Combine the following components into one pdf file and upload this file also to Canvas. Have the pdf file and Keil open on the computer during demonstration

0. Your name, professor, and EID.

1. A screenshot of your UART1 window showing the final output.

Optional Feedback : http://goo.gl/forms/rBsP9NTxSy