CSE 102 – Assignment 4

Introduction

This assignment builds on the last exercise in Studio 4, in which you are displaying an image on the 5 by 7 pixel display. In this assignment, you will use the display to show the printable characters in the ASCII character set.

Serial Input

We start by exploring the ability to input characters via the serial interface. This is the same mechanism we have used earlier to print to the PC screen. Now, we will input characters from the PC keyboard. Study the use of the Serial.available() and Serial.read() functions to enable the typing of keys on the PC and the subsequent delivery of those keystrokes to the Aruduino executing code. You can limit the keys supported to those in the printable ASCII character set (approximately codes 0x20 to 0x7f).

Include Files

It is common in software development to have "shared" information (e.g., declarations that might legitimately be used in more than one place) placed in a separate file from the bulk of the source code. These files are frequently called "header" files, and in the C world have a .h extension.

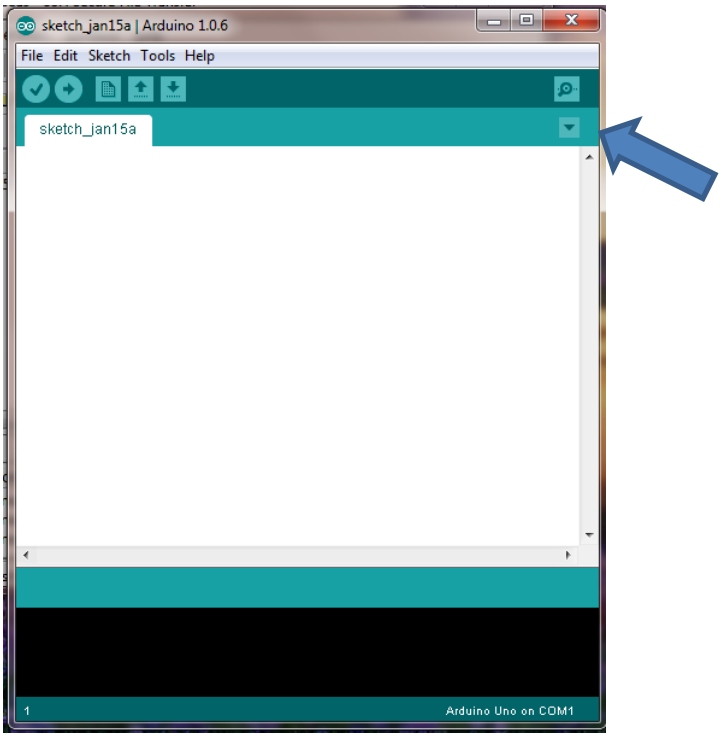
We will be using such a file to define the bit patterns that correspond to the characters we wish to display.

In your main program (the original Tab), you can reference the file `font.h` with a `#include` statement at the top of your program:

```
//my program
//
#include "font.h"
//remainder of my program
```

The above code includes the contents of `font.h` in the program (i.e., it is the same as if the text of the file were "included" right at the place of the `#include` statement.

Now we need to actually create the file. In the Arduino development environment, click on the arrow on the right side (just above the scroll bar) and create a new Tab with the name `font.h`. Populate the file with the contents of font.h.



Now take a look at what is in the file `font.h`. It defines an array called `font_5x7` that defines which LEDs should be on or off to display the 96 printable ASCII characters on a 5 by 7 pixel display. (Note: the weird `__ATTR_PROGRMEM__` that is part of the declaration is an Arduino thing that tells the compiler that we will never write to the array and therefore it can be positioned in read-only memory.)

The first index of the array is the character (indexed by its ASCII code – 0x20). The second index is the column of the 5 by 7 display. The bits of the array entry encode the individual LEDs for each row. You can explore the meaning by printing the entries in binary format (as you did for the temperature value in Studio 3).

```
c = 0x03; // '#'
Serial.println(font_5x7[c][0], BIN);
Serial.println(font_5x7[c][1], BIN);
Serial.println(font_5x7[c][2], BIN);
Serial.println(font_5x7[c][3], BIN);
Serial.println(font_5x7[c][4], BIN);
```

Hint: look at it sideways. Try the above with different values of `c`, especially values for which the character is not symmetric left and right. Check the writeup toward the end of the assignment and decide in what order you wish to complete the work (there are pros and cons for either path, and you are more than welcome to switch back and forth if you find that helpful).

Column Multixplexed Display

Using the column-based multiplexed control from lecture, write a program that accepts character input from the PC, and displays the input characters on the 5 by 7 pixel display, using the provided font file. You can initially display any image you wish. When the user presses a printable ASCII character on the PC's keyboard, the display should show that character, and continue to show that character until some other character is pressed on the keyboard.

A few things to watch out for and/or consider:

1. The polarity of the row outputs is not the same as the encoding in the font file. The font file as a 1 for the LED to be on, but the display wants a LOW on the row pin.
2. It is possible to extract the individual bit information from the font file in a single line of code (think about how you solved the problems from the last assignment). If statements are not needed. (You will not be penalized if you revert to a solution that does use if statements. I'm just nudging you to try it without to begin with.)

NOTE: the font file is missing some data!

As you will notice as soon as you look for it, the actual font information for the ASCII characters '0' through '9' is not present in the file that we provided (the entries are there, but they are all blank). As part of the assignment, please design an appropriate image for each of these characters and include it in the font file. Rest assured we will check these out when you demo!

While this writeup saves the design of the characters '0' to '9' for last, you actually might find it useful to do this earlier to help in understanding how the hex values in the character array initialization correspond to LEDs on or off on the display.

Submitting Your Work

When you have finished your assignment and demonstrated it to the instructor or TA, make sure they record your completion.