CSCI241 Guided Lab Exercise
(Keyboard Cursor Control)

Motivation:

You're given a skeleton 6502 assembly language program which displays an "*" in the center of the screen and then waits for a key to be struck. You're to add code so that by striking special keys, the "*" can be moved left, right, up or down.

Program Specifics:

1. Copy/paste the contents of **the K:\CSCI241\001-43644\Share\Lecture – Week #12 (I-O, Interrupts, DMA) (Continued)** folder into your private **K:\CSCI241\001-43644\Students\xxxxxx** folder.

2. Log into your www.replit.com account and launch your **ASM65** project. Also launch your **M6502** project in another browser window.

3. Click on the REPL icon that looks like three vertical dots at the top right of the **Files** panel and choose **Upload folder** option. Navigate into your private **K:** drive folder and into the **ASM65\TLB** folder, then click once on the **CURCTL** folder to select it, then click the **Upload** button.

4. Now drag and drop the **ASM65** project's **CURCTL** folder on the **TLB** folder so **CURCTL** is now a subfolder of **TLB**. There should be two files in the **CURCTL** folder: **BIODEF.P65** and **CURCTL.P65**.

5. You are to now add code to **CURCTL.P65** to check if the key struck is one of these special movement keys:

```
 i
j k
 m
```

   and, if so, adjust the **ROW** and **COL** position values appropriately so the "*" will next be displayed one position to the right, left, up or down. Notice there are already symbolic definitions for these ASCII keycode values, so use them. If the key struck is not a special movement key, then don't change the position of the "*", just loop back and wait for another key to be struck.

   [Make sure the **ROW** and **COL** values are limited to the range (0,1,2,…,15) and (0,1,2,…,63) respectively. For example, if you repeatedly strike the "k" key, the "*" should move one position to the right on each key strike and when it's at the right end of the row, striking "k" again should "wrap around" and restart at position 0 (i.e., the leftmost) position on the row.]

That's basically it. I strongly suggest using the incremental construction technique:

1. Implement a small portion of functionality, assemble the code, run the code and verify it works. Don't proceed until you've resolved any problems.
2. Decide what the next small portion of functionality should be.
3. Go to Step #1.

If you don't finish the program in class, I want you to complete it on your own.