CS2261 Lab 03: Arrays and Swap

Provided Files

main.c
myLib.c
myLib.h
coords.c
coords.h

Files to Edit/Add

main.c
Your Makefile
Your task.json

Instructions

In this lab, you will be completing several different TODOs, which will, piece by piece, complete a simple cactus jumping game (although the term “game” only loosely applies to this one). Your code may not compile until you complete an entire TODO block, at which point the game should compile with a new component of the final outcome (unless otherwise specified).

TODO 1 – Cacti

First, we want to create the cacti. They will sit on the top of the sand, and will move to the right (the movement has already been written).

Find the cactusCols and oldCactusCols arrays in main.c. These will be the locations of the cacti. Notice that we don’t have variables for the lengths of these.

- TODO 1.0: Create a global variable for the length of the aforementioned arrays, cactusCols and oldCactusCols. Since cactusCols and oldCactusCols contain the same number of elements, you should only need to create a single global variable for the array length.
- TODO 1.1: Find the drawCacti function. Complete this by calling the already-written eraseCactus function with each column in your oldCactusCols

array (to erase them) and the drawCactus function for each column in your cactusCols array (to draw them).
    - Hint: You'll need to use the global variable you wrote in TODO 1.0 for your iterations in this method!
- TODO 1.2: Call the drawCacti function in the main while loop to draw your cacti each frame of the game.

Compile and run. If you did everything correctly, you should see two equally-spaced cacti moving to the right, and, when they hit the right side of the screen, reset to the left.

TODO 2 – Player
The game wouldn't be much fun without a player, so let's draw one.

Open coords.c and examine the contents. For this lab (and this lab only), we have opted to draw the player by having arrays of coordinates (row arrays that match up to column arrays), and setting a pixel at each of those row-column pairs. Note that there is a macro defining the length of the coordinate arrays, which we may need later.

- TODO 2.0: In main.c, find the drawPerson function. It takes an array of rows and a same-length array of columns, along with the length of those arrays (remember: there would otherwise be no way to determine this), as well as a color. Complete this function so that it sets a pixel at each corresponding row-column pair in the arrays to the specified color.
    - Hint: don't overthink this! All you're doing is iterating over the length of the arrays and calling setPixel for each element i.
- TODO 2.1: In the main while loop, after waitForVBlank, call drawPerson with the invisiblePerson arrays using the SKYCOLOR to erase the person's previous location (this will make sense later).
    - Hint: look in coords.h for the macro to use as the numCoords parameter!
- TODO 2.2: A few lines later, call drawPerson with the visiblePerson arrays using WHITE to draw the person's current location.
    - Hint: look in coords.h for the macro to use as the numCoords parameter!

Compile and run. You should see our new player character on top of the sand, getting repeatedly smacked by cacti.

TODO 3 – Swap
For the purposes of this lab, we're going to have our player jump by swapping the coordinates of the visiblePerson with those of the invisiblePerson (who is currently safely above the cacti).

- TODO 3.0: In main.c, write a new function called swap that returns void and swaps the value of two integers passed in as parameters. Don't forget to write a prototype at the top of main.c.
    - Hint: think about what you will need to actually take in as arguments to make this work

Note: this function may NOT exploit the fact that the things we will be swapping (in later TODOs) are statically allocated and/or in arrays. It must work for any two integers.

- TODO 3.1: In our main while loop, if the Start button is pressed and the player is not already in the air, utilize your new swap function so that each element of invisiblePersonRows is swapped with each corresponding element of visiblePersonRows, and then the same for invisiblePersonCols and visiblePersonCols. This will make the player look like they've jumped in the air.
    - Hint: You'll need to call swap two different times (with different parameters) within the same for loop.
- TODO 3.2: In our main while loop, if some time has passed and the player is in the air jumping, utilize the swap function so that each element of invisiblePersonRows is swapped with each corresponding element of visiblePersonRows, and then the same for invisiblePersonCols and visiblePersonCols. This will make the player look like they've fallen back to the ground.
    - Hint: 3.2 should use the exact same code as 3.1!

Compile and run. If everything is working right, every time you press the Start button, given the player is not already in the air, the player jumps up to a place of safety above the cacti and eventually falls back down to where they started, standing on the sand. If this is the behavior of your lab, zip it up and follow the submission instructions.

Submission Instructions

Zip up your entire project folder, including all source files, the Makefile, and everything produced during compilation (including the .gba file). Submit this zip on Canvas. Name your submission Lab03_FirstameLastname, for example: "Lab03_TraskUlgo.zip".