Homework 2

loops, arrays, console output, random numbers, display path through 2D space

January 10, 2020

Abstract

The objective is to give you practice with two dimensional arrays, and with linked lists.

The context of this problem is that a two dimensional metric space exists, not centered on coordinates (0,0), but rather, offset, such that the leftmost bottom corner is at 0,0, and all other locations have at least one positive integer. Your code will navigate an object within this space, and show the trail it followed.

The problem to be solved:
A display screen is to be modeled as having 20 x 20 "fat" pixels (These are regions of the screen consisting of more than one pixel on each side.). You should model this screen as an array with 20 x 20 locations. Initialize the content of your array to all zeros. Use a random number generator, and modular arithmetic, to choose a location on this screen, in which to start. Seed the random number generator such that the list of random numbers is not the same every time you run your code. We think of a marker being at that point. Fill that array location with an (initially 1) index. Next, make your marker wander randomly. You might wish to create this appearance by making a new marker each time. Use the next index each time. Each time the marker "moves", print the display, with the index number showing. Note that a marker may reuse a location. The display should show the most recent index.

As your marker wanders through the array, you should also record every marker location in a linked list.

After the wandering is over, traverse the linked list, printing the index and location of the path travelled.

- Construct a sequence diagram for your project. You can draw it by hand and include a phone/photo, or draw it with a software tool.
- Use the test-driven development style for developing your code. Document this for each production function by:
    1. Write your function prototype, test function and stub implementation.
    2. Run your test function (which should invoke the stub and notice that the stub is insufficient as an implementation).

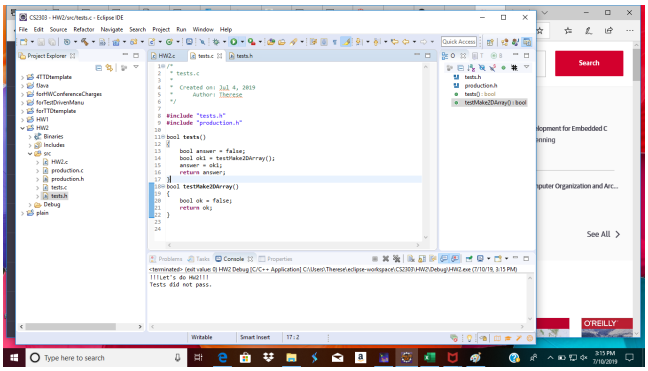

Figure 1: In the process of creating a new test.

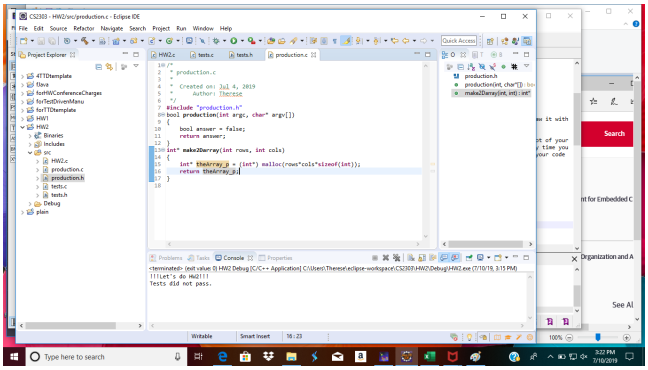

Figure 2: Starting some production code.

3. Take a screen shot showing the stub function and the report of its failure (The "before" screenshot).
4. Modify your stub to do its job.
5. Run your test function (If the implementation is right, this should be observed.).
6. Take a screen shot showing the replacement of the stub and its test results. (The "after" screenshot).

Be sure to run your code every time you add a few lines of test or production code. (See Figures 2 and 3.) Do not allow the number of warnings and errors to get large.

- Imagine a function for placing a marker in the array. Choose a known (not a random number) location. Build a test that checks for this.
- Imagine "moving"the marker (adding successive markers). To do this, it will probably help a lot to have a function that will print the array, for

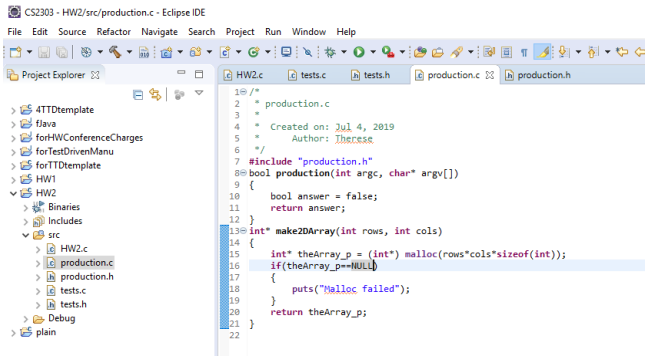

Figure 3: Adding more production code.

example, '|' between horizontally adjacent cells, and '_' between vertically adjacent cells. Don't worry about underlining the marked location. By printing the array between moves of the marker, you should be able to see the result of your code running. Build a test for this display function. You can populate the array with known data, print it, and check it (by eye if you wish).

- Use a loop to "move" the marker multiple times. Ihe index value of the marker should increase; show it as modulo 10, in the array printout, so the field is always 1 character wide. Build a test function that checks the length of your marker path.
- There are several linked list functions provided in starter code. You should read this code for understanding. Test that you can create an empty linked list. Test that you can create a linked list of length 1. Test that you can add elements onto the end of this list.
- Print, from the linked list the path the marker takes through the space. Test that the list you produce matches the list your code prints. There should be several test cases. Length zero, length 1, length greater than 1.

Things to do:

1. Either:
   (a) Make a C project from the Hello,World project.
   (b) Populate that project with tests.c, tests.h, production.c and production .h.

   or use the starter code.
2. Create the sequence diagram and include the electronic file (diagram, screenshot or photo). Make sure your name appears within the sequence diagram.

3. Place function prototypes for all of your functions from the sequence diagram into .h files.

4. As you work on the assignment, collect a sequence of befroe-and-after screen shots showing how your production code is growing.

5. Be sure to build and run often; do not allow warnings and errors to build up.

6. Show the sequence of moves by listing them, and also show the final path through the 2D space.

Grading

Criteria	Possible Poin
Project that looks like starter code	25
Sequence diagram that reflects the problem statement	25
Documentation of code development (those screenshots) that clearly follows test-driven style	25
Screenshot of path, and list of moves, that correspond	25
Total	100