Final

February 25, 2022

Abstract

This final is to be accomplished by your individual work. Due to the high improbability of identical solutions, students turning in identical solutions will be suspected of violating academic integrity.

This exam has two parts: One is like a homework, worth 200 points, the other is questioning your understanding of important concepts treated by the course.

1 Programming Problem

The context of this problem is testing the exploration behavior of a robot vaccum cleaner. Your code must produce the room finding behavior of the vacuum. The inspiration for this problem can be read at `https://www.bbc.com/news/uk-england-cambridgeshire-60084347`. Recall that exercising (but not to exhaustion) and feeling good is apt to assist your problem solving capacity `https://www.mdpi.com/2076-3425/11/5/546`.

1.1 The robot vacuum has requirements.

Our robot vacuum is expected to search through the single floor of the hotel in which it is working, and apply its vacuuming skill to each room, once per day. The implication is that a vacuum keeps track of which rooms it has vacuumed that day, and does not re-vacuum a room it has already visited that day.

The robot is required to remain within the hotel. Include in your code a test that the robot does not leave the building.

The vacuum is presented with an adjacency list representation of the floor of the hotel. You may certainly use code to convert this into an adjacency matrix representation if you wish.

1.2 Advice

Consider the diagrams we have discussed in this course. Which diagrams will you use to guide your thinking? It is probably a good idea to consider each

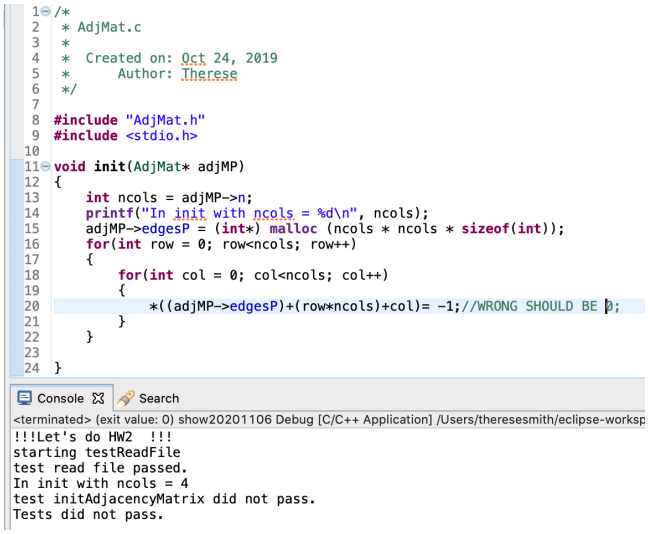

Figure 1: This is called the "before" screen shot.

diagram type and explain why or why not use that diagram with this problem. If the diagram seems applicable, express the problem or applicable part of the problem with it.

Be sure to construct test functions for each non-system function your production code uses. Put thought into the test cases. The goal is to know that a function works, when it passes all of its test cases. One test case per test function is usually not enough.

- Construct a sequence diagram for your project. Recall that sequence diagrams do not include testing. You can draw the sequence diagram by hand and include a phone/photo, or draw it with a software tool.
- Construct any other diagrams that the course has taught you might be useful in solving this kind of problem. For example, can you draw a state-transition diagram for this problem?
- Use the test-driven development style for developing your code. Be sure to include a test addressing the requirement that the vacuum will not leave the hotel. Each test function should have console output saying whether its test cases were failed or passed. Document this on a function by function basis. First the production function exists as a stub. Your test code will invoke this function, usually with multiple test cases, and a good test will be able to notice that the stub function is inadequate to pass all of the test cases. Take a screen shot of your production code function being tested in its manifestation as a stub, including the console showing that the test function fails. (See Figure 1.).

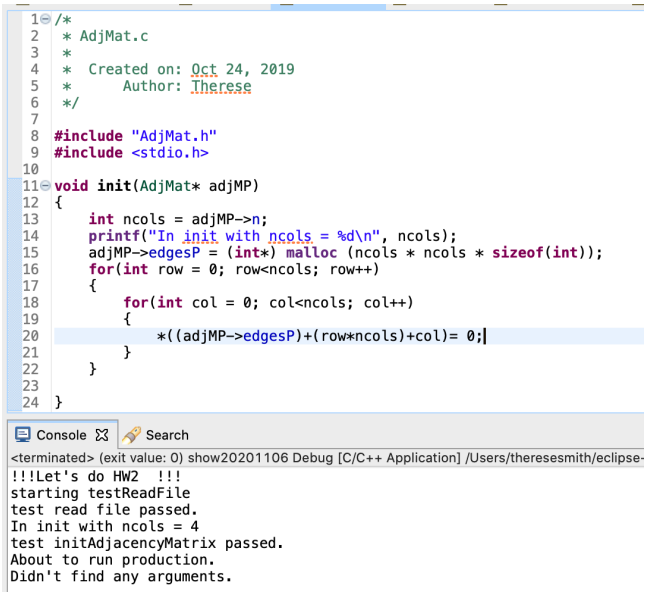

Figure 2: It is called the "after" screenshot.

This is called the "before" screen shot. After your production function has been developed sufficiently to pass all of the test cases, take another screenshot. (See Figure 2.) It is called the "after" screenshot. It should include the developed production code being tested and the console message about passing the test. Depending upon how many functions are in your sequence diagram, there could be a large number of screen shots.

Be sure to run your code every time you add a few lines of test or production code. Do not allow the number of errors to get large.

- Read from the command line a string that is the filename of the input file describing the adjacency list representation of the hotel floor, namely, its rooms.
- Read from the command line a string that is the filename of the output file recording the rooms vacuumed, in the order they were vacuumed. There is a distinction between discovering that a room is a neighboring room, and conducting the vacuuming of that room.

Things to do:

1. Make a C++ project from the Hello,World project.
2. Populate that project with tests.cpp, tests.h (or .hpp), production.cpp and production .h (or production.hpp).

3. You may use (e.g., import, or consult) code from any of the C++ code samples posted on our Canvas site.
4. Create the sequence diagram and include the electronic file (diagram, screenshot or photo). Make sure your name appears within the sequence diagram.
5. For each entity in your sequence diagram, create a class in your project.
6. Make sure every .cpp file includes its corresponding .h (or hpp) file.
7. Place function prototypes for all of your functions from the sequence diagram into appropriate .h (or .hpp) files.
8. For every function prototype from the sequence diagram/production code, invoke a test function in tests().
9. For every test function, also create a prototype (in tests.h or .hpp) and function definition (in tests.cpp).
10. Be sure to include comments identifying the three sections of a test function we have used in class.
11. Recall that each test function must invoke the function it is testing, and must report failure or success in the console.
12. As you work on the assignment, collect a sequence of screen shots showing how you have tested your production code. As always, these are pictures of production (not test) code and console output saying whether this function failed or passed its test.
13. Be sure to build and run often; do not allow errors to build up.

Grading

Criteria	Possible Points
Project that looks like starter code (such as for HW6)	20
Sequence diagram and other diagrams that reflects the problem statement	40
Documentation of code development (those development screenshots) that clearly follows test-driven style (including the modified time attribute on the screenshot files showing the sequence of development)	20
Input file that shows the adjacency list representation of the hotel floor.	20
Output file that records the order in which rooms are vacuumed.	40
Diagram of the hotel floor, as specified in the input file, as a graph, with notation showing the order in which the rooms are vacuumed.	40
Correct treatment of command line arguments	20
Total	200

Task / Level	full score	half score	quarter score
Use starter code	Do not change the test/production paradigm	some change	omit tests
Provide sequence diagram	identify instances of classes, communications that solve the problem	instances of classes or communications	neither
Provide before-and-after screenshots	show production code before and after with test results failing and passing respectively	showing test rather than production	not showing both of before and after
Screenshot results in console	correct answer	partially correct answer	no answer
Command line arguments	obtain and use	obtain	less

2 Concepts

1. Why use patterns? What mental process does the developer go through, and what benefit does this mental process confer, on the implementation code that is subsequently written?

2. Why use sequence diagrams? What mental process does the developer go through, and what benefit does this mental process confer, on the implementation code that is subsequently written?

3. Why prepare tests before writing implementation code (i.e., while the implementation is a stub)? What mental process does the developer go through, when designing a test? What benefit does this process confer on the developer, and on the implementation code that is subsequently written? Compare using test-driven development to "develop a lot then use the debugger", using figures of merit including development time, stress on mental health and value produced during the time when testing is occurring.

4. Why use pointers? For example, if you are contemplating using array indexing for a two dimensional array, and passing that array to another function (the called function), and hoping to use array indexing in the called function, what did this course advise instead, and what advantages does that advice have?

5. Explain what is bad about global variables. Explain what is bad about using break (except in switch/case constructs), continue, goto.