Deadlock Detection Program

I. Problem Statement

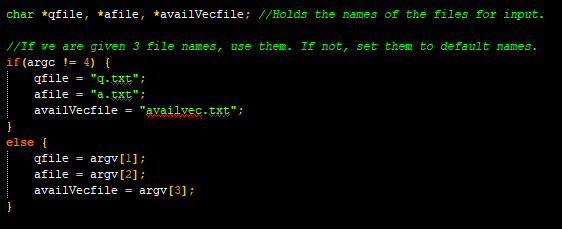
The assignment was to write a deadlock detection program. We were to read from files giving us a Q matrix, an A matrix, and an Available vector and assign them to arrays. Then, using deadlock detection algorithms, find if the current values would result in a deadlock or not.

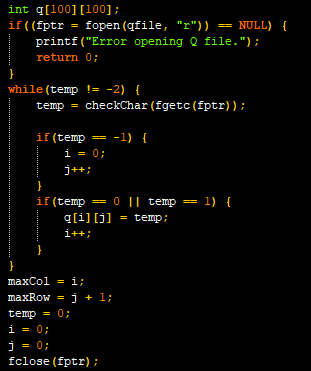
II. Initial Approach

Every part of the general design for the algorithm came from section 6.4 of the textbook. I knew I would need a way to read from files first, though. Once I had that working as intended, I could then use the algorithm to discover if a deadlock was detected or not. If I had time, I would then make it so that the user could pass file names through command line arguments.

I decided early that I would not have the user give a Resource vector as that can be found by adding up the Available vector and the columns on the A matrix. The Resource vector is printed out to the user so that they can make sure it is what they’re looking to test.

III. Process/Problems

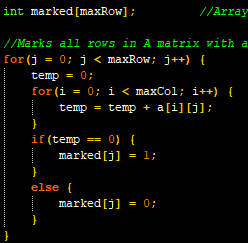


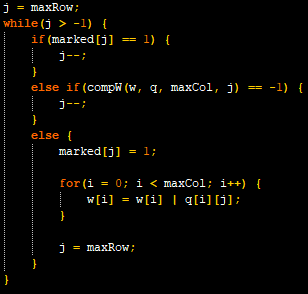


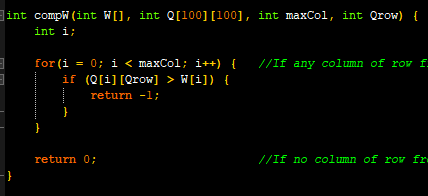
**Read from files:** I had never set up a file reader in C before so I had to do some research on this. The idea of the pointer was not complicated. I wish I had found a way to control the size of the matrix better. Instead, I initialized the matrix for Q as 100x100 and then kept up with how much space was needed. I set these values to maxCol and maxRow and used them for the size of matrix A and the vectors. This was also useful for any ‘for loops’ that I used.

The checkChar function I wrote returns an integer of 0 for char 0, integer of 1 for char 1, integer of -1 for NEW LINE, integer of -2 for end of file, and an integer of -3 for anything else such as whitespace.

I discovered an interesting problem that came up, although a little too close to deadline to fix. If you write the files in Windows and then pass them to Linux the program works properly. However, if you write them in Linux (or Linux run on NoMachine like I did) then the program won’t work. This is probably a NEW LINE problem that could have been dealt with if I had started sooner and had the time to deal with.

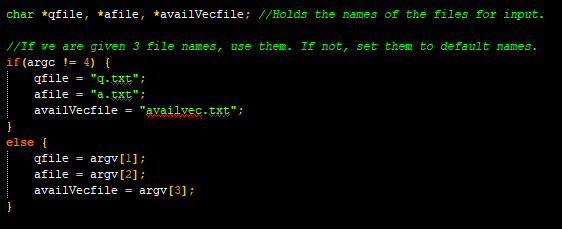






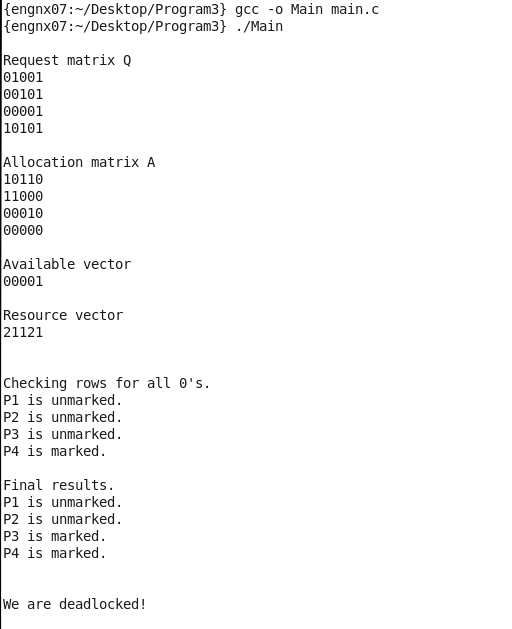
**Deadlock detection algorithm:** The first screenshot shows the initial pass through the A matrix for any rows with all 0’s to mark. The second screenshot shows how we pass through Q matrix and comparing it to the W vector (initialized to match the Available vector). If the row is unmarked, then it would check if that row lesser than or equal to the W vector then it is marked and added with bitwise OR to the W vector.

I checked that by writing a function, compW(), that would compare every part of the row. If any value of the Q matrix row was greater than the same part of W vector it would return -1. Otherwise, it would return 0.

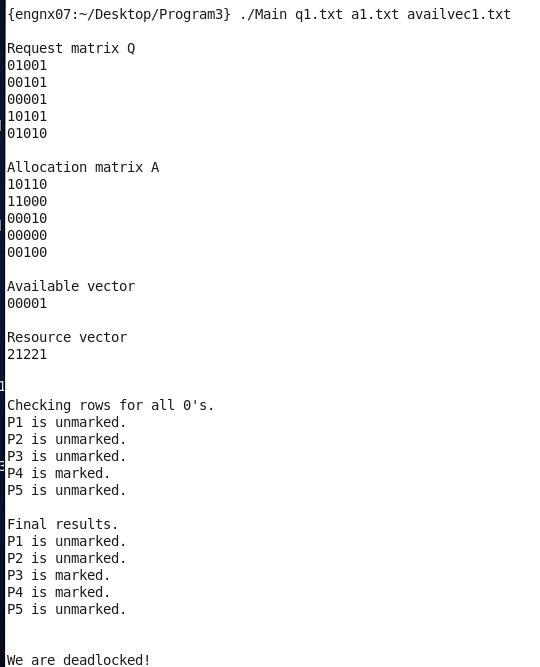


**Letting user choose their filename:** This didn’t take much and was an addition I wanted to add if I had time. Thankfully I did and the user can declare which files to read. There are no checks if the information from the file will work or not (garbage in, garbage out). However, if any of the files won’t open then an error message is given and the program closes. If the user doesn’t give 3 file names then defaults are supplied (q.txt, a.txt, and availvec.txt).

IV. Build/Final Thoughts



This first build shows us the results from the example given in the textbook. No arguments were given so the default file names have been used.



Running the program again but with different file names shows that we can open whichever files we want.

Since I had no experience using C for file reading, this program ended up being a two-part process for me. The first was properly reading from files and writing them to arrays. Once that was complete, the second part was writing the actual deadlock detection algorithm.

I wish I had more time to test different values, but I only had the chance to test two sets of data that were very similar to each other. However, I do not see any reason why this shouldn’t work for any problem set up to a 100x100 matrix.

The issue with differences between Windows and Linux NEW LINE characters does present a problem that I wish I had dealt with and is one that I will make sure to look out for in the future.