Files to submit: **All .c and .h files needed for your solution, a Makefile to compile them all**
Time it took Matthew to Complete**: 3 hours**

- All programs must compile without warnings when using the -Wall and -Werror options
- Submit only the files requested
  - Do **NOT** submit folders or compressed files such as .zip, .rar, .tar, .targz, etc
- Your program must match the output exactly to receive credit.
  - Make sure that all prompts and output match mine exactly.
  - Easiest way to do this is to copy and paste them
- All input will be valid unless stated otherwise
- Print all real numbers to two decimal places unless otherwise stated
- The examples provided in the prompts do not represent all possible input you can receive.
- All inputs in the examples in the prompt are underlined
  - You don't have to make anything underlined it is just there to help you differentiate between what you are supposed to print and what is being given to your program
- If you have questions please post them on Piazza

Restrictions

- No global variables are allowed
- Your main function may only declare variables, call other functions, and assign variables values.
- You must use at least 1 struct in your solution

For your final project you will be implementing a text based version of paint. This project emphasizes all of the topics that we have covered this quarter except for recursion.

Specifications

1. Command Line Arguments
    1. Your program should take as an **optional** command line parameters the number of rows and columns to create on the canvas.
        1. This means your program should be able to run as either
            1. `./paint.out`
            2. `./paint.out num_rows num_cols`
        2. Either both the rows and columns must be specified or neither should
        3. The number of rows and columns must be integers greater than or equal to 1
        4. If no command line arguments are provided or there is an error in either the number of command line arguments or their values then a **default board of size 10 X 10** should be created.
2. Commands
    1. Your program must be capable of accepting and executing the following commands. The bolded letter inside the parentheses is the letter used to specify the command.
        1. (**q**)uit
            1. Cease the execution of the program and free any dynamically created space.
        2. (**h**)elp
            1. Display the help information
                1. I've given you a function to print this to save you the time copying it
        3. (**w**)rite start_row start_column end_row end_column
            1. Draw a line from the starting row and column to the ending row and column
                1. Horizontal lines are drawn using -
                2. Vertical lines are drawn using |
                3. Right diagonal lines are drawn using \
                4. Left diagonal lines are drawn using /
                5. If a line you are drawing intersects a portion of a line drawn using a **different** character than the one you are drawing then a + should be written at the intersecting point
                    1. For example at the point where a horizontal line and a vertical line intersect a + should be drawn there.
                6. If a line you are drawing intersects a portion of a line drawn using the **same** character as the one you are drawing then you should use continue to use the same character
                    1. For example a horizontal line drawn over a horizontal line should all be drawn using a -
                7. Lines should be able to drawn from either direction

         1. For example a horizontal line should be able to drawn from
            1. Left to right to right: `w 0 0 0 5`
            2. Or right to left: `w 0 5 0 0`
      8. Lines that are 1 cell big should be drawn using -
      9. If the line coordinates for the lines entered do not make a straight line you should tell the user and not attempt to draw the line.
4. (**e)**rase row col
   1. Erase the character at row, col reverting it back to a blank space
      1. Blank spaces are represented as * in this program
5. (**r)**esize num_rows num_cols
   1. Resize the board to be num_rows X num_cols big
   2. The smallest the board can be resized to is 1 X 1
   3. New rows are added to the top of the canvas
   4. New columns are added to the right of the column
   5. New rows/columns are empty
   6. The board can also be resized to be smaller than it currently is
6. (**a**)dd [r | c] position
   1. Add a new, empty row/col at the position specified
   2. r stands for row and c for column so
      1. A command to add a new row at position 5 would look like: `a r 5`
      2. And a command to add a new column at position 7 would look like: `a c 7`
   3. Valid values for position are between 0 and num_rows/num_cols + 1 based on whether row or column is specified
   4. If adding a row, rows at and above the position specified are moved up by 1
   5. If adding a new column, columns at and to the right of the position specified are moved 1 to the right
7. (**d**)elete [r | c] position
   1. Delete a row column at the specified position
   2. r stands for row and c for column so
      1. A command to delete a row at position 5 would look like: d `r 5`
      2. And a command to delete a column at position 7 would look like: d `c 7`
   3. Valid values for position are between 0 and num_rows/num_cols
   4. If deleting a row, rows that are above the row deleted shift down by 1
   5. If deleting a column, columns that are to the right of the row deleted shift to the left by 1

8. (**s**)ave file_name
    1. Save the current canvas to the file with the name file_name
    2. How you save the canvas is up to you
9. (**l**)oad file_name
    1. Load a canvas from the file with the name file_name
    2. The structure of this file is up to you but the structure should be consistent with files you create from the save command

Hints

1. This is a big program and you will want to break it down a lot. Trying to do too much at once in one function is likely to lead to a lot of confusion and difficulty.
    1. In total my answer consisted of about 857 lines of C code broken across 43 functions contained in 7 files.
    2. I used 4 structs and 1 enum in constructing my answer.
2. Take the problem apart piece by piece and test each piece as you add it in. This will help a lot in finding bugs and smooth out the development of your program
    1. First start with creating and displaying your canvas
    2. After you get that working add commands in one by one
        1. This will involve reading the command
        2. And then executing the command
        3. Start with the simple commands and then do the more complicated ones.
            1. Consider doing it in the following order
                1. quit
                2. help
                3. draw (this is actually the most complicated command but kind of necessary to do early to be able to see if everything else is working)
                    1. Since it is so complicated consider breaking it down into 4 separate draw functions, one for each direction
                4. Add row/column
                5. Delete row/column
                6. Resize
                7. Save
                8. Load
3. Proper use of structs can be huge in helping to simplify the problem

Example

```
./paint.out
 9  *  *  *  *  *  *  *  *  *  *
 8  *  *  *  *  *  *  *  *  *  *
 7  *  *  *  *  *  *  *  *  *  *
 6  *  *  *  *  *  *  *  *  *  *
 5  *  *  *  *  *  *  *  *  *  *
 4  *  *  *  *  *  *  *  *  *  *
 3  *  *  *  *  *  *  *  *  *  *
 2  *  *  *  *  *  *  *  *  *  *
 1  *  *  *  *  *  *  *  *  *  *
 0  *  *  *  *  *  *  *  *  *  *
    0  1  2  3  4  5  6  7  8  9
Enter your command: w 0 0 9 9
 9  *  *  *  *  *  *  *  *  *  /
 8  *  *  *  *  *  *  *  *  /  *
 7  *  *  *  *  *  *  *  /  *  *
 6  *  *  *  *  *  *  /  *  *  *
 5  *  *  *  *  *  /  *  *  *  *
 4  *  *  *  *  /  *  *  *  *  *
 3  *  *  *  /  *  *  *  *  *  *
 2  *  *  /  *  *  *  *  *  *  *
 1  *  /  *  *  *  *  *  *  *  *
 0  /  *  *  *  *  *  *  *  *  *
    0  1  2  3  4  5  6  7  8  9
Enter your command: w 9 0 0 9
 9  \  *  *  *  *  *  *  *  *  /
 8  *  \  *  *  *  *  *  *  /  *
 7  *  *  \  *  *  *  *  /  *  *
 6  *  *  *  \  *  *  /  *  *  *
 5  *  *  *  *  \  /  *  *  *  *
 4  *  *  *  *  /  \  *  *  *  *
 3  *  *  *  /  *  *  \  *  *  *
 2  *  *  /  *  *  *  *  \  *  *
 1  *  /  *  *  *  *  *  *  \  *
 0  /  *  *  *  *  *  *  *  *  \
    0  1  2  3  4  5  6  7  8  9
Enter your command: w 4 2 4 4
 9  \  *  *  *  *  *  *  *  *  /
 8  *  \  *  *  *  *  *  *  /  *
 7  *  *  \  *  *  *  *  /  *  *
 6  *  *  *  \  *  *  /  *  *  *
 5  *  *  *  *  \  /  *  *  *  *
 4  *  *  -  -  +  \  *  *  *  *
 3  *  *  *  /  *  *  \  *  *  *
 2  *  *  /  *  *  *  *  \  *  *
 1  *  /  *  *  *  *  *  *  \  *
 0  /  *  *  *  *  *  *  *  *  \
    0  1  2  3  4  5  6  7  8  9
Enter your command: w 4 6 4 1
 9  \  *  *  *  *  *  *  *  *  /
 8  *  \  *  *  *  *  *  *  /  *
 7  *  *  \  *  *  *  *  /  *  *
 6  *  *  *  \  *  *  /  *  *  *
 5  *  *  *  *  \  /  *  *  *  *
 4  *  -  -  -  +  +  -  *  *  *
 3  *  *  *  /  *  *  \  *  *  *
 2  *  *  /  *  *  *  *  \  *  *
 1  *  /  *  *  *  *  *  *  \  *
 0  /  *  *  *  *  *  *  *  *  \
    0  1  2  3  4  5  6  7  8  9
Enter your command: w 8 6 0 6
 9  \  *  *  *  *  *  *  *  *  /
 8  *  \  *  *  *  *  |  *  /  *
 7  *  *  \  *  *  *  |  /  *  *
 6  *  *  *  \  *  *  +  *  *  *
 5  *  *  *  *  \  /  |  *  *  *
 4  *  -  -  -  +  +  +  *  *  *
 3  *  *  *  /  *  *  +  *  *  *
 2  *  *  /  *  *  *  |  \  *  *
 1  *  /  *  *  *  *  |  *  \  *
 0  /  *  *  *  *  *  |  *  *  \
    0  1  2  3  4  5  6  7  8  9
Enter your command: a r 3
10  \  *  *  *  *  *  *  *  *  /
 9  *  \  *  *  *  *  |  *  /  *
 8  *  *  \  *  *  *  |  /  *  *
 7  *  *  *  \  *  *  +  *  *  *
 6  *  *  *  *  \  /  |  *  *  *
 5  *  -  -  -  +  +  +  *  *  *
 4  *  *  *  /  *  *  +  *  *  *
 3  *  *  *  *  *  *  *  *  *  *
 2  *  *  /  *  *  *  |  \  *  *
 1  *  /  *  *  *  *  |  *  \  *
 0  /  *  *  *  *  *  |  *  *  \
    0  1  2  3  4  5  6  7  8  9
Enter your command: a c 6
10  \  *  *  *  *  *  *  *  *  *  /
 9  *  \  *  *  *  *  *  |  *  /  *
 8  *  *  \  *  *  *  *  |  /  *  *
 7  *  *  *  \  *  *  *  +  *  *  *
 6  *  *  *  *  \  /  *  |  *  *  *
 5  *  -  -  -  +  +  *  +  *  *  *
 4  *  *  *  /  *  *  *  +  *  *  *
 3  *  *  *  *  *  *  *  *  *  *  *
 2  *  *  /  *  *  *  *  |  \  *  *
 1  *  /  *  *  *  *  *  |  *  \  *
 0  /  *  *  *  *  *  *  |  *  *  \
    0  1  2  3  4  5  6  7  8  9 10
Enter your command: d r 10
 9  *  \  *  *  *  *  *  |  *  /  *
 8  *  *  \  *  *  *  *  |  /  *  *
 7  *  *  *  \  *  *  *  +  *  *  *
 6  *  *  *  *  \  /  *  |  *  *  *
 5  *  -  -  -  +  +  *  +  *  *  *
 4  *  *  *  /  *  *  *  +  *  *  *
 3  *  *  *  *  *  *  *  *  *  *  *
 2  *  *  /  *  *  *  *  |  \  *  *
 1  *  /  *  *  *  *  *  |  *  \  *
 0  /  *  *  *  *  *  *  |  *  *  \
    0  1  2  3  4  5  6  7  8  9 10
Enter your command: d c 5
 9  *  \  *  *  *  *  |  *  /  *
 8  *  *  \  *  *  *  |  /  *  *
 7  *  *  *  \  *  *  +  *  *  *
 6  *  *  *  *  \  *  |  *  *  *
 5  *  -  -  -  +  *  +  *  *  *
 4  *  *  *  /  *  *  +  *  *  *
 3  *  *  *  *  *  *  *  *  *  *
 2  *  *  /  *  *  *  |  \  *  *
 1  *  /  *  *  *  *  |  *  \  *
 0  /  *  *  *  *  *  |  *  *  \
    0  1  2  3  4  5  6  7  8  9
Enter your command: r 4 5
3 * * * * *
2 * * / * *
1 * / * * *
0 / * * * *
  0 1 2 3 4
Enter your command: e 1 1
3 * * * * *
2 * * / * *
1 * * * * *
0 / * * * *
  0 1 2 3 4
Enter your command: r 7 7
```

```
6 * * * * * * *
5 * * * * * * *
4 * * * * * * *
3 * * * * * * *
2 * * / * * * *
1 * * * * * * *
0 / * * * * * *
  0 1 2 3 4 5 6
Enter your command: s ex.txt
6 * * * * * * *
5 * * * * * * *
4 * * * * * * *
3 * * * * * * *
2 * * / * * * *
1 * * * * * * *
0 / * * * * * *
  0 1 2 3 4 5 6
Enter your command: r 1 1
0 /
  0
Enter your command: l ex.txt
6 * * * * * * *
5 * * * * * * *
4 * * * * * * *
3 * * * * * * *
2 * * / * * * *
1 * * * * * * *
0 / * * * * * *
  0 1 2 3 4 5 6
Enter your command: q
```