# Program Evaluation Instructions:

Please make sure that everything you write in this document uses a RED COLOR to make your comments stand out from the instructions.

**Initial evaluation questions:**Does the program compile? Put an ‘X’ under the appropriate column *(yes* or *no)*. If the answer is *No*, you MUST provide comments/feedback that explain what the problem is. Be as specific as possible, as the other group is dependent on your accurate comments to help improve their program.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Yes** | **No** | **Comments/Feedback** *(required for “No” answer)* |
| Does the program compile? | X |  |  |

*If the program does not compile, then the evaluation cannot continue.   
The program has failed the assignment, and you do not need to fill out the rest of this evaluation form.*

**Evaluation:**

Rate each criterion below. For each one, put an ‘X’ under the appropriate column *(great, good, fair,* or *poor)*. For each item which is not marked as *Excellent*, you MUST provide comments/feedback that fully support the rating that you have given. Be as specific as possible, as the other group is dependent on your accurate comments to help improve their program. Test cases are provided on the next few pages.

Rating Meanings:

* Excellent: There are **NO PROBLEMS AT ALL** with the program for this particular criteria item.
* Good: The problem has one or more **MINOR** issues with this criteria item *(i.e., small mistakes; easily correctable)*
* Fair: The program has **MAJOR** problems with this criteria item *(i.e., group has ignored/failed many of the items in this criteria; program is not following standard coding practices; mistakes made are causing major problems with the program execution, logic, memory issues, etc.)*
* Poor: The program has completely **FAILED** this criteria item *(i.e., group has completely ignored this criteria item or made little to no attempt to implement this item in the program; program completely fails this criteria item; the mistakes made will take significant effort to correct)*

# Criteria & Ratings:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Criteria** | **Rating** | | | |  |
| **Excellent** | **Good** | **Fair** | **Poor** | **Comments/Feedback** *(required for non-Excellent ratings)* |
| Program compiles without any warnings | **X** |  |  |  |  |
| Program is correctly formatted/indented |  |  |  |  |  |
| Program is commented correctly; Comments are clear, correct, complete |  |  |  |  |  |
| Variable names are sensible |  |  |  |  |  |
| Program does not attempt to read/write past the end of any arrays used |  |  |  |  |  |
| Follows all guidelines and requirements that have been set for this assignment *see requirements listed below* |  |  |  |  |  |
| Program works on all test cases *see test cases listed below* |  | **X** |  |  | Fill and Undo are the only features that don’t really work |

# Additional Comments:

Use this area to write additional comments if you need more space than what is provided in the table above.

# Requirements

If the program fails to meet requirements, you **MUST** add comments under the failed requirement below indicating the problem that you found within the program. Comments should be RED in color. Please do not write comments such as “passed” if the program passed the requirement; only comment if the requirement failed.

**General Requirements:** There are some general requirements that all programs in this course should follow, such as:

* Do not declare any variables with the **string** data type
* Do not declare any global variables *(global constants are allowed – i.e., has the word* ***const*** *in front)*
* Do not use **break** statements to exit a loop *(break is needed to exit a switch statement however)*
* Every function must begin with a set of comments which describe the purpose of the function, the purpose of each parameter the function requires, the changes the function will make to the reference parameters, and the purpose of the return value (if not void)
* Comments should be written throughout the code to explain what the code does. Typically, comments should be written above the lines that they are explaining – and not out to the side of those lines

**Specific Requirements:** The following are requirements specific to this program that you should look for in the source code:

1. The numbers 20, 255, 240, 22, or 80 should not be used anywhere within the program code (i.e. inside of a function). The only place they should appear is in the constants defined at the top of the program. Instead the constants should be used whenever a reference to these numbers are needed.
2. Functions should do EXACTLY what the assignment called for – nothing more and nothing less.
3. The three recursive functions MUST be written to work using recursion, and not using loops:
   1. *drawBoxesRecursive*
   2. *fillRecursive*
   3. *treeRecursive*
4. Comments should be written throughout the code to explain what the code does. Typically, comments should be written above the lines that they are explaining – and not out to the side of those lines

# Test Cases

If the program fails a test case, you **MUST** add comments under the failed test case below indicating the problem that you found when executing the test case. Comments should be RED in color. Please do not write comments such as “passed” if the program passed the requirement; only comment if the test case failed.

If any test case fails, stop the program, and restart it again before continuing on with additional test cases. If you can't complete a test because a previous one isn't working correctly – this should be noted *(for example, if the program doesn't correctly print an initial file to the screen, then none of the other test cases will be able to be performed)*. I have listed the basic things that the program should do. If you note anything odd about how the program reacts, or the output from the program, please indicate your findings.

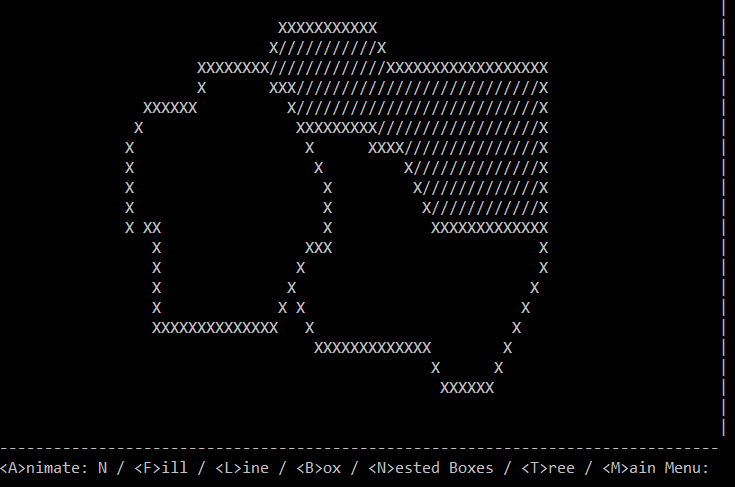
1. *Test Initial State:* Start the program.
   1. The program should have a <*D>raw* option and a *<C>lear* option added to the main menu, in addition to all other menu options that were previously there.  
        
      
   2. Do a quick test of the Edit, Move, Replace, Load and Save options to make sure they still work correctly. This is to ensure that no changes to the program have caused these options to stop working.
2. *Test Clear:* Start the program.  
   1. Enter **L**. For the filename, enter **ecu**. The program should display the ECU.txt file contents on the screen (which is simply huge ‘ECU’ letters).
   2. Enter **C**. The screen should be cleared back to a blank canvas.

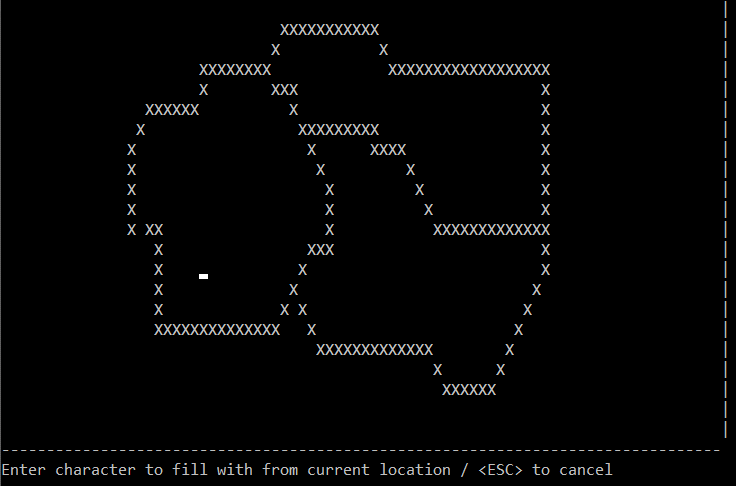
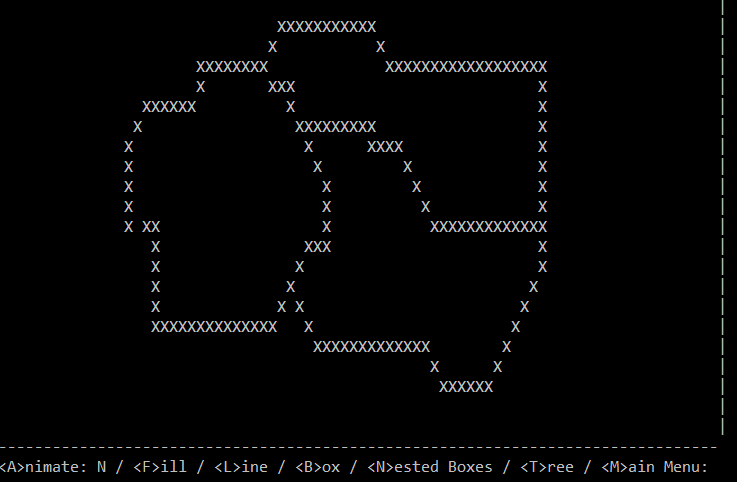
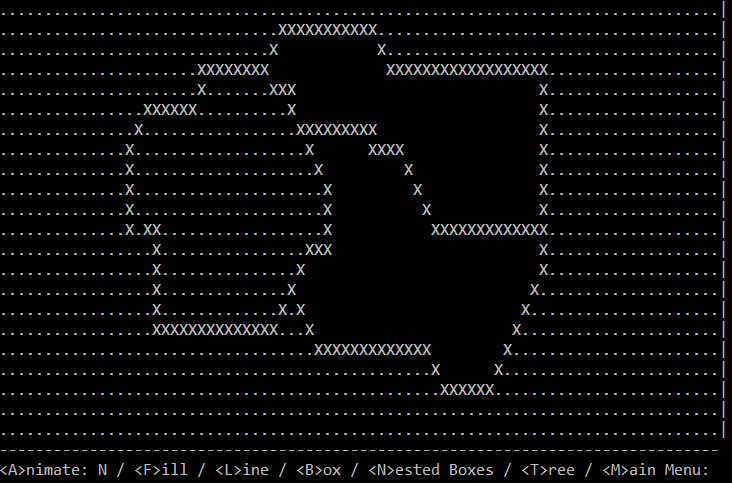
1. *Test Draw Menu:*   
   1. Enter **D.** The program should replace the main menu with the draw menu:  
        
      
   2. Enter **P**. Since this is not a valid menu option, the program should not do anything, and should remain with the draw menu displayed.
   3. If the draw menu is not on the screen, get back to the draw menu.  
      Press **M**. The program should remove the draw menu from the screen and go back to the main menu.

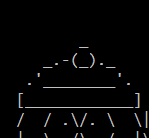
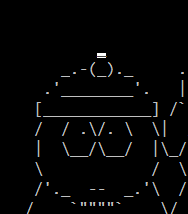
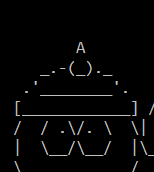
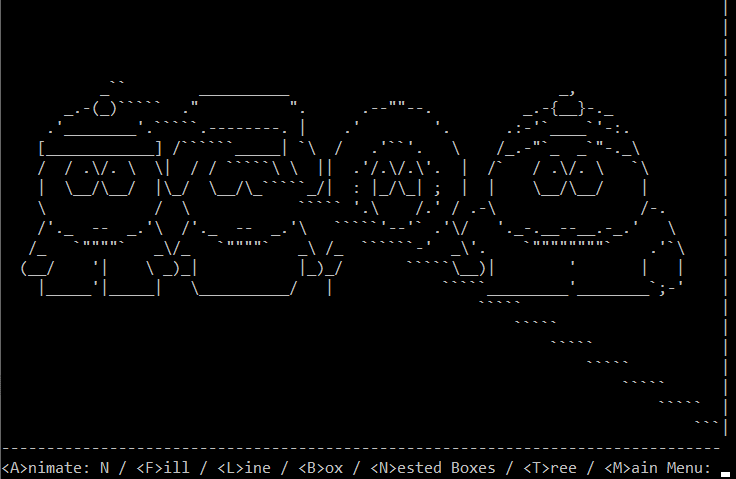
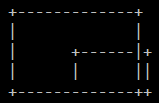
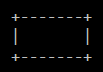
**Note: For ALL remaining tests:**

* All prompts printed to the screen, from options in the draw menu, should appear at the screen location of where the menu was previously printed. Unlike the options from the previous program which may print 2-3 lines of output, under the canvas, all of the options from the draw menu should only print a single line of output under the canvas. Each new prompt should erase the previous one and print at the same location as the previous one.
* Please make sure that the tests are **NOT PERFORMED inside of a shared terminal window** while doing a Live Share with your group. The shared terminal will usually not allow these tests to be accurately performed.

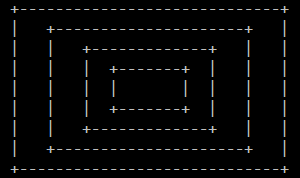
1. *Test Fill:*   
   1. From the main menu, enter **L** and load the **blobs2** file.
   2. From the draw menu, enter **F**. You should be asked to: *Enter character to fill with from current location / <ESC> to cancel*
   3. Using the arrow keys, move the cursor to the approximate point shown in the picture to the right.
   4. Try pressing **F4**, **Backspace**, **Tab**, **Enter**. None of these keys should have any effect. If they do, please restart test case 4 again before proceeding any further.
   5. Enter **/**. The program should fill the blob in which the cursor was placed with slashes as shown below, and the draw menu should reappear on the screen. Did not work. Went crazy, canvas is gone, just see draw menu

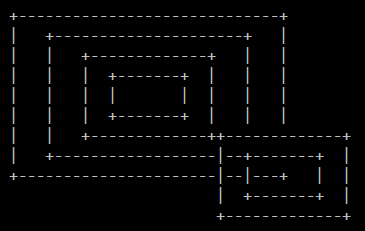


* 1. Enter **F**. Move the cursor back to approximately the same location on the screen. Press **Space**. The program should fill the blob in which the cursor was placed with spaces so that it appears as it did originally.  
       
     
  2. Enter **F**. Move the cursor to the approximate location shown here --->
  3. Press **period**. The program should fill the blob in which the cursor was placed with slashes, as well as the space outside the blob, as shown below:  
       
     
  4. Go back to the main menu and enter option **U** to undo the fill. All of the periods should be removed from the image.
  5. Enter **F**. Leave the cursor in the top left corner of the screen and press the **period**. The image should be filled with periods again as was seen in case <h> above.
  6. Enter **F**. The fill prompt should appear. Enter **ESC** to cancel the fill operation. The draw menu should reappear.
  7. Go back to the main menu and enter option **U**. All of the periods should be removed from the image *(i.e., the fill from case <j> should still be able to be undone, even though you cancelled the fill operation from case <k>)*.

1. *Test Line:*   
   1. From the main menu choose option **L** and load the file: **southpark**.
   2. From the draw menu, enter **L**. You should be asked to: *Type any letter to choose start point / <ESC> to cancel*
   3. Move the cursor to the top of the first character’s hat:   
      and then press **A** to mark the start point. The program should have left an “A” on the screen at the spot that  you marked, and the prompt should now say: *Type any letter to choose end point / <ESC> to cancel*
   4. Press **ESC** to cancel the line drawing. The “A” which was on the screen should be removed, and the image should be back like it was before (with a dash where the “A” was).   
      the A was removed but a line was drawn from the top right corner to the top of the character’s hat
   5. Repeat step <c> to put the “A” back in place again.
   6. When asked for an end point, move the cursor to the bottom right corner of the screen, and press any letter key. After doing so, a line should be drawn from the top of the first character’s hat to the bottom right corner of the screen similar to this:  
        
      
   7. Go back to the main menu and choose option **U** to undo the line. The line should be removed from the screen and the characters should be back as they were before the line was drawn.  
      did not undo the line
2. *Test Box:* Start this test with a blank canvas.  
   1. From the draw menu, enter **B**. You should be prompted to: *Enter size:*
   2. Enter **2**. You should be prompted to: *Type any letter to choose box center, or <C> for screen center / <ESC> to cancel*
   3. Enter **C**. A box of size 2 should be drawn in the center of the screen. 
   4. Enter **B**. Enter size **4**. Move the cursor to the top left corner of the previously drawn box. Press any letter key (except C). A second box should be drawn, and the screen should look similar to this:

* 1. Enter **B**. Enter size **8**. Press **ESC** to cancel.
  2. Go back to the main menu and choose **U** to undo. The box of size 4 should be removed from the screen.  
     didnt undo

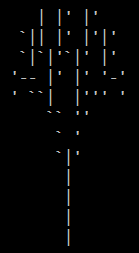
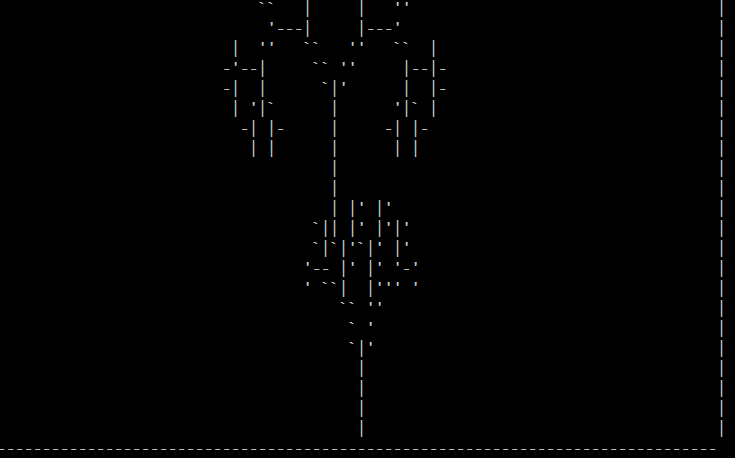
1. *Test Nested Boxes:* Start this test with a blank canvas.  
   1. From the draw menu, enter **N**. You should be prompted to: *Enter size of largest box:*
   2. Enter **9**. You should be prompted to: *Type any letter to choose box center, or <C> for screen center / <ESC> to cancel*
   3. Enter **c** (lowercase). Four nested boxes should be drawn in the center of the screen.



* 1. Enter **N** and enter size **4**.
  2. Position the cursor at the bottom right corner of the largest box that was previously drawn.
  3. Press any letter other than “c”. The following figure should now be present on the screen:
  4. Enter **N**. Enter size **8**. Press **ESC** to cancel.
  5. Go back to the main menu and choose **U** to undo. The nested boxes of size 4 should be removed from the screen.

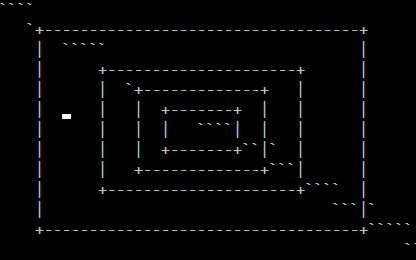
Didn’t undo

1. *Test Tree:* Start this test with a blank canvas.  
   1. From the draw menu, choose **T.** You should be prompted to *Enter approximate tree height:*
   2. Enter **12**. You should be prompted to *Enter branch angle:*
   3. Enter **30**. You should be prompted to *Type any letter to choose start point, or <C> for bottom center / <ESC> to cancel*
   4. Enter **C**. A tree similar to the following should have been drawn at the bottom center of the screen, and the draw menu should be back on the screen. Keep in mind that it may not look exactly the same as the one shown here, but should be a pretty close match.



* 1. Choose **T.** Enter size **15**. Enter angle **45**.   
     Move the cursor to the top left corner of the previously drawn tree. Enter any letter other than “c”. The result should be similar to the diagram shown here on the right.
  2. Choose **T.** Enter size **10**. Enter angle **25**. Enter **ESC**.
  3. Go back to the main menu and choose **U** to undo. The most recently drawn tree should be removed from the screen.

Doesn’t undo

1. *Test Animate:* Start this test with a blank canvas.  
   1. At the draw menu, the animate option should show “N”: 
   2. Enter **A**. The animate option should now show “Y”: 
   3. Enter **M** to return to the main menu.
   4. Enter **D** to return to the draw menu. The animate option should still show “Y”. If it does not, restart the program again, and then repeat steps <a> and <b> above. Goes back to N
   5. Draw a line from the top left corner to the bottom right corner of the screen. You should see the line animated when drawn.
   6. Draw a box in the center of the screen with size **10**. You should see the box drawing animated.
   7. Draw nested boxes in the center of the screen with size **6**. You should see the box drawing animated, and no box should be drawn more than once.
   8. Choose the fill option, and position the cursor at approximately the location shown in the image to the right. Fill with any character of your choice. The fill should be animated when drawn.  
      fill doesn’t work and doesn’t allow you to enter anything or escape after

* 1. Draw a tree with height 12, and branch angle 60. The tree drawing should be animated.