Friday 12/13: V.1.c: Tested for errors. Debugged some of the code and changed it to make sure the rest of the commands aren't silenced after an expression is interpreted, and fixed a NullPointerException by adding a check for a null argument.
Thursday 12/12 at home: V.1.b and V.1.c: Changed ParseTree and Parser and the lexer so that they would support the entering of sci commands wrapped in angle brackets. Added a syntax checker to ParseTree and integrated parsing with the rest of the project. Tested for errors.
Thursday 12/12: V.1.b and V.1.c: Fixed the implementation of unary operators and started work on supporting commands. Wrote the ParseTree class to parse an Abstract Syntax Tree of nodes.
Wednesday 12/11: V.1.b: Fixed the error described below: turns out that the extra parentheses made the local left and right null and then overrrode the left and right of the returned middle node from the recursive call. Started testing unary operators.
Tuesday 12/10: V.1.b: Debugged the parser, found that it was passing the wrong index to the recursive call, fixed that. Added toString method to Node class to visualize trees. Tested the parser. It works for "3", "3 + 4", and "3 + 4 * 6", but not "(3 + 4) * 6" (produces a tree with only the node for "*"). Continued debugging.
Monday 12/09: V.1.b: Continued working on the parenthetical expression parser. Ran into a problem: originally, I thought that inner parenthetical expressions would only be either left or right, but numbers are also parenthesizes, and they need to be the middle of their own node, so how do I work around that? Thought about it.

Friday 12/06: V.1.b: Continued working on an original parser to recursively parse parenthetical expressions. I'm a little uncertain about the exact procedure, but I have the general idea worked out for the algorithm.
Thursday 12/05: V.1.b: Implemented the algorithm to properly parenthesize expressions and started working on a method to parse parenthetical expressions. Not sure how it'll turn out, but I think it could work.
Wednesday 12/04: V.1.b: Looked over created grammar tried to imagine how a recursive descent parser would work. I decided it wouldn't properly account for precedence, so I researched other solutions and cogitated upon several found here:
https://en.wikipedia.org/wiki/Operator-precedence_parser#Precedence_climbing_method
Tuesday 12/03: V.1.b: Ran into a road block: how should I take operator precedence into account? I researched this problem and ran into a couple solutions, but none to my liking. Thought about it and started writing a revised grammar to take precedence into account
Monday 12/02: V.1.b: Wrote grammar for accepted expressions. Started working on creating the functions for a recursive descent parser, since my grammar is pretty simple. Having no difficulty so far. Inspired from here:
https://stackoverflow.com/questions/25049751/constructing-an-abstract-syntax-tree-with-a-list-of-tokens

Friday 11/22: V.1.a, V.1.b: Added auxiliary lexing methods to remove whitespace tokens and tell total length of all captured tokens (might be useful when parsing). Reorganized folder structure

with all of the parsing stuff. Created Node and Abstract Syntax Tree classes. Started reading about parsing tokens.
Thursday 11/21: V.1.a: Finished implementing the lexer and made sure lists could only be alphanumeric + underscores (not starting with a number so that the lexer can distinguish between numbers and strings).
Wednesday 11/20: V.1.a: I started working on implementing the lexer by creating a TokenType enum and Token class to represent the tokens the lexer will parse the entered string into. I then worked on the actual lexing of the string, using regex and named groups.
http://giocc.com/writing-a-lexer-in-java-1-7-using-regex-named-capturing-groups.html
Tuesday 11/19: IV.7: I fixed the error handling by checking for the scenario in which the error is thrown and dealing with it already, so it now correctly saves the images and has pretty error handling. I started researching lexers in Java.
Monday 11/18: IV.7: instead of taking a screenshot, I directly drew the image onto an image buffer, which I then write to a file. I tested, and it works as designed, except the error handling with ImageIO.write is wonky.

Friday 11/15: IV.6, IV.7: added the categorical version of the boxplot and started work on saving the image. I created the command and was able to take a screenshot, but it doesn't wait until the window gets focus.
Thursday, 11/14: IV.6: finished drawing the rest of the boxplot onto the window and drew the scale and numbers on the x-axis to go along with it.
Wednesday, 11/13: IV.6: I worked out the logic for making the pixel spaces between the numbers of the summary proportional to the actual difference between the numbers and drew the skeleton of the boxplot.
Tuesday, 11/12: IV.5 and IV.6: I fixed some division by zero errors in certain cases when calculating the auto values. I started work on the boxplot, creating the command, and passing in the neessary five number summary of the data.
Monday, 11/11: IV.5: I created the code to automatically determine the step and minimum value for the histogram (no algorithm or anything, I just kinda tweaked until I liked the look). I then made sure that the colors only accounted for the bars actually drawn (so that no color gets skipped). I also created the quantitative version of the hist command too.

Friday, 11/7: IV.5: I copied the code from the bar graph to calculate the yaxis labels and the heights of each bar, and fitted it to the histogram function. I then tested the code. It worked first try; yay!
Thursday 11/7: IV.5: I fixed the code to make the x-axis labels; I had drawn them off screen by mistake.
Wednesday, 11/6: IV.5: tried to make the x-axis lines and labels for the histogram. I am still debugging it; for some reason, nothing is drawn on the screen, but the code does execute
Tuesday, 11/5: IV.5: started working on the categorical histogram command. I thought about the implementation of the actual drawing of the histogram. did the basic framework of the command

Monday, 11/4: IV.4: apparently it wasn't ironed out; the angle measures had to be in radians, so I fixed that. I also fixed an error where the first and last sections of the pie chart were the same color.

Friday, 11/1: IV.4: fixed the drawing of the pie chart, it had been not finding the percentage of the circle correctly, so I fixed that. put labels on the pie chart, had some trouble with the trig, but I eventually got it ironed out. shrinked the pie chart size slightly.
Thursday, 10/31: IV.4: forgot to cast something to a double, so it truncated my percentage to 0 and nothing was showing up. fixed that and now the pie chart should work. will work on tuning the size and colors tomorrow.
Wednesday, 10/30: IV.3 and IV.4: fixed the scale bug by accounting for the width of each scale line, started work on pie charts by calculating the percentage for each string value
Tuesday, 10/29: IV.3: worked on when the data is bigger than the y-axis can handle (when going by ones), had trouble with getting the coefficient and the scales to the correct values and aligning that with the bars
Monday, 10/28: IV.3: worked on drawing the scale of the bars and y-axis, had trouble with aligning the labels on the y-axis and keeping the two scales/heights aligned

Friday, 10/25: IV.3: drew the bars and their labels onto the window, chose colors for the bars
Thursday, 10/24: IV.3: got input from console for the labels and title of the bar graph, added the actual bar graph command, worked on spacing the bars and deciding sizes
Wednesday, 10/23: IV.2 & IV.3: allowed for changing the drawing on the window, worked on drawing axes and title of a graph
Tuesday, 10/22: IV.2: worked on drawing on the GUI window (rectangles, axes, text)
Monday, 10/21: IV.1 & IV.2: created the object that allows the graphing module to interact with the command interface, added show and hide commands to show/hide the GUI window

Friday, 10/18: III.3 added the categorical list versions of the distribution and five number summary commands, ended up fixing an error relating to calling other commands
Thursday, 10/17: III.3, added the distribution command for quantitative lists
Wednesday, 10/16: PSAT
Tuesday, 10/15: III.3, added the five number summary command for quantitative lists

Thursday, 10/10: III.3
Wednesday, 10/9: III.2 and III.3
Tuesday, 10/8: III.2
Monday, 10/7: III.2

Friday, 10/4: III.2
Thursday, 10/3: III.2
Wednesday, 10/2: III.2
Tuesday, 10/1: III.2
Monday, 9/30: III.1 and III.2

Friday, 9/27: II.3 and II.5
Thursday, 9/26: II.2 and II.4
Wednesday, 9/25: I.6, II.1, II.2
Tuesday, 9/24: I.5 and I.6
Monday, 9/23: I.5 and I.6