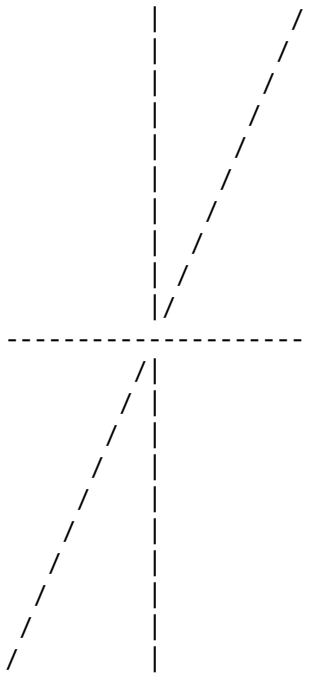
Beard Loading APCS1 pd 3
Ishtiaque Mahdi, Shakil Rafi, Mansour Elsharawy
1/9/16

Final Project Proposal v2:

Line Forge

Line Forge is a graphing calculator set in the terminal. Upon running the program, the user can either type **help** for information about how to use the program or jump right into using it. Our program has two main uses. The first is to evaluate an expression, utilizing a class MathString that takes the string representation of a mathematical expression or equation and handles an appropriate output. If the user inputs an expression (ie: does not have an equal sign), then this part of the program will be activated. For example, the user could input something like **3+4*2** and the program will output **11**. We will do this by first splitting the expression into numbers, operators, and parentheses and then simplifying the expression using the order of operations. The second use is to graph functions. If the user inputs an equation (ie: has an equal sign), then this part of the program will be activated. For example, the user could input y=x and the program would output the graph with window frame of [-10,10] using class AxisGraph, which displays a 2D array of Points (a class we will make that implements x and y coordinates according to the Cartesian system. Each Point in AxisGraph would plug in its x and y values into the equation and check for equality to a certain degree):



After graphing the given equation, the program will ask for further input where the user can perform multiple operations such as zoom, translate, or scale the graph to get a better view of the specific part of the graph that they need to focus on. Then, they can go on to graphing another function or evaluating an expression.

MVP:

A program that can ask for and accept expression inputs to be evaluated. At first, the supported expressions would only be simple one-operation expressions like **3+1** or **5*8** and then would be expanded into supporting any size expressions, parentheses, and exponents (using ^).

Features to add:

Should time allow it, we would like to add these features to Line Forge (in order of priority)

- Storing functions into letters like f(x)
- Graphing multiple functions together (and implementing them on top of one another. Ex: g(x) = f(2x) for a previously graphed f(x))
- Matching the graph size to the terminal size
- Support for common functions such as sin, cos, tan
- Support for graphing inequalities
- Equation solver (give the program an equation that includes one variable and it will solve for that variable)
- Dfferent functions being differentiated by different colors using ANSI escape codes

UML - Line Forge

MathString

- String mathSentence

+ ArrayList knownFuncs

+ String functions (find,replace,etc)

+ basic operator evaluation (+,-,x, /)

+ PEMDAS evaluator

+ parenthesis evaluation (3 (5 + 7))

+ pipe evaluation (| x + 2 |)

+ function evaluation (f(x), h(x))

+ equality test

+ equation solver (maybe)

Woo

+ main

Function

- String tag (ex “f”)

- String exp (ex. “x^2 + 5”)

Point

+ double x

+ double y

+ String onOrOff

+ SetOn()

+ SetOff()

Axis Graph

- Point [][] graph

+ graphRelation (takes a and y)

+ graphFunction (function def, save in Math String

+ constructor (assigns coordinates to the points of the array in accordance with some default range)
+ Range (maxVal for x and y)
+ Move (delta x, delta y)