Team Gwuacamohli APCS1 pd08
Clara Mohri, Wenting Li, Jessica Wu 2018-01-02

# Final Project Proposal
# Gwuacamohli's Mathematical Utility

1. Graphing functionality:
   Grapher class
   Attributes:
   - int xmin, xmax, ymin, ymax
   -String[][] _graph
   a. User is able to graph a function that is inputted
      i. Input will hopefully be a string
         If this is unattainable, the user will be asked for a, b, c, d, etc. values in the function (as in ax^n + bx^n-1… )
         1. **Big part:**
            parsing string to turn into mathematical equation
            This will be done by using the .split() method for Strings
            Strings will be split by mathematical operations: +, -, /, *
            This way, the equation will be contained in a String array
            Any x in the string can be identified using .indexOf method
            A for loop goes through all the domain and all these numbers are raised to the exponent (if there is an exponent). Using .indexOf("^"), the exponent is found, and the number following this is the criteria for the for-loop.
            Finally, these x values from the loop are multiplied by the coefficient. Coefficient found by knowing the index of x, and knowing the only thing in front of this is the coefficient
            Converting Strings to Integers is done using Integer.parseInt()
            Every value found this way replaces the expression
   b. User can also determine domain and range of viewing window for graph
      This is done by having setRange and setDomain methods
      These methods change the values for xmin, ymin, xmax, ymax
   c. The graph will be made using the concept of a matrix as a graph. • symbols will indicate a point on the graph. The graph will be made using a for-loop that goes through a mathematical expression for all the x values contained in the domain
   d. User can ask for zeros, y-intercepts, average slope on viewing window, and other mathematical interests

e. User can ask for table of vals for x and y of an equation
   Table class
   Attributes:
   -xmin, xmax
f. Graphing more than one equation
   i. Maybe the user can ask for intersection of graphs

2. Simple math:
   Calculator class
   Attributes:
   -ArrayList<Object> []
   (Contains parsed input)
   a. If user inputs for example "3+3", then 6 will be returned
   b. Parsing Strings as explained in 1 will be also used here
   c. + , - , * , / , ^
   d. Potentially, we would like to find a way to implement PEMDAS for more complicated math
3. Spreadsheet / Table
   Spreadsheet class
   Attributes:
   -int numcols, numrows
   -Object[][] _table
   a. User determines amount of cols and rows
   b. User can have statistics for rows and cols
   c. User can ask for basic statistical calculations including mean, median, mode
   d. User can sort rows/columns if instanceof Comparable

**To-do List:**

1. Create a functional Grapher class, in which user is asked for coefficients for each instead of gets to input String equation
   a. Domain and range can be changed
2. Create a functional Spreadsheet class
3. Calculator class, using String parsing similar to that of Grapher
4. Later: Parsing equations/user input as described
5. Calculating zeros, intercepts for _graph
6. Plotting several functions on the same _graph
7. Further expansion

**How we are making meaningful use of OOP:**

- The Grapher class has its own methods to graph functions. It also has attributes and its own toString() method.
- The Table class has methods for filling in a matrix based on an equation. It also has its own toString() method.
- The Calculator has an attribute of type ArrayList<Object> that contains the parsed user input
- The Spreadsheet class has methods for data analysis, attributes for numcol and numrow, and its own toString() method.

**How we are making meaningful use of first term concepts:**

- Using two-dimensional arrays for our matrices
- Overriding toString methods
- Using ArrayList
- Using for loops
- Using instanceof boolean
- Using sorting algorithms to sort rows/cols
- Using constructors and overloaded constructors
- Using Keyboard interface