# Description of Program

MathTools is a collection of functions for working with mathematical objects, designed to work in a similar manner to the Wolfram language.

The program consists of several features that would be useful to a math student:

* Basic graphing
* Calculation of derivatives
* Calculate of definite integrals
* Basic simplification of equations

# Inputs to Program

The program is used by calling any of the methods

To use the program as a library, [[todo: how to import other projects as libaries]] and then make use of any of its classes and their methods.

To interact with the program's graphical user interface, simply double click the precomiled .jar file to begin. Then, enter your equation, and what sorts of operations you'd like to perform.

# Classes

## Expression

A recursive class that builds up large expressions from small pieces.

### Fields

|  |  |
| --- | --- |
| Expression left | The piece of the expression on the left of the operator |
| Expression right | The piece of the expression on the right of the operator (unused if expression only has one argument, such as sine) |
| String op | The operation for the expression |
| boolean isStatic | Defines the expression as a constant, such as a given integer. |
| double val | The value of the expression, if it is a constant |
| Expression derivative | Stores the derivative when it is found, so it doesn't have to be computed again |

### Methods

|  |  |
| --- | --- |
| public Expression() | Creates an expression which counts as the independent variable (x) |
| public Expression(double i) | Creates a static expression with the value of i |
| public Expression(Expression leftArg, String opArg, Expression rightArg) | Creates an expression with the given left and right hand arguments, and the given operator |
| public Expression(String opArg, Expression arg) | Creates an expression with the given argument and the given operator, for cases where only one argument is taken, such as sine |
| public Expression(String arg) | Creates an expression from parsing the string |
| public double eval(double x) | Returns the value of the function at a given x value |
| public double eval() | Returns the value of the function at x=0, useful for expressions which are constant. |
| public double integrate(Interval i) | Given an interval, returns the value of the integral over that interval |
| public double integrate(String s) | Parses the interval to an interval object, then returns the value of the integral over that interval |
| public Expression derivative() | Calculates the derivative of the function |
| public String toString() | Returns the string representation of the function for easier reading and debugging |

## Graph

### Fields

|  |  |
| --- | --- |
| double xStart, yStart, xEnd, yEnd | Doubles that indicate the range and domain of the graph |
| double xTickInterval, yTickInterval | Doubles indicating the interval between tick-marks on the x and y axes |
| int width, height | The width and height of the graph, in pixels |
| ArrayList<Relation> relations | A list of relations to be graphed |
| boolean useTicks | Whether or not to put tick-marks on the axes |
| boolean tickLines | True if the ticks should extend all the way across the screen, false for line segments as tick-marks. |

### Methods

|  |  |
| --- | --- |
| public Graph(int width, int height, double xStart, double yStart, double xEnd, double yEnd, double xTickInterval, double yTickInterval) | Construct a graph with the given height, width, domain, range, and distance between tick marks. |
| public Graph(int width, int height, double xStart, double yStart, double xEnd, double yEnd, double xTickInterval, double yTickInterval, Relation r) | Construct a graph with the same parameters as above, with a relation. Shortcut for creating a graph and then adding a relation. |
| public Graph(int width, int height, double xStart, double yStart, double xEnd, double yEnd, double xTickInterval, double yTickInterval, Expression e) | Construct a graph with the same parameters as the first constructor, with a relation created from the given expression. Shortcut for creating a graph, then adding a relation created from the expression. |
| public Graph(int width, int height, double xStart, double yStart, double xEnd, double yEnd, double xTickInterval, double yTickInterval, String s) | Create a graph with the same parameters as the first constructor, with a relation created from the expression created by parsing the string. |
| public Graph(Relation r) | Create a graph with the given relation, and default parameters of 500x500 pixels, over a domain of -10 to 10, a range of -10 to 10, and tick marks one unit apart. |
| public Graph(Expression e) | Same as above, but creating the relation from the given expression. |
| public Graph(String s) | Same as above, but creating the relation from the given string. |
| protected void paintComponent(Graphics g) | Draws the graph, the tick marks, and any relations on the graph. |
| private void drawRelation(Relation, Graphics2D g2d) | Given a relation and a graphics2d object, draws the relation, including holes, asymptotes, and any oter restrictions on the domain. |

## Relation

A class for graphing, holds an expression and any other data needed to graph it.

### Fields

|  |  |
| --- | --- |
| Expression exp; | The expression associated with a given relation |
| Color color; | The colour with which to draw the relation. |
| Interval interval; | The interval on which the expression should be graphed. |
| char axis; | x or y, indicates which variable is the independent variable. |

### Methods

|  |  |
| --- | --- |
| public Relation(Expression exp, Interval i, char axis, Color color) | Creates a relation with the given values. |
| public Relation(Expression exp, interval i, char axis) | Creates a relation with the given values, defaults to black for colour. |
| public Relation(Expression exp, interval i) | Creates a relation with the given values, defaults to black for colour and (-infinity, infinity) for the interval. |
| public Relation(Expression exp, char axis) | Creates a relation with the given values, defaults to black for colour and 'x' for the independent variable. |
| public Relation(double d, Interval i) | Creates a horizontal line along the given interval. |
| public Relation(double d, Interval i, char axis) | Creates a straight line, parallel to the given access, at the specified value. Used for drawing tick marks. |
| public Relation(double d, char axis, Color color) | Creates a straight line, parallel to the given axis, at the specified value. Used for drawing asymptotes |
| public double eval(Double d) | Returns the value of the function at a given value |
| public double integrate() | Returns the value of the integral over the interval associated with the relation. |
| public Relation derivative() | Calculates the derivative of the function and returns a new relation with the same properties as the original, except for the expression which is now the derivative of the original expression. |
| public String toString() | Returns the string representation of the function associated with the relation for easier reading and debugging |

## Interval

A class for graphing, holds an expression and any other data needed to graph it.

### Fields

|  |  |
| --- | --- |
| double start, end | The leftmost and rightmost value of the interval. |
| boolean includeStart, includeEnd; | Whether the interval is open or closed on each side; |
| boolean infStart; | Whether or not the interval goes to negative infinity. |
| boolean infEnd; | Whether or not the interval goes to positive infinity. |

### Methods

|  |  |
| --- | --- |
| public Interval(double d) | Creates an interval of length 0, at the specified value. |
| public Interval (char inclStart, double start, double end, char inclEnd) | Creates an interval from the given start value to the given end value, inclStart is either '[' indicating the interval is closed or '(' indicating it is open, and similar follows for inclEnd with ']' and ')'. |
| public Interval(double end, char inclEnd) | Creates an interval on (-infinity, x) or (-infinity, x] where x is the given value, open or closed based on inclEnd. |
| public Interval(char inclStart, double start) | Creates an interval on (x, infinity) or [x, infinity) where x is the given value, open or closed based on inclStart. |
| public Interval() | Creates an intrval on (-infinity, infinity). |
| public Interval(String s) | Parses the string into an interval, using -inf and inf to represent negative and positive infinity. |
| public boolean isInInterval(double d) | Returns true if the value is on the interval, false if the value is not. |
| public String toString() | Returns the string representation of the interval. |

## PartialExp

A class for simplifying the object given by initial string parsing into one expression. Consists of either an operator or an expression.

### Fields

|  |  |
| --- | --- |
| Expression exp; | The expression represented by the object, if it represents an expression. |
| char op; | The operator represented by the object, if it represents an operator. |
| String type; | Either “op” for operator or “exp” for expression. |

### Methods

|  |  |
| --- | --- |
| public PartialExp(char arg) | Creates a partial expression with the given operator. |
| public PartialExp(Expression arg) | Creates a partial expression with the given expression. |

## MathTools

Contains the methods for derivatives, simplification, and input parsing.

### Methods

|  |  |
| --- | --- |
| public static Expression derivative(Expression e) | Returns the simplified derivative of an expression |
| private static Expression derivativeInternal(Expression e) | Internal method that calculates the derivative on an expression, before returning it to the derivative method for simplification. |
| private static Expression derivativeTerm(Expression e) | Called by derivativeInternal to handle special cases for derivatives such as sin, cos, and tan. |
| public static Expression parse(String s) | Parses the given string into an expression. |
| public static ArrayList<PartialExp> simplify(ArrayList<PartialExp> e) | Used in parsing strings, used for concatenating multiple expression objects into one. |
| public static Expression simplify(Expression e) | Calls simplifyInternal until no further changes are made on the expression is as simple as the method can make it. |
| private static Expression simplifyInternal(Expression e) | Looks for opportunities to simplify the structue of an expression, and if any are found returns the simplified version. |