Program takes all inputs with expressions and evaluates them to return some value. This includes plain numbers, and unary expressions. It only returns a number.

**Expression:**

* No matter what input is used, there is always at least one unary Expression for the program to evaluate.
  + This is where the composition relationship of expression being composed of one or more Operators is derived from. An Expression cannot exist without at least one Operator.
* Possible Operators (Unary and/or Binary):
  + +
  + -
  + /
  + \*
  + NULL (returns 0 unless a Number object is present)
* An Expression may include numbers, but does not require them
  + This is an example of an aggregate relationship. The Expression can have a Number, but it does not require it to remain an Expression. There can be 0 to many Numbers.

**Number:**

* Contains a floating point number
* Can only be part of an expression
  + A Number alone in an Expression with no operations being performed on it, which returns the Number itself

**Operator:**

* Has two types (child classes): Binary, Unary
  + Binary: takes two Expressions and returns the evaluated Number
  + Unary: takes one Expression (likely a Number) and returns the evaluated Number
* The program accepts +, -, /, \*, and no operator, each of which are either a unary or binary Operator depending on the present Expression(s)