Reviewer Name: Ruqian Yuan

Reviewed Name: LAKSHAY SANTOSH KUCHERIYA

Code coverage analysis:

Method Name	Code coverage	Proposed test(s) to include
Num::equals(Expr * e)	100% line coverage, 80 % branch coverage	Num equals to Add / Var
Var::equals(Expr * e)	100% line coverage, 80 % branch coverage	Var equals to Add / Mult
Add::equals(Expr * e)	100% line coverage, 80 % branch coverage	Add equals to Mult / Var
Mult::equals(Expr * e)	100% line coverage, 80 % branch coverage	Mult equals to Mult / Num / Var
Add:has_variable()	100% line coverage, 80 % branch coverage	Composite Add like Add(Add, Add), Add(Num, Var) Basically the other untested cases of the Cartesian product of { Num, Var, Add, Mult } * { Num, Var, Add, Mult }
Mult:has_variable()	100% line coverage, 80 % branch coverage	Composite Multlike Mult(Add, Add), Mult(Mult, Mult) Basically the other untested cases of the Cartesian product of { Num, Var, Add, Mult } * { Num, Var, Add, Mult }
Num / Var / Add / Mult::subst()	100% line coverage, 80 % branch coverage	Basically the other untested cases of the Cartesian product of { Num, Var, Add, Mult } * { contains element to substitute, does not contains element to substitute }

Thoughts / suggestions to improve the code or the tests:

Could use section to organize tests in the same categories like put all the Add::equals in one section to improve readability
Could use intermediate variable to store long Expression declaration to improve readability
Could map out the Cartesian products of the branches first and use that as the blueprint to write tests.