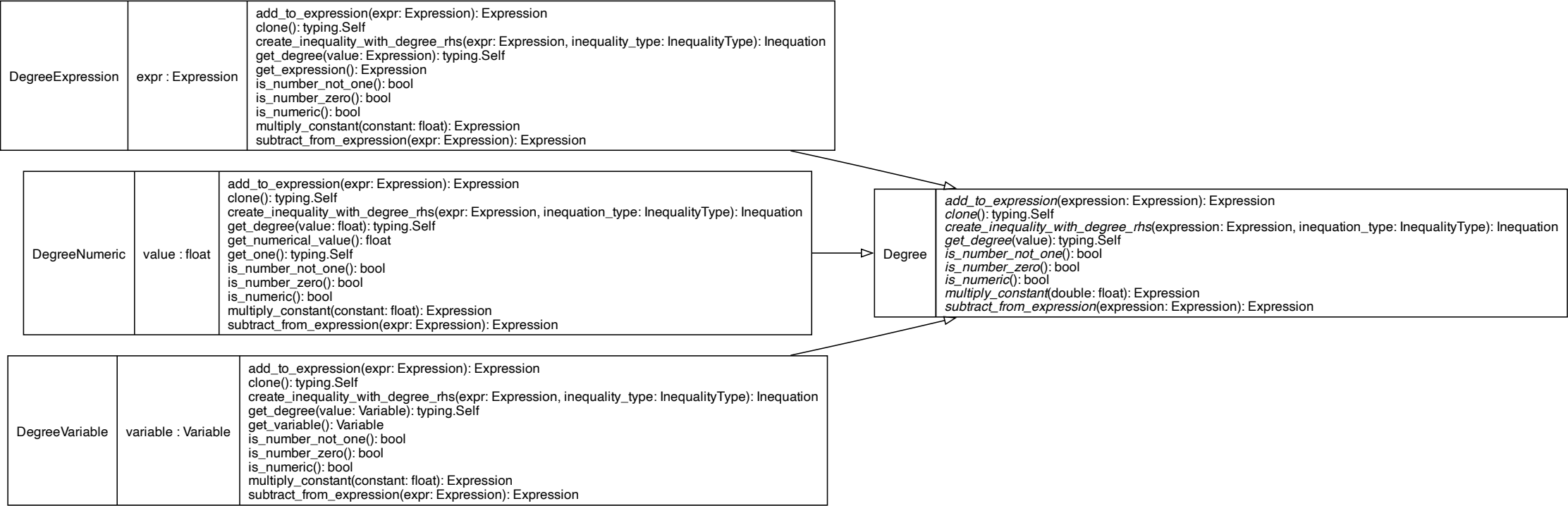
DegreeExpression
expr : Expression
add_to_expression(expr: Expression): Expression
clone(): typing.Self
create_inequality_with_degree_rhs(expr: Expression, inequality_type: InequalityType): Inequation
get_degree(value: Expression): typing.Self
get_expression(): Expression
is_number_not_one(): bool
is_number_zero(): bool
is_numeric(): bool
multiply_constant(constant: float): Expression
subtract_from_expression(expr: Expression): Expression
DegreeNumeric
value : float
add_to_expression(expr: Expression): Expression
clone(): typing.Self
create_inequality_with_degree_rhs(expr: Expression, inequation_type: InequalityType): Inequation
get_degree(value: float): typing.Self
get_numerical_value(): float
get_one(): typing.Self
is_number_not_one(): bool
is_number_zero(): bool
is_numeric(): bool
multiply_constant(constant: float): Expression
subtract_from_expression(expr: Expression): Expression
DegreeVariable
variable : Variable
add_to_expression(expr: Expression): Expression
clone(): typing.Self
create_inequality_with_degree_rhs(expr: Expression, inequality_type: InequalityType): Inequation
get_degree(value: Variable): typing.Self
get_variable(): Variable
is_number_not_one(): bool
is_number_zero(): bool
is_numeric(): bool
multiply_constant(constant: float): Expression
subtract_from_expression(expr: Expression): Expression
Degree
add_to_expression(expression: Expression): Expression
clone(): typing.Self
create_inequality_with_degree_rhs(expression: Expression, inequation_type: InequalityType): Inequation
get_degree(value): typing.Self
is_number_not_one(): bool
is_number_zero(): bool
is_numeric(): bool
multiply_constant(double: float): Expression
subtract_from_expression(expression: Expression): Expression