MSDscript

Generated by Doxygen 1.9.6

Chapter 1

MSDScript

passing arguments through command line, execute with –help, –test ...

Author

Juisheng Hung (Rason)

Date

01-16-2023

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

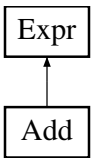
Class Documentation

5.1 Add Class Reference

Add class inherits from Expr class, representing addition for two expressions.

`#include <expr.h>`

Inheritance diagram for Add:



Public Member Functions

- Add (Expr *lhs, Expr *rhs)

  Constructor for Add object.
- bool equals (Expr *e) override

  Judge if this Add class object equals to another object.
- int interp () override

  Interpret Add object to an integer value.
- bool has_variable () override

  Judge if the Add object contains any Variable.
- Expr * subst (std::string string, Expr *e) override

  Substitute the Variable inside Add object with another Expr.
- void print (std::ostream &ostream) override

  print the expression into most basic format (with parentheses, no space)
- void pretty_print (std::ostream &ostream) override

  *print the expression into a pretty format (avoids unnecessary parentheses, with space around + / *)*
- precedence_t pretty_print_at () override

  implementation helper function of pretty_print for classifying case

Public Member Functions inherited from Expr

- virtual bool equals (Expr *e)=0

  Judge if this Expr class object equals to another object.
- virtual int interp ()=0

  Interpret Expr object to an integer value.
- virtual bool has_variable ()=0

  Judge if the Expr object contains any Variable.
- virtual Expr * subst (std::string string, Expr *e)=0

  Substitute the Variable inside Expr object with another Expr.
- virtual void print (std::ostream &ostream)=0

  print the expression into most basic format (with parentheses, no space)
- virtual void pretty_print (std::ostream &ostream)=0

  *print the expression into a pretty format (avoids unnecessary parentheses, with space around + / *)*
- virtual precedence_t pretty_print_at ()=0

  implementation helper function of pretty_print for classifying case
- std::string **to_string** ()

  converting expression to string with basic format
- std::string **to_pretty_string** ()

  converting expression to string with a pretty format

Public Attributes

- Expr * **lhs**

  the Expr object that makes up the left hand side of the Add object
- Expr * **rhs**

  the Expr object that makes up the right hand side of the Add object

5.1.1 Detailed Description

Add class inherits from Expr class, representing addition for two expressions.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Add()

```
Add::Add (
            Expr * lhs,
            Expr * rhs )
```

Constructor for Add object.

Parameters

lhs	an Expr object on the left hand side
rhs	an Expr object on the right hand side

5.1.3 Member Function Documentation

5.1.3.1 equals()

```
bool Add::equals (
            Expr * e )  [override], [virtual]
```

Judge if this Add class object equals to another object.

Parameters

e	an Expr pointer to Expr object waited to be compared

Returns

returns a boolean, true if two object equals, otherwise false

Implements Expr.

5.1.3.2 has_variable()

```
bool Add::has_variable ( )  [override], [virtual]
```

Judge if the Add object contains any Variable.

Returns

returns a boolean, true if the Expr object contains any Variable, otherwise false

Implements Expr.

5.1.3.3 interp()

```
int Add::interp ( )  [override], [virtual]
```

Interpret Add object to an integer value.

Returns

returns the actual integer value (lhs + rhs) of the Add, if it contains Variable, throw an exception

Implements Expr.

5.1.3.4 pretty_print()

```
void Add::pretty_print (
            std::ostream & ostream )  [override], [virtual]
```

print the expression into a pretty format (avoids unnecessary parentheses, with space around + / $*$)

Parameters

ostream	deliver string through this output stream

Implements Expr.

5.1.3.5 pretty_print_at()

```
precedence_t Add::pretty_print_at ( )  [override], [virtual]
```

implementation helper function of pretty_print for classifying case

Returns

precedence_t type enum

Implements Expr.

5.1.3.6 print()

```
void Add::print (
            std::ostream & ostream )  [override], [virtual]
```

print the expression into most basic format (with parentheses, no space)

Parameters

ostream	deliver string through this output stream

Implements Expr.

5.1.3.7 subst()

```
Expr * Add::subst (
            std::string string,
            Expr * e )  [override], [virtual]
```

Substitute the Variable inside Add object with another Expr.

Parameters

string	first argument, a target string that is waited to be substituted
e	second argument, an Expr pointer to object that is going to substitute the Variable inside expression

Returns

returns the new Expr pointer to object after substitution, return the original object if string Variable not found

Implements Expr.

The documentation for this class was generated from the following files:

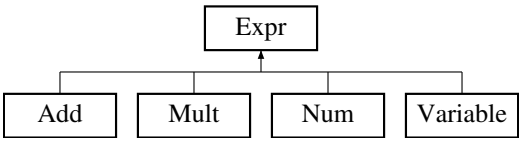
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.h
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.cpp

5.2 Expr Class Reference

Abstract expression class
(pure abstract class)

```
#include <expr.h>
```

Inheritance diagram for Expr:



Public Member Functions

- virtual bool equals (Expr *e)=0

  Judge if this Expr class object equals to another object.
- virtual int interp ()=0

  Interpret Expr object to an integer value.
- virtual bool has_variable ()=0

  Judge if the Expr object contains any Variable.
- virtual Expr * subst (std::string string, Expr *e)=0

  Substitute the Variable inside Expr object with another Expr.
- virtual void print (std::ostream &ostream)=0

  print the expression into most basic format (with parentheses, no space)
- virtual void pretty_print (std::ostream &ostream)=0

  *print the expression into a pretty format (avoids unnecessary parentheses, with space around + / *)*
- virtual precedence_t pretty_print_at ()=0

  implementation helper function of pretty_print for classifying case
- std::string **to_string** ()

  converting expression to string with basic format
- std::string **to_pretty_string** ()

  converting expression to string with a pretty format

5.2.1 Detailed Description

Abstract expression class
(pure abstract class)

5.2.2 Member Function Documentation

5.2.2.1 equals()

```
virtual bool Expr::equals (
            Expr * e )  [pure virtual]
```

Judge if this Expr class object equals to another object.

Parameters

e	an Expr pointer to object waited to be compared

Returns

returns a boolean, true if two object equals, otherwise false

Implemented in Num, Variable, Add, and Mult.

5.2.2.2 has_variable()

```
virtual bool Expr::has_variable ( )  [pure virtual]
```

Judge if the Expr object contains any Variable.

Returns

returns a boolean, true if the Expr object contains any Variable, otherwise false

Implemented in Num, Variable, Add, and Mult.

5.2.2.3 interp()

```
virtual int Expr::interp ( )  [pure virtual]
```

Interpret Expr object to an integer value.

Returns

returns the actual integer value of the Expr, if it contains Variable, throw an exception

Implemented in Num, Variable, Add, and Mult.

5.2.2.4 pretty_print()

```
virtual void Expr::pretty_print (
            std::ostream & ostream )  [pure virtual]
```

print the expression into a pretty format (avoids unnecessary parentheses, with space around + / ∗)

Parameters

ostream	deliver string through this output stream

Implemented in Num, Variable, Add, and Mult.

5.2.2.5 pretty_print_at()

```
virtual precedence_t Expr::pretty_print_at ( )  [pure virtual]
```

implementation helper function of pretty_print for classifying case

Returns

precedence_t type enum

Implemented in Num, Variable, Add, and Mult.

5.2.2.6 print()

```
virtual void Expr::print (
            std::ostream & ostream )  [pure virtual]
```

print the expression into most basic format (with parentheses, no space)

Parameters

ostream	deliver string through this output stream

Implemented in Num, Variable, Add, and Mult.

5.2.2.7 subst()

```
virtual Expr * Expr::subst (
            std::string string,
            Expr * e )  [pure virtual]
```

Substitute the Variable inside Expr object with another Expr.

Parameters

string	first argument, a target string that is waited to be substituted
e	second argument, an Expr pointer to object that is going to substitute the Variable inside expression

Returns

returns the new Expr pointer to object after substitution, return the original object if string Variable not found

Implemented in Num, Variable, Add, and Mult.

The documentation for this class was generated from the following files:

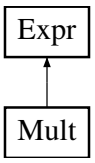
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.h
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.cpp

5.3 Mult Class Reference

Mult class inherits from Expr class, representing multiplication for two expressions.

`#include <expr.h>`

Inheritance diagram for Mult:



Public Member Functions

- Mult (Expr *lhs, Expr *rhs)

  Constructor for Mult object.
- bool equals (Expr *e) override

  Judge if this Mult class object equals to another object.
- int interp () override

  Interpret Mult object to an integer value.
- bool has_variable () override

  Judge if the Mult object contains any Variable.
- Expr * subst (std::string string, Expr *e) override

  Substitute the Variable inside Mult object with another Expr.
- void print (std::ostream &ostream) override

  print the expression into most basic format (with parentheses, no space)
- void pretty_print (std::ostream &ostream) override

  *print the expression into a pretty format (avoids unnecessary parentheses, with space around + / *)*
- precedence_t pretty_print_at () override

  implementation helper function of pretty_print for classifying case

Public Member Functions inherited from Expr

- virtual bool equals (Expr ∗e)=0

  Judge if this Expr class object equals to another object.
- virtual int interp ()=0

  Interpret Expr object to an integer value.
- virtual bool has_variable ()=0

  Judge if the Expr object contains any Variable.
- virtual Expr ∗ subst (std::string string, Expr ∗e)=0

  Substitute the Variable inside Expr object with another Expr.
- virtual void print (std::ostream &ostream)=0

  print the expression into most basic format (with parentheses, no space)
- virtual void pretty_print (std::ostream &ostream)=0

  print the expression into a pretty format (avoids unnecessary parentheses, with space around + / ∗)
- virtual precedence_t pretty_print_at ()=0

  implementation helper function of pretty_print for classifying case
- std::string **to_string** ()

  converting expression to string with basic format
- std::string **to_pretty_string** ()

  converting expression to string with a pretty format

Public Attributes

- Expr ∗ **lhs**

  the Expr object that makes up the left hand side of the Mult object
- Expr ∗ **rhs**

  the Expr object that makes up the right hand side of the Mult object

5.3.1 Detailed Description

Mult class inherits from Expr class, representing multiplication for two expressions.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 Mult()

```
Mult::Mult (
            Expr * lhs,
            Expr * rhs )
```

Constructor for Mult object.

Parameters

lhs	an Expr object on the left hand side
rhs	an Expr object on the right hand side

5.3.3 Member Function Documentation

5.3.3.1 equals()

```
bool Mult::equals (
            Expr * e )  [override], [virtual]
```

Judge if this Mult class object equals to another object.

Parameters

e	an Expr pointer to Expr object waited to be compared

Returns

returns a boolean, true if two object equals, otherwise false

Implements Expr.

5.3.3.2 has_variable()

```
bool Mult::has_variable ( )  [override], [virtual]
```

Judge if the Mult object contains any Variable.

Returns

returns a boolean, true if the Expr object contains any Variable, otherwise false

Implements Expr.

5.3.3.3 interp()

```
int Mult::interp ( )  [override], [virtual]
```

Interpret Mult object to an integer value.

Returns

returns the actual integer value (lhs $*$ rhs) of the Mult, if it contains Variable, throw an exception

Implements Expr.

5.3.3.4 pretty_print()

```
void Mult::pretty_print (
            std::ostream & ostream )  [override], [virtual]
```

print the expression into a pretty format (avoids unnecessary parentheses, with space around + / $*$)

Parameters

ostream	deliver string through this output stream

Implements Expr.

5.3.3.5 pretty_print_at()

```
precedence_t Mult::pretty_print_at ( )  [override], [virtual]
```

implementation helper function of pretty_print for classifying case

Returns

precedence_t type enum

Implements Expr.

5.3.3.6 print()

```
void Mult::print (
            std::ostream & ostream )  [override], [virtual]
```

print the expression into most basic format (with parentheses, no space)

Parameters

ostream	deliver string through this output stream

Implements Expr.

5.3.3.7 subst()

```
Expr * Mult::subst (
            std::string string,
            Expr * e )  [override], [virtual]
```

Substitute the Variable inside Mult object with another Expr.

Parameters

string	first argument, a target string that is waited to be substituted
e	second argument, an Expr pointer to object that is going to substitute the Variable inside expression

Returns

returns the new Expr pointer to object after substitution, return the original object if string Variable not found

Implements Expr.

The documentation for this class was generated from the following files:

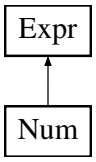
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.h
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.cpp

5.4 Num Class Reference

Num class inherits from Expr class, representing pure number.

`#include <expr.h>`

Inheritance diagram for Num:



Public Member Functions

- Num (int val)

  Constructor for Num object.
- bool equals (Expr *e) override

  Judge if this Num class object equals to another object.
- int interp () override

  Interpret Num object to an integer value.
- bool has_variable () override

  Judge if the Num object contains any Variable.
- Expr * subst (std::string string, Expr *e) override

  Substitute the Variable inside Num object with another Expr.
- void print (std::ostream &ostream) override

  print the expression into most basic format (with parentheses, no space)
- void pretty_print (std::ostream &ostream) override

  *print the expression into a pretty format (avoids unnecessary parentheses, with space around + / *)*
- precedence_t pretty_print_at () override

  implementation helper function of pretty_print for classifying case

Public Member Functions inherited from Expr

- virtual bool equals (Expr *e)=0

  Judge if this Expr class object equals to another object.
- virtual int interp ()=0

  Interpret Expr object to an integer value.
- virtual bool has_variable ()=0

  Judge if the Expr object contains any Variable.
- virtual Expr * subst (std::string string, Expr *e)=0

  Substitute the Variable inside Expr object with another Expr.
- virtual void print (std::ostream &ostream)=0

  print the expression into most basic format (with parentheses, no space)
- virtual void pretty_print (std::ostream &ostream)=0

  *print the expression into a pretty format (avoids unnecessary parentheses, with space around + / *)*
- virtual precedence_t pretty_print_at ()=0

  implementation helper function of pretty_print for classifying case
- std::string **to_string** ()

  converting expression to string with basic format
- std::string **to_pretty_string** ()

  converting expression to string with a pretty format

Public Attributes

- int **val**

  the integer value of the Num object

5.4.1 Detailed Description

Num class inherits from Expr class, representing pure number.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 Num()

```
Num::Num (
            int val )  [explicit]
```

Constructor for Num object.

Parameters

val	integer value of Num

5.4.3 Member Function Documentation

5.4.3.1 equals()

```
bool Num::equals (
            Expr * e )  [override], [virtual]
```

Judge if this Num class object equals to another object.

Parameters

e	an Expr pointer to Expr object waited to be compared

Returns

returns a boolean, true if two object equals, otherwise false

Implements Expr.

5.4.3.2 has_variable()

```
bool Num::has_variable ( )  [override], [virtual]
```

Judge if the Num object contains any Variable.

Returns

returns a boolean, always return false

Implements Expr.

5.4.3.3 interp()

```
int Num::interp ( )  [override], [virtual]
```

Interpret Num object to an integer value.

Returns

returns the actual integer value of the Num

Implements Expr.

5.4.3.4 pretty_print()

```
void Num::pretty_print (
            std::ostream & ostream )  [override], [virtual]
```

print the expression into a pretty format (avoids unnecessary parentheses, with space around + / $*$)

Parameters

ostream	deliver string through this output stream

Implements Expr.

5.4.3.5 pretty_print_at()

```
precedence_t Num::pretty_print_at ( )  [override], [virtual]
```

implementation helper function of pretty_print for classifying case

Returns

precedence_t type enum

Implements Expr.

5.4.3.6 print()

```
void Num::print (
            std::ostream & ostream )  [override], [virtual]
```

print the expression into most basic format (with parentheses, no space)

Parameters

ostream	deliver string through this output stream

Implements Expr.

5.4.3.7 subst()

```
Expr * Num::subst (
            std::string string,
            Expr * e )  [override], [virtual]
```

Substitute the Variable inside Num object with another Expr.

Parameters

string	first argument, a target string that is waited to be substituted
e	second argument, an Expr pointer to object that is going to substitute the Variable inside expression

Returns

returns this object, since there is no Variable in Num object

Implements Expr.

The documentation for this class was generated from the following files:

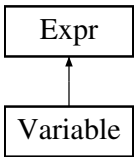
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.h
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.cpp

5.5 Variable Class Reference

Variable class inherits from Expr class, representing pure variable.

`#include <expr.h>`

Inheritance diagram for Variable:



Public Member Functions

- Variable (std::string varName)

  Constructor for Variable object.
- bool equals (Expr ∗e) override

  Judge if this Variable class object equals to another object, overrides function in superclass.
- int interp () override

  Interpret Variable object to an integer value.
- bool has_variable () override

  Judge if the Variable object contains any Variable.
- Expr ∗ subst (std::string string, Expr ∗e) override

  Substitute the Variable object with another Expr.
- void print (std::ostream &ostream) override

  print the expression into most basic format (with parentheses, no space)
- void pretty_print (std::ostream &ostream) override

  print the expression into a pretty format (avoids unnecessary parentheses, with space around + / ∗)
- precedence_t pretty_print_at () override

  implementation helper function of pretty_print for classifying case

Public Member Functions inherited from Expr

- virtual bool equals (Expr *e)=0

  Judge if this Expr class object equals to another object.
- virtual int interp ()=0

  Interpret Expr object to an integer value.
- virtual bool has_variable ()=0

  Judge if the Expr object contains any Variable.
- virtual Expr * subst (std::string string, Expr *e)=0

  Substitute the Variable inside Expr object with another Expr.
- virtual void print (std::ostream &ostream)=0

  print the expression into most basic format (with parentheses, no space)
- virtual void pretty_print (std::ostream &ostream)=0

  *print the expression into a pretty format (avoids unnecessary parentheses, with space around + / *)*
- virtual precedence_t pretty_print_at ()=0

  implementation helper function of pretty_print for classifying case
- std::string **to_string** ()

  converting expression to string with basic format
- std::string **to_pretty_string** ()

  converting expression to string with a pretty format

Public Attributes

- std::string **name**

  the string name that makes up the Variable object

5.5.1 Detailed Description

Variable class inherits from Expr class, representing pure variable.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Variable()

```
Variable::Variable (
            std::string varName )  [explicit]
```

Constructor for Variable object.

Parameters

varName	a string that can be seen as the label of the Variable

5.5.3 Member Function Documentation

5.5.3.1 equals()

```
bool Variable::equals (
            Expr * e )  [override], [virtual]
```

Judge if this Variable class object equals to another object, overrides function in superclass.

Parameters

e	an Expr pointer to Expr object waited to be compared

Returns

returns a boolean, true if two object equals, otherwise false

Implements Expr.

5.5.3.2 has_variable()

```
bool Variable::has_variable ( )  [override], [virtual]
```

Judge if the Variable object contains any Variable.

Returns

returns a boolean, always return true

Implements Expr.

5.5.3.3 interp()

```
int Variable::interp ( )  [override], [virtual]
```

Interpret Variable object to an integer value.

Returns

A Variable doesn't have specific integer value, throw an exception

Implements Expr.

5.5.3.4 pretty_print()

```
void Variable::pretty_print (
            std::ostream & ostream )  [override], [virtual]
```

print the expression into a pretty format (avoids unnecessary parentheses, with space around + / $*$)

Parameters

ostream	deliver string through this output stream

Implements Expr.

5.5.3.5 pretty_print_at()

```
precedence_t Variable::pretty_print_at ( )  [override], [virtual]
```

implementation helper function of pretty_print for classifying case

Returns

precedence_t type enum

Implements Expr.

5.5.3.6 print()

```
void Variable::print (
            std::ostream & ostream )  [override], [virtual]
```

print the expression into most basic format (with parentheses, no space)

Parameters

ostream	deliver string through this output stream

Implements Expr.

5.5.3.7 subst()

```
Expr * Variable::subst (
            std::string string,
            Expr * e )  [override], [virtual]
```

Substitute the Variable object with another Expr.

Parameters

string	first argument, a target string that is waited to be substituted
e	second argument, an Expr pointer to object that is going to substitute the Variable inside expression

Returns

returns the new Expr pointer to object after substitution, return the original object if string Variable not found

Implements Expr.

The documentation for this class was generated from the following files:

- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.h
- /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.cpp

Chapter 6

File Documentation

6.1 /Users/rasonhung/Study/MSD/CS6015/MSDScript/cmdline.h File Reference

actual function that executes command line script

```
#include <iostream>
#include <string>
```

Functions

- void use_arguments (int argc, const char ∗argv[ ])

  Take arguments from command line as input, execute corresponding output as required.

6.1.1 Detailed Description

actual function that executes command line script

6.1.2 Function Documentation

6.1.2.1 use_arguments()

```
void use_arguments (
            int argc,
            const char * argv[] )
```

Take arguments from command line as input, execute corresponding output as required.

Parameters

argc	first argument, the integer number of arguments passed into
argv	second argument, the pointer to the array of characters that is passed into as parameter

Returns

returns void
"--help": if it is the next argument after program name, print out help message, and do not examine other arguments
"--test": if it is the only argument after program name, then print out test result, otherwise, will be treated as invalid argument input any other strings as input: invalid argument, exit the program with 1

6.2 /Users/rasonhung/Study/MSD/CS6015/MSDScript/cmdline.h

Go to the documentation of this file.

```
00001 //
00002 //  cmdline.h
00003 //  CommandLine
00004 //
00005 //  Created by Rason Hung on 1/16/23.
00006 //
00007
00014 #pragma include once
00015 #include <iostream>
00016 #include <string>
00017
00025 void use_arguments(int argc, const char *argv[]);
```

6.3 /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.h File Reference

expression class

```
#include <cstdio>
#include <string>
#include <sstream>
#include <stdexcept>
#include <utility>
```

Classes

- class Expr

  Abstract expression class
  (pure abstract class)
- class Num

  Num class inherits from Expr class, representing pure number.
- class Variable

  Variable class inherits from Expr class, representing pure variable.
- class Add

  Add class inherits from Expr class, representing addition for two expressions.
- class Mult

  Mult class inherits from Expr class, representing multiplication for two expressions.

Enumerations

- enum **precedence_t** { **prec_none** , **prec_add** , **prec_mult** }

6.3.1 Detailed Description

expression class

Contains the blueprint of the superclass - Expr, with its subclass - Num, Add, Mult, Variable

6.4 /Users/rasonhung/Study/MSD/CS6015/MSDScript/expr.h

Go to the documentation of this file.

```
00001 //
00002 //  expr.h
00003 //  ExpressionClasses
00004 //
00005 //  Created by Rason Hung on 1/22/23.
00006 //
00007
00015 #pragma include once
00016 #include <cstdio>
00017 #include <string>
00018 #include <sstream>
00019 #include <stdexcept>
00020 #include <utility>
00021
00022 typedef enum {
00023     prec_none,  // = 0
00024     prec_add,   // = 1
00025     prec_mult,  // = 2
00026 } precedence_t;
00027
00028
00032 class Expr {
00033 public:
00039     virtual bool equals(Expr *e) = 0;
00040
00045     virtual int interp() = 0;
00046
00051     virtual bool has_variable() = 0;
00052
00059     virtual Expr* subst(std::string string, Expr* e)=0;
00060
00061     //TODO: do we need to handle with negative expression?
00066     virtual void print(std::ostream &ostream) = 0;
00067
00072     virtual void pretty_print(std::ostream &ostream) = 0;
00073
00078     virtual precedence_t pretty_print_at() = 0;
00079
00083     std::string to_string();
00084
00088     std::string to_pretty_string(); // if not required - only for test use
00089 };
00090
00091
00092
00093
00096 class Num : public Expr {
00097 public:
00098     int val;
00099
00104     explicit Num(int val);
00105
00111     bool equals(Expr *e) override;
00112
00117     int interp() override;
00118
00123     bool has_variable() override;
00124
00131     Expr* subst(std::string string, Expr* e) override;
00132
00133     void print(std::ostream &ostream) override;
```

```
00134
00135     void pretty_print(std::ostream &ostream) override;
00136
00137     precedence_t pretty_print_at() override;
00138 };
00139
00140
00141
00144 class Variable : public Expr {
00145 public:
00146     std::string name;
00147
00152     explicit Variable(std::string varName);
00153
00159     bool equals(Expr *e) override;
00160
00165     int interp() override;
00166
00171     bool has_variable() override;
00172
00179     Expr* subst(std::string string, Expr* e) override;
00180
00181     void print(std::ostream &ostream) override;
00182
00183     void pretty_print(std::ostream &ostream) override;
00184
00185     precedence_t pretty_print_at() override;
00186 };
00187
00188
00189
00190
00193 class Add : public Expr {
00194 public:
00195     Expr *lhs;
00196     Expr *rhs;
00197
00203     Add(Expr *lhs, Expr *rhs);
00204
00210     bool equals(Expr *e) override;
00211
00216     int interp() override;
00217
00222     bool has_variable() override;
00223
00230     Expr* subst(std::string string, Expr* e) override;
00231
00232
00233     void print(std::ostream &ostream) override;
00234
00235     void pretty_print(std::ostream &ostream) override;
00236
00237     precedence_t pretty_print_at() override;
00238 };
00239
00240
00241
00242
00245 class Mult : public Expr {
00246 public:
00247     Expr *lhs;
00248     Expr *rhs;
00249
00255     Mult(Expr *lhs, Expr *rhs);
00256
00262     bool equals(Expr *e) override;
00263
00268     int interp() override;
00269
00274     bool has_variable() override;
00275
00282     Expr* subst(std::string string, Expr* e) override;
00283
00284     void print(std::ostream &ostream) override;
00285
00286     void pretty_print(std::ostream &ostream) override;
00287
00288     precedence_t pretty_print_at() override;
00289 };
00290
00291
```

Index