

MSDscript

Generated by Doxygen 1.9.6

1 MSDScript	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Class Documentation	9
5.1 Add Class Reference	9
5.1.1 Detailed Description	10
5.1.2 Member Function Documentation	10
5.1.2.1 equals()	10
5.1.2.2 has_variable()	11
5.1.2.3 interp()	11
5.1.2.4 pretty_print()	11
5.1.2.5 print()	11
5.1.2.6 subst()	12
5.2 Expr Class Reference	12
5.2.1 Detailed Description	13
5.2.2 Member Function Documentation	13
5.2.2.1 equals()	13
5.2.2.2 has_variable()	13
5.2.2.3 interp()	13
5.2.2.4 pretty_print()	14
5.2.2.5 print()	14
5.2.2.6 subst()	14
5.3 Mult Class Reference	14
5.3.1 Detailed Description	15
5.3.2 Member Function Documentation	15
5.3.2.1 equals()	15
5.3.2.2 has_variable()	16
5.3.2.3 interp()	16
5.3.2.4 pretty_print()	16
5.3.2.5 print()	17
5.3.2.6 subst()	17
5.4 Num Class Reference	18
5.4.1 Detailed Description	19
5.4.2 Member Function Documentation	19
5.4.2.1 equals()	19
5.4.2.2 has_variable()	19

5.4.2.3 interp()	19
5.4.2.4 pretty_print()	20
5.4.2.5 print()	20
5.4.2.6 subst()	20
5.5 Var Class Reference	21
5.5.1 Detailed Description	22
5.5.2 Member Function Documentation	22
5.5.2.1 equals()	22
5.5.2.2 has_variable()	22
5.5.2.3 interp()	22
5.5.2.4 pretty_print()	23
5.5.2.5 print()	23
5.5.2.6 subst()	23
6 File Documentation	25
6.1 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/cmdline.cpp	
File Reference	25
6.1.1 Detailed Description	25
6.1.2 Function Documentation	25
6.1.2.1 use_arguments()	25
6.2 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/cmdline.hpp	
File Reference	26
6.2.1 Detailed Description	26
6.2.2 Function Documentation	26
6.2.2.1 use_arguments()	26
6.3 cmdline.hpp	27
6.4 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.cpp	
File Reference	27
6.4.1 Detailed Description	27
6.5 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp	
File Reference	27
6.5.1 Detailed Description	28
6.6 expr.hpp	28
6.7 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/main.cpp	
File Reference	30
6.7.1 Detailed Description	30
6.8 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/tests.cpp	
File Reference	30
6.8.1 Detailed Description	30
Index	31

Chapter 1

MSDScript

Author

Lakshay Santosh Kucheriya

Date

Feburary 7, 2023

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Expr	12
Add	9
Mult	14
Num	18
Var	21

Chapter 3

Class Index

3.1 Class List

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

Add	Derived Class from Expr for representing a expression involving addition	9
Expr	Base Class for representing a expression	12
Mult	Derived Class from Expr for representing a expression involving addition	14
Num	Derived Class from Expr for representing a number	18
Var	Derived Class from Expr for representing a expression involving Variable	21

Chapter 4

File Index

4.1 File List

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

/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/ cmdline.cpp	
Contains the code for identifying the argument given ahead of running the executable on the command line is valid and if it is valid executes the command	25
/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/ cmdline.hpp	
Header file for cmdline.cpp file	26
/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/ expr.cpp	
Contains expression class definition	27
/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/ expr.hpp	
Header file for expr.cpp file	27
/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/ main.cpp	
Entry point for the project	30
/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/ tests.cpp	
All the tests forl the methods present in this project	30

Chapter 5

Class Documentation

5.1 Add Class Reference

Derived Class from [Expr](#) for representing a expression involving addition.

```
#include <expr.hpp>
```

Inheritance diagram for Add:



Public Member Functions

- **Add** ([Expr](#) *lhs, [Expr](#) *rhs)
Constructor.
- bool [equals](#) ([Expr](#) *e)
Checks for the equality between the left hand side and the right hand side.
- int [interp](#) ()
This function interprets the value of the expression.
- bool [has_variable](#) ()
This function determines if the the expression consists of a variable or not.
- [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)
This function substitutes the expression with the combination of sub-expressions if possible.
- void [print](#) (std::ostream &out)
Prints the expression.
- void [pretty_print](#) (std::ostream &out)
Prints the expression with more clarity.

Public Member Functions inherited from [Expr](#)

- virtual bool [equals](#) ([Expr](#) *e)=0
- virtual int [interp](#) ()=0
- virtual bool [has_variable](#) ()=0
- virtual [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)=0
- virtual void [print](#) (std::ostream &out)=0
- virtual void [pretty_print](#) (std::ostream &out)=0
- std::string [to_string](#) ()
- std::string [to_pretty_string](#) ()

Public Attributes

- [Expr](#) * **lhs**
Left hand side Expression.
- [Expr](#) * **rhs**
Right hand side Expression.

5.1.1 Detailed Description

Derived Class from [Expr](#) for representing a expression involving addition.

5.1.2 Member Function Documentation

5.1.2.1 [equals\(\)](#)

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

Checks for the equality between the left hand side and the right hand side.

Parameters

<i>e</i>	Expression
----------	------------

Returns

boolean value of LHS = RHS

Implements [Expr](#).

5.1.2.2 has_variable()

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

This function determines if the the expression consists of a variable or not.

Returns

boolean value for the expression has a variable or not

Implements [Expr](#).

5.1.2.3 interp()

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

This function interprets the value of the expression.

Returns

integer value of the computed expression

Implements [Expr](#).

5.1.2.4 pretty_print()

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

Prints the expression with more clarity.

Parameters

out	out	output stream
-----	-----	---------------

Implements [Expr](#).

5.1.2.5 print()

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

Prints the expression.

Parameters

out	out	output stream
-----	-----	---------------

Implements [Expr](#).

5.1.2.6 subst()

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

This function substitutes the expression with the combination of sub-expressions if possible.

Parameters

s	first argument, string which can be replaced as a part of the expression
e	second argument, Expression

Returns

Expression which is modified if combination of sub expressions was possible

Implements [Expr](#).

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

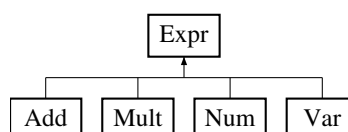
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/[expr.hpp](#)
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/[expr.cpp](#)

5.2 Expr Class Reference

Base Class for representing a expression.

```
#include <expr.hpp>
```

Inheritance diagram for Expr:



Public Member Functions

- virtual bool [equals](#) ([Expr](#) *e)=0
- virtual int [interp](#) ()=0
- virtual bool [has_variable](#) ()=0
- virtual [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)=0
- virtual void [print](#) (std::ostream &out)=0
- virtual void [pretty_print](#) (std::ostream &out)=0
- std::string [to_string](#) ()
- std::string [to_pretty_string](#) ()

5.2.1 Detailed Description

Base Class for representing a expression.

5.2.2 Member Function Documentation

5.2.2.1 [equals\(\)](#)

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

Implemented in [Num](#), [Add](#), [Mult](#), and [Var](#).

5.2.2.2 [has_variable\(\)](#)

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

Implemented in [Num](#), [Add](#), [Mult](#), and [Var](#).

5.2.2.3 [interp\(\)](#)

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

Implemented in [Num](#), [Add](#), [Mult](#), and [Var](#).

5.2.2.4 pretty_print()

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

Implemented in [Num](#), [Add](#), [Mult](#), and [Var](#).

5.2.2.5 print()

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

Implemented in [Num](#), [Add](#), [Mult](#), and [Var](#).

5.2.2.6 subst()

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

Implemented in [Num](#), [Add](#), [Mult](#), and [Var](#).

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

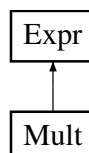
- [/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp](#)

5.3 Mult Class Reference

Derived Class from [Expr](#) for representing a expression involving addition.

```
#include <expr.hpp>
```

Inheritance diagram for Mult:



Public Member Functions

- **Mult** ([Expr](#) *lhs, [Expr](#) *rhs)
Constructor.
- bool [equals](#) ([Expr](#) *e)
Checks for the equality between the left hand side and the right hand side.
- int [interp](#) ()
This function interprets the value of the expression.
- bool [has_variable](#) ()
This function determines if the the expression consists of a variable or not.
- [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)
This function substitutes the expression with the combination of sub-expressions if possible.
- void [print](#) (std::ostream &out)
Prints the expression.
- void [pretty_print](#) (std::ostream &out)
Prints the expression with more clarity.

Public Member Functions inherited from [Expr](#)

- virtual bool [equals](#) ([Expr](#) *e)=0
- virtual int [interp](#) ()=0
- virtual bool [has_variable](#) ()=0
- virtual [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)=0
- virtual void [print](#) (std::ostream &out)=0
- virtual void [pretty_print](#) (std::ostream &out)=0
- std::string [to_string](#) ()
- std::string [to_pretty_string](#) ()

Public Attributes

- [Expr](#) * lhs
Left hand side Expression.
- [Expr](#) * rhs
Right hand side Expression.

5.3.1 Detailed Description

Derived Class from [Expr](#) for representing a expression involving addition.

5.3.2 Member Function Documentation

5.3.2.1 [equals\(\)](#)

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

Checks for the equality between the left hand side and the right hand side.

Parameters

<i>e</i>	Expression
----------	------------

Returns

boolean value of LHS = RHS

Implements [Expr](#).

5.3.2.2 has_variable()

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

This function determines if the the expression consists of a variable or not.

Returns

boolean value for the expression has a variable or not

Implements [Expr](#).

5.3.2.3 interp()

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

This function interprets the value of the expression.

Returns

integer value of the computed expression

Implements [Expr](#).

5.3.2.4 pretty_print()

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

Prints the expression with more clarity.

Parameters

out	<i>out</i>	output stream
-----	------------	---------------

Implements [Expr](#).

5.3.2.5 print()

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

Prints the expression.

Parameters

out	<i>out</i>	output stream
-----	------------	---------------

Implements [Expr](#).

5.3.2.6 subst()

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

This function substitutes the expression with the combination of sub-expressions if possible.

Parameters

<i>s</i>	first argument, string which can be replaced as a part of the expression
<i>e</i>	second argument, Expressiont

Returns

Expression which is modified if combination of sub expressions was possible

Implements [Expr](#).

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

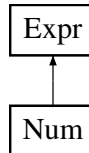
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/[expr.hpp](#)
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/[expr.cpp](#)

5.4 Num Class Reference

Derived Class from [Expr](#) for representing a number.

```
#include <expr.hpp>
```

Inheritance diagram for Num:



Public Member Functions

- **Num** (int [val](#))
Constructor.
- bool [equals](#) ([Expr](#) *e)
Checks for the equality between the left hand side and the right hand side.
- int [interp](#) ()
This function interprets the value of the integer.
- bool [has_variable](#) ()
This function determines if the the expression consists of a variable or not.
- [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)
This function substitutes the expression with the combination of sub-expressions if possible.
- void [print](#) (std::ostream &out)
Prints the expression.
- void [pretty_print](#) (std::ostream &out)
Prints the expression with more clarity.

Public Member Functions inherited from [Expr](#)

- virtual bool [equals](#) ([Expr](#) *e)=0
- virtual int [interp](#) ()=0
- virtual bool [has_variable](#) ()=0
- virtual [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)=0
- virtual void [print](#) (std::ostream &out)=0
- virtual void [pretty_print](#) (std::ostream &out)=0
- std::string [to_string](#) ()
- std::string [to_pretty_string](#) ()

Public Attributes

- int **val**
Value of the number.

5.4.1 Detailed Description

Derived Class from [Expr](#) for representing a number.

5.4.2 Member Function Documentation

5.4.2.1 equals()

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

Checks for the equality between the left hand side and the right hand side.

Parameters

<i>e</i>	Expression
----------	------------

Returns

boolean value of LHS = RHS

Implements [Expr](#).

5.4.2.2 has_variable()

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

This function determines if the the expression consists of a variable or not.

Returns

boolean value for the number has a variable or not

Implements [Expr](#).

5.4.2.3 interp()

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

This function interprets the value of the integer.

Returns

integer value of the number

Implements [Expr](#).

5.4.2.4 pretty_print()

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

Prints the expression with more clarity.

Parameters

out	<i>out</i>	output stream
-----	------------	---------------

Implements [Expr](#).

5.4.2.5 print()

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

Prints the expression.

Parameters

out	<i>out</i>	output stream
-----	------------	---------------

Implements [Expr](#).

5.4.2.6 subst()

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

This function substitutes the expression with the combination of sub-expressions if possible.

Parameters

s	first argument, string which can be replaced as a part of the expression
e	second argument, Expression

Returns

Expression which is modified if combination of sub expressions was possible

Implements [Expr](#).

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

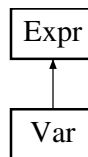
- [/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp](#)
- [/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.cpp](#)

5.5 Var Class Reference

Derived Class from [Expr](#) for representing a expression involving Variable.

```
#include <expr.hpp>
```

Inheritance diagram for Var:



Public Member Functions

- **Var** (std::string [name](#))
Constructor.
- bool [equals](#) ([Expr](#) *e)
Checks for the equality between the left hand side and the right hand side.
- int [interp](#) ()
This function interprets the value of the Variable.
- bool [has_variable](#) ()
This function determines if the the expression consists of a variable or not.
- [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)
This function substitutes the expression with the combination of sub-expressions if possible.
- void [print](#) (std::ostream &out)
Prints the expression.
- void [pretty_print](#) (std::ostream &out)
Prints the expression with more clarity.

Public Member Functions inherited from [Expr](#)

- virtual bool [equals](#) ([Expr](#) *e)=0
- virtual int [interp](#) ()=0
- virtual bool [has_variable](#) ()=0
- virtual [Expr](#) * [subst](#) (std::string s, [Expr](#) *e)=0
- virtual void [print](#) (std::ostream &out)=0
- virtual void [pretty_print](#) (std::ostream &out)=0
- std::string [to_string](#) ()
- std::string [to_pretty_string](#) ()

Public Attributes

- std::string **name**
name of the variable

5.5.1 Detailed Description

Derived Class from [Expr](#) for representing a expression involving Variable.

5.5.2 Member Function Documentation

5.5.2.1 equals()

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

Checks for the equality between the left hand side and the right hand side.

Parameters

<i>e</i>	Expression
----------	------------

Returns

boolean value of LHS = RHS

Implements [Expr](#).

5.5.2.2 has_variable()

```
bool Var::has_variable ( ) [virtual]
```

This function determines if the the expression consists of a variable or not.

Returns

boolean value for the variable is a variable or not

Implements [Expr](#).

5.5.2.3 interp()

```
int Var::interp ( ) [virtual]
```

This function interprets the value of the Variable.

Returns

error message as value of the variable cannot be iterpreted

Implements [Expr](#).

5.5.2.4 pretty_print()

```
void Var::pretty_print (
    std::ostream & out ) [virtual]
```

Prints the expression with more clarity.

Parameters

out	<i>out</i>	output stream
-----	------------	---------------

Implements [Expr](#).

5.5.2.5 print()

```
void Var::print (
    std::ostream & out ) [virtual]
```

Prints the expression.

Parameters

out	<i>out</i>	output stream
-----	------------	---------------

Implements [Expr](#).

5.5.2.6 subst()

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

This function substitutes the expression with the combination of sub-expressions if possible.

Parameters

s	first argument, string which can be replaced as a part of the expression
e	second argument, Expressiont

Returns

Expression which is modified if combination of sub expressions was possible

Implements [Expr](#).

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

- [/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp](#)
- [/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.cpp](#)

Chapter 6

File Documentation

6.1 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/↵ MSDScript/MSDScript/cmdline.cpp File Reference

Contains the code for identifying the argument given ahead of running the executable on the command line is valid and if it is valid executes the command.

```
#include "cmdline.hpp"
```

Functions

- void `use_arguments` (int argc, char *argv[])
Identifies the argument given ahead of running the executable.

6.1.1 Detailed Description

Contains the code for identifying the argument given ahead of running the executable on the command line is valid and if it is valid executes the command.

Author

Lakshay Santosh Kucheriya

6.1.2 Function Documentation

6.1.2.1 `use_arguments()`

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

Identifies the argument given ahead of running the executable.

Parameters

<i>argc</i>	first parameter, stores the number of command line arguments passed by the user
<i>argv</i>	second parameter, is array of character pointers listing all the arguments.

6.2 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/cmdline.hpp File Reference

Header file for `cmdline.cpp` file.

```
#include "catch.h"
#include <stdio.h>
#include <iostream>
#include <string>
```

Functions

- void `use_arguments` (int argc, char *argv[])
Identifies the argument given ahead of running the executable.

6.2.1 Detailed Description

Header file for `cmdline.cpp` file.

Author

Lakshay Santosh Kucheriya

6.2.2 Function Documentation

6.2.2.1 `use_arguments()`

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

Identifies the argument given ahead of running the executable.

Parameters

<i>argc</i>	first parameter, stores the number of command line arguments passed by the user
<i>argv</i>	second parameter, is array of character pointers listing all the arguments.

6.3 cmdline.hpp

[Go to the documentation of this file.](#)

```
00001 //
00002 //  cmdline.hpp
00003 //  MSDScript
00004 //
00005 //  Created by Lakshay Santosh Kucheriya on 1/16/23.
00006 //
00007
00015 #ifndef cmdline_hpp
00016 #define cmdline_hpp
00017 //
00018 #include "catch.h"
00019 #include <stdio.h>
00020 #include <iostream>
00021 #include <string>
00022
00023 void use_arguments(int argc, char* argv[]) ;
00024
00025 #endif /* cmdline_hpp */
```

6.4 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.cpp File Reference ↩

Contains expression class definition.

```
#include "expr.hpp"
```

Functions

- [operator_precedence](#) `pretty_print_at (Expr *e)`

6.4.1 Detailed Description

Contains expression class definition.

Author

Lakshay Santosh Kucheriya

6.5 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp File Reference ↩

Header file for [expr.cpp](#) file.

```
#include <stdio.h>
#include <string>
#include <iostream>
#include <stdexcept>
#include <sstream>
```

Classes

- class [Expr](#)
Base Class for representing a expression.
- class [Num](#)
Derived Class from [Expr](#) for representing a number.
- class [Add](#)
Derived Class from [Expr](#) for representing a expression involving addition.
- class [Mult](#)
Derived Class from [Expr](#) for representing a expression involving addition.
- class [Var](#)
Derived Class from [Expr](#) for representing a expression involving Variable.

Enumerations

- enum [operator_precedence](#) { [precedence_none](#) = 0 , [precedence_add](#) = 1 , [precedence_mult](#) = 2 }
Enumeration for assigning the precedence of the operators.

Functions

- [operator_precedence](#) [pretty_print_at](#) ([Expr](#) *e)

6.5.1 Detailed Description

Header file for [expr.cpp](#) file.

Author

Lakshay Santosh Kucheriya

6.6 [expr.hpp](#)

[Go to the documentation of this file.](#)

```
00001 //
00002 //  expr.hpp
00003 //  MSDScript
00004 //
00005 //  Created by Lakshay Santosh Kucheriya on 1/23/23.
00006 //
00007
00015 #ifndef expr_hpp
00016 #define expr_hpp
00017
00018 #include <stdio.h>
00019 #include <string>
00020 #include <iostream>
00021 #include <stdexcept>
00022 #include <sstream>
00023
00024
00027 class Expr // Base class
00028 {
00029 public:
00030     virtual bool equals(Expr *e) = 0 ; // This function checks for the equality for the LHS and the
    RHS.
00031     virtual int interp() = 0 ; // This function interprets the value of the expression/variable.
00032     virtual bool has_variable() = 0 ; // This function determines if the the expression consists of a
    variable or not.
```



```

00033     virtual Expr* subst(std::string s, Expr* e) = 0; // This function substitutes the expression with
the combination of sub-expressions if possible.
00034     virtual void print(std::ostream &out) = 0; // This function prints the expression.
00035     virtual void pretty_print(std::ostream &out) = 0; // This is an extension of the print function
with minor changes.
00036
00037     std::string to_string() {
00038         std::stringstream st("");
00039         this->print(st);
00040         return st.str();
00041     }
00042
00043
00044     std::string to_pretty_string() {
00045         std::stringstream st("");
00046         this->pretty_print(st);
00047         return st.str();
00048     }
00049
00050 };
00051
00052
00053 class Num : public Expr
00054 {
00055 public:
00056     int val;
00057
00058     Num(int val);
00059     bool equals(Expr *e) ;
00060     int interp();
00061     bool has_variable();
00062     Expr* subst(std::string s, Expr* e) ;
00063     void print(std::ostream &out);
00064     void pretty_print(std::ostream &out);
00065 };
00066
00067
00068 class Add : public Expr
00069 {
00070 public:
00071     Expr *lhs;
00072     Expr *rhs;
00073
00074     Add(Expr *lhs, Expr *rhs);
00075     bool equals(Expr *e) ;
00076     int interp();
00077     bool has_variable();
00078     Expr* subst(std::string s, Expr* e) ;
00079     void print(std::ostream &out);
00080     void pretty_print(std::ostream &out);
00081 };
00082
00083
00084 class Mult : public Expr
00085 {
00086 public:
00087     Expr *lhs;
00088     Expr *rhs;
00089
00090     Mult(Expr *lhs, Expr *rhs);
00091     bool equals(Expr *e) ;
00092     int interp();
00093     bool has_variable();
00094     Expr* subst(std::string s, Expr* e) ;
00095     void print(std::ostream &out);
00096     void pretty_print(std::ostream &out);
00097 };
00098
00099
00100 class Var : public Expr
00101 {
00102 public:
00103     std::string name;
00104     Var(std::string name);
00105
00106     bool equals(Expr *e);
00107     int interp();
00108     bool has_variable();
00109     Expr* subst(std::string s, Expr* e) ;
00110     void print(std::ostream &out);
00111     void pretty_print(std::ostream &out);
00112 };
00113
00114
00115 enum operator_precedence {

```

```

00128     precedence_none = 0,
00129     precedence_add = 1,
00130     precedence_mult = 2
00131 };
00132
00133 operator_precedence pretty_print_at (Expr *e);
00134
00135
00136 #endif /* expr_hpp */

```

6.7 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/↵ MSDScript/MSDScript/main.cpp File Reference

Entry point for the project.

```

#include <iostream>
#include "cmdline.hpp"

```

Functions

- `int main (int argc, char **argv)`

6.7.1 Detailed Description

Entry point for the project.

Author

Lakshay Santosh Kucheriya

6.8 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/↵ MSDScript/MSDScript/tests.cpp File Reference

contains all the tests forl the methods present in this project

```

#include "catch.h"
#include "expr.hpp"
#include <stdio.h>

```

Functions

- `TEST_CASE ("equals")`
- `TEST_CASE ("Tests for checking interp")`
- `TEST_CASE ("Test for has_variable")`
- `TEST_CASE ("Test for subst")`
- `TEST_CASE ("to_string")`
- `TEST_CASE ("to_pretty_string")`

6.8.1 Detailed Description

contains all the tests forl the methods present in this project

Author

Lakshay Santosh Kucheriya

Index

[/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/cmdline.cpp](#),
[25](#)
[/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/cmdline.hpp](#),
[26](#), [27](#)
[/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/expr.cpp](#),
[27](#)
[/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/expr.hpp](#),
[27](#), [28](#)
[/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/main.cpp](#),
[30](#)
[/Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/tests.cpp](#),
[30](#)

Add, [9](#)
 [equals](#), [10](#)
 [has_variable](#), [10](#)
 [interp](#), [11](#)
 [pretty_print](#), [11](#)
 [print](#), [11](#)
 [subst](#), [12](#)

cmdline.cpp
 [use_arguments](#), [25](#)

cmdline.hpp
 [use_arguments](#), [26](#)

[equals](#)
 Add, [10](#)
 Expr, [13](#)
 Mult, [15](#)
 Num, [19](#)
 Var, [22](#)

Expr, [12](#)
 [equals](#), [13](#)
 [has_variable](#), [13](#)
 [interp](#), [13](#)
 [pretty_print](#), [13](#)
 [print](#), [14](#)
 [subst](#), [14](#)

[has_variable](#)
 Add, [10](#)
 Expr, [13](#)
 Mult, [16](#)
 Num, [19](#)
 Var, [22](#)

[interp](#)
 Add, [11](#)
 Expr, [13](#)
 Mult, [16](#)

[Num](#), [18](#)
 [equals](#), [19](#)
 [has_variable](#), [19](#)
 [interp](#), [19](#)
 [pretty_print](#), [19](#)
 [print](#), [20](#)
 [subst](#), [20](#)

[pretty_print](#)
 Add, [11](#)
 Expr, [13](#)
 Mult, [16](#)
 Num, [19](#)
 Var, [22](#)

[print](#)
 Add, [11](#)
 Expr, [14](#)
 Mult, [17](#)
 Num, [20](#)
 Var, [23](#)

[subst](#)
 Add, [12](#)
 Expr, [14](#)
 Mult, [17](#)
 Num, [20](#)
 Var, [23](#)

[use_arguments](#)
 cmdline.cpp, [25](#)
 cmdline.hpp, [26](#)

Var, [21](#)
 [equals](#), [22](#)
 [has_variable](#), [22](#)
 [interp](#), [22](#)
 [pretty_print](#), [22](#)
 [print](#), [23](#)
 [subst](#), [23](#)