

Contents

1	继承	关系索	引											1
	1.1	类继承	《关系			 	-	 1						
2	类索	3 1												3
	2.1	类列表	ŧ			 		 3						
3	文件	索引												5
	3.1	文件列]表			 		 5						
4	类说	明												7
	4.1	Atom 🕏	类参考 .			 		 7						
		4.1.1	详细描述	<u>.</u>		 		 8						
		4.1.2	构造及机	「构函数说	明 .	 		 8						
			4.1.2.1	Atom() .		 		 8						
			4.1.2.2	\sim Atom()		 		 9						
		4.1.3	成员函数	放明		 		 9						
			4.1.3.1	copy() .		 		 9						
			4.1.3.2	display()		 		 9						
			4.1.3.3	eval()		 		 10						
	4.2	Bool 类	总参考			 		 11						
		4.2.1	详细描述	<u>.</u>		 		 13						
		4.2.2	构造及机	「构函数说	明 .	 		 13						
			4.2.2.1	Bool() [1	/2] .	 		 13						
			4.2.2.2	Bool() [2	/2] .	 		 13						
			4.2.2.3	~Bool()		 	 		 	 	 	 		 14

ii CONTENTS

	4.2.3	成员函数说明	14
		4.2.3.1 display()	14
		4.2.3.2 print()	14
	4.2.4	友元及相关函数文档	15
		4.2.4.1 Bracket	15
		4.2.4.2 Data	15
		4.2.4.3 Runtime	15
	4.2.5	类成员变量说明	15
		4.2.5.1 value	15
4.3	Bracke	t 类参考	15
	4.3.1	详细描述	18
	4.3.2	构造及析构函数说明	18
		4.3.2.1 Bracket() [1/2]	18
		4.3.2.2 Bracket() [2/2]	18
		4.3.2.3 ~Bracket()	19
	4.3.3	成员函数说明	19
		4.3.3.1 copy()	19
		4.3.3.2 display()	19
		4.3.3.3 eval()	20
		4.3.3.4 genParaData()	22
	4.3.4	友元及相关函数文档	22
		4.3.4.1 Function	22
		4.3.4.2 Runtime	22
	4.3.5	类成员变量说明	22
		4.3.5.1 func	22
		4.3.5.2 para	23
4.4	Code 3	类 <u>参考</u>	23
	4.4.1	详细描述	23
	4.4.2	成员函数说明	24
		4.4.2.1 FinalShell()	24

CONTENTS

	4.4.3	类成员变量说明	25
		4.4.3.1 brackets	25
4.5	Data ∄	参考	25
	4.5.1	详细描述	28
	4.5.2	构造及析构函数说明	28
		4.5.2.1 Data() [1/2]	28
		4.5.2.2 Data() [2/2]	28
		4.5.2.3 ~Data()	29
	4.5.3	成员函数说明	29
		4.5.3.1 bigger()	29
		4.5.3.2 check()	30
		4.5.3.3 cite()	31
		4.5.3.4 display() [1/2]	32
		4.5.3.5 display() [2/2]	32
		4.5.3.6 divide()	33
		4.5.3.7 equal()	33
		4.5.3.8 minus()	34
		4.5.3.9 mod()	35
		4.5.3.10 newData()	36
		4.5.3.11 plus()	37
		4.5.3.12 print() [1/2]	38
		4.5.3.13 print() [2/2]	38
		4.5.3.14 smaller()	39
		4.5.3.15 times()	40
		4.5.3.16 uncite()	41
	4.5.4	友元及相关函数文档	41
		4.5.4.1 Bool	41
		4.5.4.2 Integer	41
		4.5.4.3 VarEnv	41
	4.5.5	类成员变量说明	41

iv CONTENTS

		4.5.5.1 cited	 	 41
		4.5.5.2 falseData	 	 42
		4.5.5.3 trueData	 	 42
4.6	Function	on 类参考	 	 42
	4.6.1	详细描述	 	 43
	4.6.2	构造及析构函数说明	 	 43
		4.6.2.1 Function()	 	 43
		4.6.2.2 ∼Function()	 	 43
	4.6.3	成员函数说明	 	 43
		4.6.3.1 call()	 	 43
		4.6.3.2 eval()	 	 44
		4.6.3.3 newFunction()	 	 45
	4.6.4	类成员变量说明	 	 45
		4.6.4.1 body	 	 45
		4.6.4.2 FuncEnv	 	 45
		4.6.4.3 paras	 	 46
4.7	Integer	类参考	 	 46
	4.7.1	详细描述	 	 49
	4.7.2	构造及析构函数说明	 	 49
		4.7.2.1 Integer() [1/2]	 	 49
		4.7.2.2 Integer() [2/2]	 	 49
		4.7.2.3 ∼Integer()	 	 50
	4.7.3	成员函数说明	 	 50
		4.7.3.1 display()	 	 50
		4.7.3.2 divide()	 	 50
		4.7.3.3 minus()	 	 51
		4.7.3.4 plus()	 	 52
		4.7.3.5 print()	 	 53
		4.7.3.6 times()	 	 53
	4.7.4	友元及相关函数文档	 	 54

CONTENTS

		4.7.4.1 Data	54
		4.7.4.2 Runtime	54
	4.7.5	类成员变量说明	54
		4.7.5.1 value	54
4.8	Runtim	не 类参考	55
	4.8.1	详细描述	55
	4.8.2	构造及析构函数说明	56
		4.8.2.1 Runtime()	56
		4.8.2.2 ~Runtime()	56
	4.8.3	成员函数说明	56
		4.8.3.1 assignVar()	56
		4.8.3.2 assignVarToTop()	57
		4.8.3.3 display()	57
		4.8.3.4 getVar()	57
		4.8.3.5 popVarEnv()	58
		4.8.3.6 pushVarEnv()	58
		4.8.3.7 setVar()	58
	4.8.4	类成员变量说明	59
		4.8.4.1 varEnvs	59
4.9	Symbo	ll 类参考	59
	4.9.1	详细描述	61
	4.9.2	构造及析构函数说明	62
		4.9.2.1 Symbol()	62
		4.9.2.2 ~Symbol()	62
	4.9.3	成员函数说明	62
		4.9.3.1 copy()	62
		4.9.3.2 display()	62
		4.9.3.3 eval()	63
	4.9.4	友元及相关函数文档	64
		4.9.4.1 Bracket	64

vi

			4.9.4.2	Function	 	 	 	 	 	64
			4.9.4.3	Runtime	 	 	 	 	 	64
		4.9.5	类成员变	量说明	 	 	 	 	 	64
			4.9.5.1	name	 	 	 	 	 	64
	4.10	VarEnv	类参考		 	 	 	 	 	64
		4.10.1	详细描述		 	 	 	 	 	65
		4.10.2	构造及析	构函数说明	 	 	 	 	 	65
			4.10.2.1	VarEnv()	 	 	 	 	 	65
			4.10.2.2	\sim VarEnv() .	 	 	 	 	 	65
		4.10.3	成员函数	说明	 	 	 	 	 	65
			4.10.3.1	assignVar() .	 	 	 	 	 	65
			4.10.3.2	display()	 	 	 	 	 	66
				getVar()						66
		4.10.4	类成员变	量说明	 	 	 	 	 	66
				vars						66
5	文件	说明								69
	5.1	Atom.c		考						69
		5.1.1	函数说明		 	 	 	 	 	70
			5.1.1.1	getWord() .	 	 	 	 	 	70
			5.1.1.2	isBlank()	 	 	 	 	 	70
			5.1.1.3	printSpace()	 	 	 	 	 	71
	5.2	Atom.h	文件参考		 	 	 	 	 	71
	5.3	Bool.cp	p 文件参	考	 	 	 	 	 	72
	5.4	Code.c	pp 文件参	考	 	 	 	 	 	73
		5.4.1	函数说明		 	 	 	 	 	73
			5.4.1.1	isValidCode()	 	 	 	 	 	73
	5.5	Code.h	文件参考		 	 	 	 	 	74
	5.6	Data.cp	p 文件参	考	 	 	 	 	 	75
	5.7	Data.h	文件参考		 	 	 	 	 	75
	5.8	Functio	n.cpp 文件	参考	 	 	 	 	 	76
	5.9	Functio	n.h 文件参	考	 	 	 	 	 	77
	5.10	Integer	.cpp 文件	多考	 	 	 	 	 	79
	5.11	main.cp	op 文件参	考	 	 	 	 	 	79
		5.11.1	函数说明		 	 	 	 	 	80
			5.11.1.1	main()	 	 	 	 	 	80
	5.12	Runtim	e.cpp 文件	参考	 	 	 	 	 	80
	5.13	Runtim	e.h 文件参	考	 	 	 	 	 	81
	5.14	VarEnv	.cpp 文件	多考	 	 	 	 	 	82
	5.15	VarEnv	.h 文件参	考	 	 	 	 	 	83
索	引									85

Chapter 1

继承关系索引

1.1 类继承关系

此继承关系列表按字典顺序粗略的排序:

Atom	
Bracket	15
Symbol	59
Code	
Data	25
Bool	
Integer	46
Function	
Runtime	
VarEnv	64

2 继承关系索引

Chapter 2

类索引

2.1 类列表

这里列出了所有类、结构、联合以及接口定义等,并附带简要说明:

Atom .							 							 					 						7
Bool							 							 					 						- 11
Bracket							 							 					 						15
Code .							 							 											23
Data							 							 											25
Function							 							 											42
Integer																									
Runtime																									
Symbol																									
VarEnv							 							 		_			 						64

4 类索引

Chapter 3

文件索引

3.1 文件列表

这里列出了所有文件,并附带简要说明:

Atom.cpp	69
Atom.h	71
Bool.cpp	72
Code.cpp	73
Code.h	74
Data.cpp	75
Data.h	75
Function.cpp	76
Function.h	77
Integer.cpp	79
main.cpp	79
Runtime.cpp	80
Runtime.h	81
VarEnv.cpp	82
VarEny h	83

文件索引

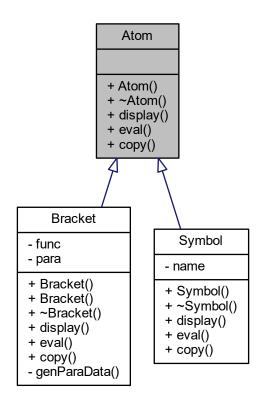
Chapter 4

类说明

4.1 Atom 类参考

#include <Atom.h>

类 Atom 继承关系图:



Atom 的协作图:

Atom

- + Atom()
- + ~Atom()
- + display()
- + eval()
- + copy()

Public 成员函数

- Atom ()
- virtual ∼Atom ()
- virtual void display (int indent)=0
- virtual Data * eval (Runtime *runtime)=0
- virtual Atom * copy ()=0

4.1.1 详细描述

用于存储语法结构的抽象类

4.1.2 构造及析构函数说明

4.1.2.1 Atom()

Atom::Atom () [inline]

空构造函数

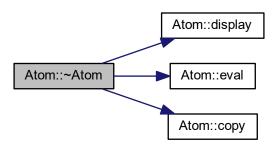
返回

4.1 Atom 类参考 9

4.1.2.2 \sim Atom()

```
virtual Atom::~Atom ( ) [inline], [virtual]
```

虚析构函数函数调用图:



4.1.3 成员函数说明

4.1.3.1 copy()

```
virtual Atom* Atom::copy ( ) [pure virtual]
```

复制一个Atom 并返回

返回

在 Symbol, 以及 Bracket 内被实现.

这是这个函数的调用关系图:



4.1.3.2 display()

```
virtual void Atom::display (
                int indent ) [pure virtual]
```

显示语法树,并且使用 indent 级的缩进

参数

indent

在 Symbol, 以及 Bracket 内被实现.

这是这个函数的调用关系图:



4.1.3.3 eval()

根据 runtime 来求Atom 对应的值

参数

runtime 用于获取变量对应的数据、进行赋值运算等

返回

当前Atom 的值,比如如果当前Atom 是 (add 1 2) 则返回一个值为 3 的数据

在 Symbol, 以及 Bracket 内被实现.

这是这个函数的调用关系图:



该类的文档由以下文件生成:

• Atom.h

4.2 Bool 类参考 11

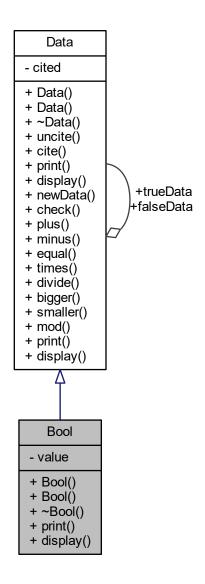
4.2 Bool 类参考

#include <Data.h>

类 Bool 继承关系图:



Bool 的协作图:



Public 成员函数

- Bool (bool value)
- Bool (bool value, int)
- virtual \sim Bool ()
- virtual void print ()
- · virtual void display ()

Private 属性

• bool value

4.2 Bool 类参考 13

友元

- class Data
- class Bracket
- class Runtime

额外继承的成员函数

4.2.1 详细描述

表示一个Bool 数据

4.2.2 构造及析构函数说明

```
4.2.2.1 Bool() [1/2]
```

```
Bool::Bool (

bool value )
```

根据 value 为 true 或 false 来初始化

参数

value 用于初始化 value 属性

返回

```
4.2.2.2 Bool() [2/2]
```

用于初始化全局使用的 true 和 false

参数

value

返回

4.2.2.3 ∼Bool()

Bool:: \sim Bool () [virtual]

析构函数

4.2.3 成员函数说明

4.2.3.1 display()

void Bool::display () [virtual]

打印当前对象的详细信息

实现了 Data.

函数调用图:



4.2.3.2 print()

void Bool::print () [virtual]

打印当前对象的值

实现了 Data.

这是这个函数的调用关系图:



4.3 Bracket 类参考 15

4.2.4 友元及相关函数文档

4.2.4.1 Bracket

```
friend class Bracket [friend]
```

4.2.4.2 Data

```
friend class Data [friend]
```

4.2.4.3 Runtime

```
friend class Runtime [friend]
```

4.2.5 类成员变量说明

4.2.5.1 value

```
bool Bool::value [private]
```

存储该对象的值

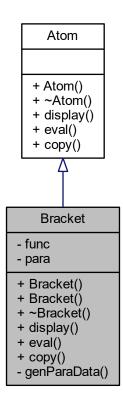
该类的文档由以下文件生成:

- Data.h
- Bool.cpp

4.3 Bracket 类参考

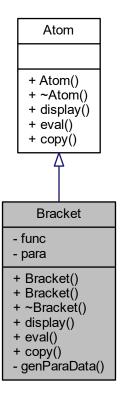
#include <Atom.h>

类 Bracket 继承关系图:



4.3 Bracket 类参考 17

Bracket 的协作图:



Public 成员函数

- Bracket (const std::string &str, unsigned long &pos)
- Bracket ()
- virtual \sim Bracket ()
- virtual void display (int indent)
- virtual Data * eval (Runtime *runtime)
- virtual Atom * copy ()

Private 成员函数

std::vector< Data * > * genParaData (Runtime *runtime)

Private 属性

- std::string func
- std::list< Atom * > para

友元

- class Runtime
- class Function

4.3.1 详细描述

用于存储一个括号的语法结构的类

4.3.2 构造及析构函数说明

4.3.2.1 Bracket() [1/2]

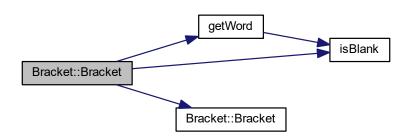
从 pos 位置开始,从 str 字符串中取出一个完整的括号表达式,并生成该表达式对应的语法树存储在新建的Bracket 对象里

参数

str	源代码
pos	用于作为"光标"所在位置,函数会修改 pos 到获取的完整的括号表达式最后一个字符的位置加一

返回

函数调用图:



4.3.2.2 Bracket() [2/2]

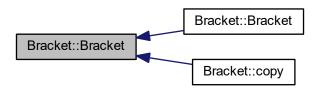
Bracket::Bracket ()

默认的空构造函数

4.3 Bracket 类参考 19

返回

这是这个函数的调用关系图:



```
4.3.2.3 ∼Bracket()
```

```
Bracket::~Bracket () [virtual]
析构函数
```

4.3.3 成员函数说明

4.3.3.1 copy()

```
Atom * Bracket::copy () [virtual] 复制一份当前对象并返回
```

返回

复制得到的对象

实现了 Atom.

函数调用图:



4.3.3.2 display()

打印Bracket 对象对应的语句,并使用 indent 级的缩进

参数

indent 打印时使用的缩进深度

实现了 Atom.

函数调用图:



4.3.3.3 eval()

根据 runtime 来求本对象对应的表达式的值

参数

runtime 用于获取求值时需要的变量的值以及进行赋值操作等

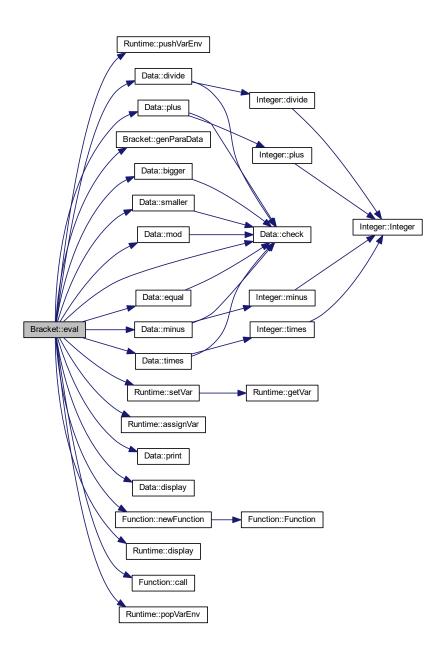
返回

本对象对应表达式的值

实现了 Atom.

4.3 Bracket 类参考 21

函数调用图:



这是这个函数的调用关系图:



4.3.3.4 genParaData()

根据 para 属性生成参数的具体数据

参数

返回

返回一个根据参数生成的数据的 vector

这是这个函数的调用关系图:



4.3.4 友元及相关函数文档

4.3.4.1 Function

```
friend class Function [friend]
```

4.3.4.2 Runtime

```
friend class Runtime [friend]
```

4.3.5 类成员变量说明

4.3.5.1 func

```
std::string Bracket::func [private]
```

用于存储该表达式对应的函数名称,比如 (test 1 2) 对应的表达式的函数名称就是 test

4.4 Code 类参考 23

4.3.5.2 para

```
std::list<Atom *> Bracket::para [private]
```

用于存储参数对应的语法结构

该类的文档由以下文件生成:

- Atom.h
- Atom.cpp

4.4 Code 类参考

#include <Code.h>

Code 的协作图:

Code

- brackets

+ FinalShell()

静态 Public 成员函数

• static void FinalShell ()

Private 属性

• std::list < std::string * > brackets

4.4.1 详细描述

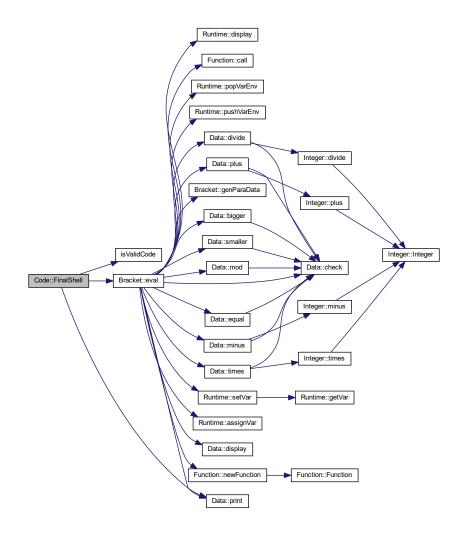
用于提供一个交互运行代码的Shell

4.4.2 成员函数说明

4.4.2.1 FinalShell()

void Code::FinalShell () [static]

用于提供一个交互运行代码的Shell 函数调用图:



这是这个函数的调用关系图:



4.5 Data 类参考 25

4.4.3	类成员变量说明
-------	---------

4.4.3.1 brackets

std::list<std::string *> Code::brackets [private]

用于存储用户输入的代码

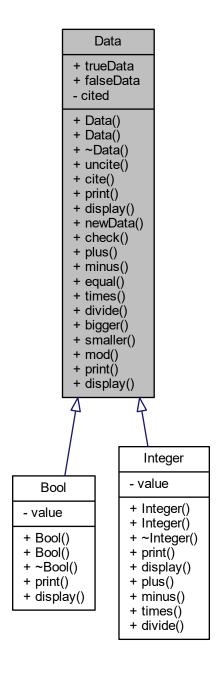
该类的文档由以下文件生成:

- Code.h
- Code.cpp

4.5 Data 类参考

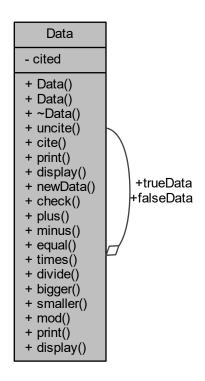
#include <Data.h>

类 Data 继承关系图:



4.5 Data 类参考 27

Data 的协作图:



Public 成员函数

- Data ()
- Data (int)
- virtual ~Data ()
- void uncite ()
- void cite ()
- virtual void print ()=0
- virtual void display ()=0

静态 Public 成员函数

- static Data * newData (const std::string &str)
- static void check (Data *data)
- static Data * plus (std::vector < Data *> *dataPara)
- static Data * minus (std::vector < Data *> *dataPara)
- static Data * equal (std::vector < Data *> *dataPara)
- static Data * times (std::vector < Data *> *dataPara)
- static Data * divide (std::vector < Data *> *dataPara)
- static Data * bigger (std::vector < Data *> *dataPara)
- static Data * smaller (std::vector< Data *> *dataPara)
- static Data * mod (std::vector< Data *> *dataPara)
- static Data * print (std::vector< Data *> *dataPara)
- static Data * display (std::vector < Data *> *dataPara)

静态 Public 属性

```
    static Data * trueData = new Bool(true, 1)
    static Data * falseData = new Bool(false, 1)
```

Private 属性

· unsigned int cited

友元

- class Integer
- class Bool
- class VarEnv

4.5.1 详细描述

表示数据的抽象类,数据的类型与值由派生类存储与管理,Data 类只负责对数据进行引用计数

4.5.2 构造及析构函数说明

```
4.5.2.1 Data() [1/2]

Data::Data ()

默认构造函数,将引用次数设为 0
```

返回

```
4.5.2.2 Data() [2/2]

Data::Data (

int )
```

用于生成公用的 true false 数据对象,为了避免这两个对象被析构,在函数体中会将引用次数初始化为 1

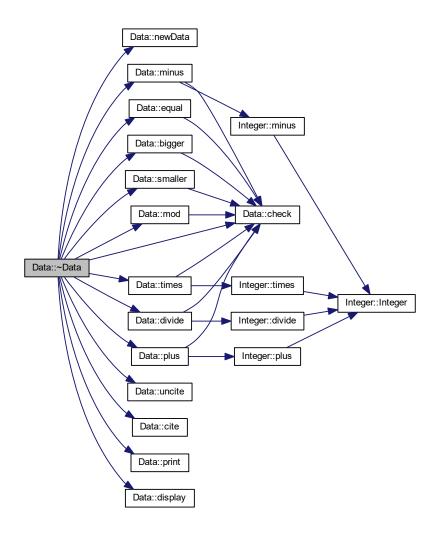
返回

4.5 Data 类参考 29

4.5.2.3 \sim Data()

```
virtual Data::~Data ( ) [inline], [virtual]
```

虚析构函数函数调用图:



4.5.3 成员函数说明

4.5.3.1 bigger()

比较两个数的大小

参数

dataPara

被判断的是 dataPara 的前两个数据,更多的数据将会被忽略,被比较的数据只可以 是Integer 类

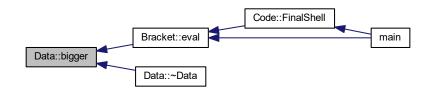
返回

一个Bool 类型的数据,如果 dataPara[0] > dataPara[1]则值为 true,否则为 false

函数调用图:



这是这个函数的调用关系图:



4.5.3.2 check()

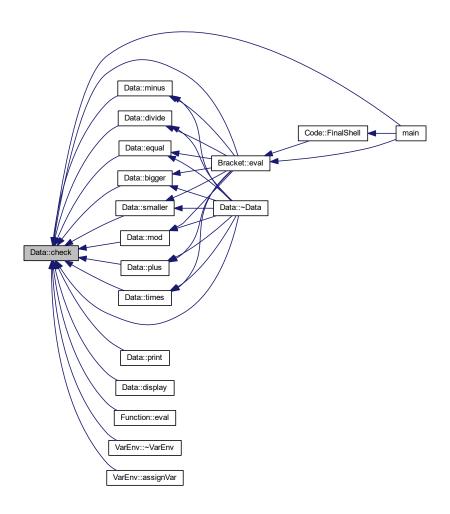
检查 data 对象的引用次数是否为 0,如果为 0,则析构这个对象

参数

data 指向被检查的对象的指针

4.5 Data 类参考 31

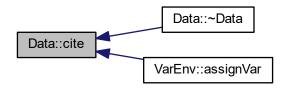
这是这个函数的调用关系图:



4.5.3.3 cite()

void Data::cite ()

将对象的引用次数增加1这是这个函数的调用关系图:



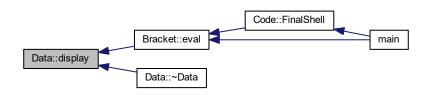
4.5.3.4 display() [1/2]

```
virtual void Data::display ( ) [pure virtual]
```

打印对象的详细信息,包括被引用次数等

在 Bool, 以及 Integer 内被实现.

这是这个函数的调用关系图:



4.5.3.5 display() [2/2]

逐个打印一组数据的详细信息

参数

dataPara 被打印的数据

返回

一个值为 true 的Bool 类型数据

函数调用图:



4.5 Data 类参考 33

4.5.3.6 divide()

两个整数的除法

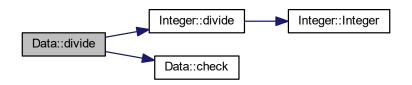
参数

dataPara dataPara[0]是被除数,dataPara[1]是除数(不能为 0),更多的参数会被忽略

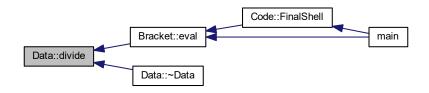
返回

除法的结果

函数调用图:



这是这个函数的调用关系图:



4.5.3.7 equal()

判断两个数据是否相等

参数

dataPara

被判断的是 dataPara 的前两个数据,更多的数据将会被忽略,被比较的数据可以是Bool 类的对象,也可以是Integer 类的对象

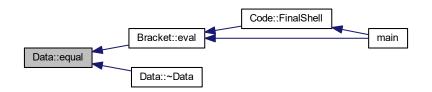
返回

一个Bool 类型的数据

函数调用图:



这是这个函数的调用关系图:



4.5.3.8 minus()

两个整数的减法

参数

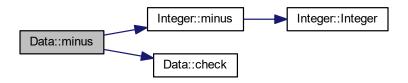
dataPara dataPara[0]是被减数,dataPara[1]是减数,更多的参数会被忽略

返回

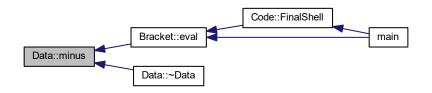
减法的结果

4.5 Data 类参考 35

函数调用图:



这是这个函数的调用关系图:



4.5.3.9 mod()

求两个数的余数

参数

dataPara dataPara[0]是被除数, dataPara[1]是除数(不能为 0), 更多的参数会被忽略

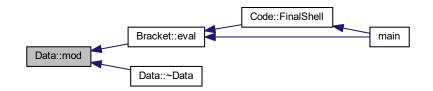
返回

dataPara[0]除以 dataPara[1]得到的余数

函数调用图:



这是这个函数的调用关系图:



4.5.3.10 newData()

根据 str 的内容构造一个数据

参数

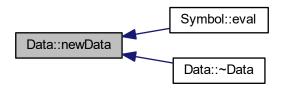
str 作为构造数据时的依据

4.5 Data 类参考 37

返回

str 对应的数据, 比如 str=="123" 则返回一个指向值为 123 的整数类对象的指针

这是这个函数的调用关系图:



4.5.3.11 plus()

任意数量的整数的加法

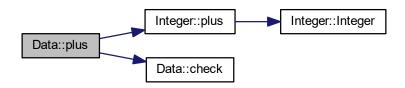
参数

dataPara 被加数

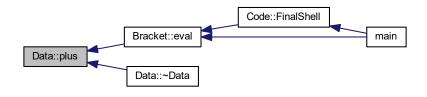
返回

得到的加法的结果

函数调用图:



这是这个函数的调用关系图:



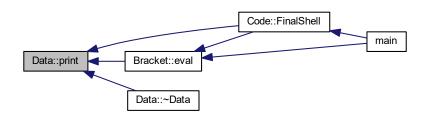
4.5.3.12 print() [1/2]

virtual void Data::print () [pure virtual]

打印对象

在 Bool, 以及 Integer 内被实现.

这是这个函数的调用关系图:



4.5.3.13 print() [2/2]

逐个打印一组数据

参数

dataPara	被打印的数据

4.5 Data 类参考 39

返回

一个值为 true 的Bool 类型数据

函数调用图:



4.5.3.14 smaller()

比较两个数的大小

参数

dataPara 被判断的是 dataPara 的前两个数据,更多的数据将会被忽略,被比较的数据只可以是Integer 类

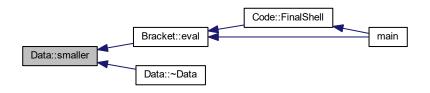
返回

一个Bool 类型的数据,如果 dataPara[0] < dataPara[1]则值为 true,否则为 false

函数调用图:



这是这个函数的调用关系图:



4.5.3.15 times()

任意数量的整数的乘法

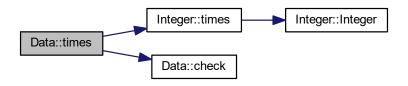
参数

dataPara 被乘数

返回

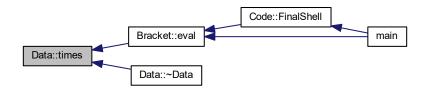
得到的乘法的结果

函数调用图:



4.5 Data 类参考 41

这是这个函数的调用关系图:



4.5.3.16 uncite()

void Data::uncite ()

将对象的引用次数减少1这是这个函数的调用关系图:



4.5.4 友元及相关函数文档

4.5.4.1 Bool

friend class Bool [friend]

4.5.4.2 Integer

friend class Integer [friend]

4.5.4.3 VarEnv

friend class VarEnv [friend]

4.5.5 类成员变量说明

4.5.5.1 cited

unsigned int Data::cited [private]

记录当前对象被引用的次数

4.5.5.2 falseData

```
Data * Data::falseData = new Bool(false, 1) [static]
```

用于作为全局唯一的值为 false 的Bool 类对象使用

4.5.5.3 trueData

```
Data * Data::trueData = new Bool(true, 1) [static]
```

用于作为全局唯一的值为 true 的Bool 类对象使用

该类的文档由以下文件生成:

- Data.h
- Data.cpp

4.6 Function 类参考

#include <Function.h>

Function 的协作图:

Function

- paras
- body
- FuncEnv
- + Function()
- + ~Function()
- + eval()
- + newFunction()
- + call()

Public 成员函数

- Function ()
- ∼Function ()
- Data * eval (std::vector < Data *> *parasData)

4.6 Function 类参考 43

静态 Public 成员函数

- static void newFunction (std::list< Atom *> *source)
- static Data * call (std::string &name, std::vector< Data *> *parasData)

Private 属性

- std::list< std::string > paras
- std::list< Atom * > body

静态 Private 属性

• static std::map< std::string, Function * > FuncEnv

4.6.1 详细描述

用户自定义的函数

4.6.2 构造及析构函数说明

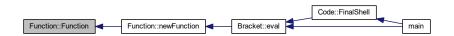
4.6.2.1 Function()

```
Function::Function ( )
```

默认构造函数

返回

这是这个函数的调用关系图:



4.6.2.2 ∼Function()

```
Function::\simFunction ( )
```

析构函数

4.6.3 成员函数说明

4.6.3.1 call()

调用名称为 name 的函数,并将相关参数传递过去

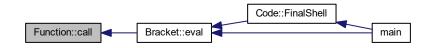
参数

name	被调用函数的名称
parasData	调用函数时使用的参数

返回

指向函数返回的数据的指针

这是这个函数的调用关系图:



4.6.3.2 eval()

根据参数值求函数的返回值

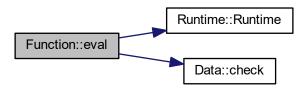
参数

parasData	参数值
-----------	-----

返回

指向函数返回的数据的指针

函数调用图:



4.6 Function 类参考 45

4.6.3.3 newFunction()

```
void Function::newFunction (
          std::list< Atom *> * source ) [static]
```

根据传入的语法树构造一个新的函数,并录入FuncEnv属性中

参数

source

包括构造Function 对象所需要的全部信息,比如: (function (test a b) (assign a (+ a b)) (echo a)) 传入参数的内容就是 (test a b) (assign a (+ a b)) (echo a) 对应的语法树

函数调用图:



这是这个函数的调用关系图:



4.6.4 类成员变量说明

4.6.4.1 body

```
std::list<Atom *> Function::body [private]
```

存储函数体

4.6.4.2 FuncEnv

```
std::map< std::string, Function * > Function::FuncEnv [static], [private]
```

存储函数名及其对应的Function 对象

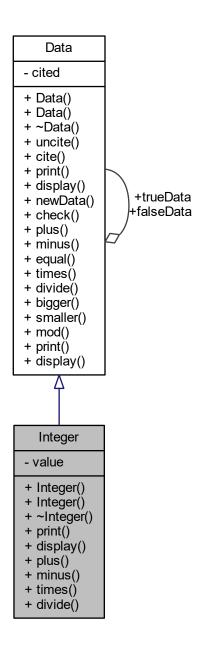
46 类说明 4.6.4.3 paras std::list<std::string> Function::paras [private] 用于存储函数的参数名称 该类的文档由以下文件生成: • Function.h • Function.cpp 4.7 Integer 类参考

4.7 Integer 类参考 47

类 Integer 继承关系图:

Data + trueData + falseData - cited + Data() + Data() + ~Data() + uncite() + cite() + print() + display() + newData() + check() + plus() + minus() + equal() + times() + divide() + bigger() + smaller() + mod() + print() + display() Integer - value + Integer() + Integer() + ~Integer() + print() + display() + plus() + minus() + times() + divide()

Integer 的协作图:



Public 成员函数

- Integer ()
- Integer (const long &value)
- virtual ∼Integer ()
- virtual void print ()
- virtual void display ()

4.7 Integer 类参考 49

静态 Public 成员函数

```
    static Data * plus (std::vector< Data *> *dataPara)
    static Data * minus (std::vector< Data *> *dataPara)
    static Data * times (std::vector< Data *> *dataPara)
    static Data * divide (std::vector< Data *> *dataPara)
```

Private 属性

• long value

友元

- class Data
- · class Runtime

额外继承的成员函数

4.7.1 详细描述

表示一个整数

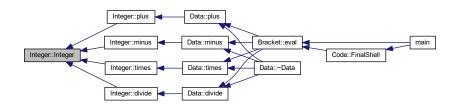
4.7.2 构造及析构函数说明

```
4.7.2.1 Integer() [1/2]

Integer::Integer ( )
```

构造函数 返回

这是这个函数的调用关系图:



```
4.7.2.2 Integer() [2/2]

Integer::Integer (

const long & value)
```

将值设置为 value

参数

value 用于初始化 value 属性的整数

返回

```
4.7.2.3 \simInteger()
```

```
Integer::~Integer ( ) [virtual]
```

析构函数

4.7.3 成员函数说明

4.7.3.1 display()

```
void Integer::display ( ) [virtual]
```

打印当前对象的详细信息

实现了 Data.

4.7.3.2 divide()

两个整数的除法

参数

dataPara dataPara[0]是被除数, dataPara[1]是除数(不能为0), 更多的参数会被忽略

4.7 Integer 类参考 51

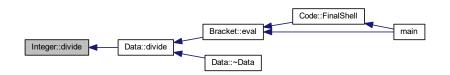
返回

除法的结果

函数调用图:



这是这个函数的调用关系图:



4.7.3.3 minus()

两个整数的减法

参数

dataPara dataPara[0]是被减数, dataPara[1]是减数, 更多的参数会被忽略

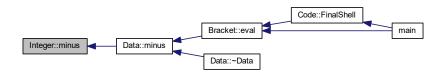
返回

减法的结果

函数调用图:



这是这个函数的调用关系图:



4.7.3.4 plus()

任意数量的整数的加法

参数

dataPara 被加数

4.7 Integer 类参考 53

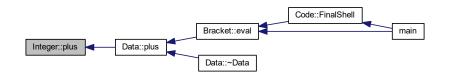
返回

得到的加法的结果

函数调用图:



这是这个函数的调用关系图:



4.7.3.5 print()

```
void Integer::print ( ) [virtual]
```

打印当前对象的值

实现了 Data.

4.7.3.6 times()

任意数量的整数的乘法

参数

dataPara 被乘数

返回

得到的乘法的结果

函数调用图:



这是这个函数的调用关系图:



4.7.4 友元及相关函数文档

4.7.4.1 Data

friend class Data [friend]

4.7.4.2 Runtime

friend class Runtime [friend]

4.7.5 类成员变量说明

4.7.5.1 value

long Integer::value [private]

存储对象的数值

该类的文档由以下文件生成:

- Data.h
- Integer.cpp

4.8 Runtime 类参考 55

4.8 Runtime 类参考

#include <Runtime.h>

Runtime 的协作图:

Runtime - varEnvs + Runtime() + ~Runtime() + assignVar() + assignVarToTop() + setVar() + getVar() + pushVarEnv() + popVarEnv() + display()

Public 成员函数

- Runtime ()
- ∼Runtime ()
- void assignVar (std::string &name, Data *data)
- void assignVarToTop (std::string &name, Data *data)
- void setVar (std::string &name, Data *data)
- Data * getVar (std::string &name)
- void pushVarEnv ()
- void popVarEnv ()
- void display ()

Private 属性

std::forward_list< VarEnv * > varEnvs

4.8.1 详细描述

运行时,运行一段代码时,会从Runtime 对象中获得变量名对应的数据,以及进行赋值操作等

4.8.2 构造及析构函数说明

4.8.2.1 Runtime()

```
Runtime::Runtime ( )
```

构造函数

返回

这是这个函数的调用关系图:



4.8.2.2 \sim Runtime()

Runtime:: \sim Runtime ()

析构函数

4.8.3 成员函数说明

4.8.3.1 assignVar()

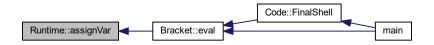
将变量名 name 指向 data 数据,如果没有在Runtime 中找到 name 变量,就会新建一个名称为 name 的变量,并指向 data,如果找到了,则将 name 原来指向的数据的被引用次数减少一,在将 name 指向 data

参数

name	被赋值的变量名称
data	待赋值的数据

4.8 Runtime 类参考 57

这是这个函数的调用关系图:



4.8.3.2 assignVarToTop()

在最顶层的VarEnv 中新建变量并赋值,用于实现函数调用时传递参数

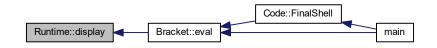
参数

name	被赋值的变量名称
data	待赋值的数据

4.8.3.3 display()

```
void Runtime::display ( )
```

打印该对象中所有的变量的详细信息这是这个函数的调用关系图:



4.8.3.4 getVar()

获取 name 变量名对应的数据

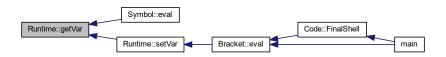
参数

name | 待获取的变量的名称

返回

name 对应的数据

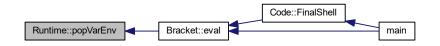
这是这个函数的调用关系图:



4.8.3.5 popVarEnv()

```
void Runtime::popVarEnv ( )
```

出栈一个VarEnv 并析构它这是这个函数的调用关系图:



4.8.3.6 pushVarEnv()

```
void Runtime::pushVarEnv ( )
```

构造一个VarEnv 并入栈这是这个函数的调用关系图:



4.8.3.7 setVar()

```
void Runtime::setVar (
          std::string & name,
          Data * data )
```

将 name 对应的整数重新设置为 data 的值,和 assign 的区别在于,set 会改变 name 指向的数据的值

4.9 Symbol 类参考 59

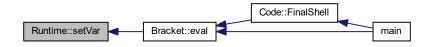
参数

name	被赋值的变量名称
data	待赋值的数据

函数调用图:



这是这个函数的调用关系图:



4.8.4 类成员变量说明

4.8.4.1 varEnvs

std::forward_list<VarEnv *> Runtime::varEnvs [private]

作为一个存储变量的栈使用

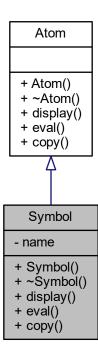
该类的文档由以下文件生成:

- Runtime.h
- Runtime.cpp

4.9 Symbol 类参考

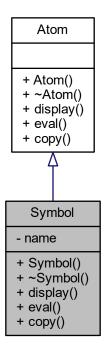
#include <Atom.h>

类 Symbol 继承关系图:



4.9 Symbol 类参考 61

Symbol 的协作图:



Public 成员函数

- Symbol (const std::string &str)
- virtual ∼Symbol ()
- virtual void display (int indent)
- virtual Data * eval (Runtime *runtime)
- virtual Atom * copy ()

Private 属性

• std::string name

友元

- class Runtime
- class Bracket
- class Function

4.9.1 详细描述

用于存储一个符号对应的语法结构,这个符号可以是一个常量,也可以是一个变量名,比如 (test 1 2 a) 中的 1、2、a 都是一个符号

4.9.2 构造及析构函数说明

```
4.9.2.1 Symbol()
```

```
Symbol::Symbol ( {\tt const\ std::string\ \&\ \it str}\ )
```

根据 str 来构造

参数

```
str 比如为"true" "false" "123" "a" 等
```

返回

```
4.9.2.2 \simSymbol()
```

```
Symbol::~Symbol ( ) [virtual]
```

析构函数

4.9.3 成员函数说明

4.9.3.1 copy()

```
Atom * Symbol::copy ( ) [virtual]
```

复制当前对象并返回

返回

返回复制得到的对象

实现了 Atom.

4.9.3.2 display()

打印本对象对应的符号,并使用 indent 级的缩进

4.9 Symbol 类参考 63

参数

indent 打印时使用的缩进级数

实现了 Atom.

函数调用图:



4.9.3.3 eval()

获取当前对象的值,比如如果 name 属性是一个常量,比如"123",就返回一个值为 123 的整数数据,如果是变量名就在 runtime 中查询该变量名对应的数据并且返回

参数

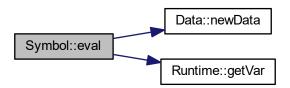
runtime 用于获取变量对应的数据

返回

如果 name 属性是一个常量,比如"123",就返回一个值为 123 的整数数据,如果是变量名就在 runtime 中查询该变量名对应的数据并且返回

实现了 Atom.

函数调用图:



4.9.4 友元及相关函数文档

4.9.4.1 Bracket

```
friend class Bracket [friend]
```

4.9.4.2 Function

```
friend class Function [friend]
```

4.9.4.3 Runtime

```
friend class Runtime [friend]
```

4.9.5 类成员变量说明

4.9.5.1 name

```
std::string Symbol::name [private]
```

用于存储符号对应的字符串,比如"a" "123" "true" "false" 等

该类的文档由以下文件生成:

- Atom.h
- Atom.cpp

4.10 VarEnv 类参考

#include <VarEnv.h>

VarEnv 的协作图:

VarEnv - vars + VarEnv() + ~VarEnv() + getVar() + assignVar() + display()

4.10 VarEnv 类参考 65

Public 成员函数

- VarEnv ()
- ∼VarEnv ()
- Data * getVar (const std::string &name)
- void assignVar (const std::string &name, Data *data)
- · void display ()

Private 属性

• std::map< std::string, Data * > vars

4.10.1 详细描述

变量环境

4.10.2 构造及析构函数说明

4.10.2.1 VarEnv()

```
VarEnv::VarEnv ( )
```

构造函数

返回

4.10.2.2 \sim VarEnv()

```
VarEnv::~VarEnv ( )
```

析构函数函数调用图:



4.10.3 成员函数说明

4.10.3.1 assignVar()

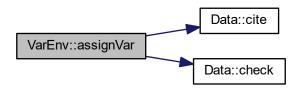
将变量名 name 指向 data 数据,如果没有在Runtime 中找到 name 变量,就会新建一个名称为 name 的变量,并指向 data,如果找到了,则将 name 原来指向的数据的被引用次数减少一,在将 name 指向 data

66 类说明

参数

name	被赋值的变量名称
data	待赋值的数据

函数调用图:



4.10.3.2 display()

```
void VarEnv::display ( )
```

打印该对象中所有的变量的详细信息

4.10.3.3 getVar()

获取 name 变量名对应的数据

参数

name	待获取的变量的名称
------	-----------

返回

name 对应的数据,如果未找到,返回的是NULL

4.10.4 类成员变量说明

4.10.4.1 vars

```
std::map<std::string, Data *> VarEnv::vars [private]
```

用于存储变量名及其对应的数据

该类的文档由以下文件生成:

4.10 VarEnv 类参考 67

- VarEnv.h
- VarEnv.cpp

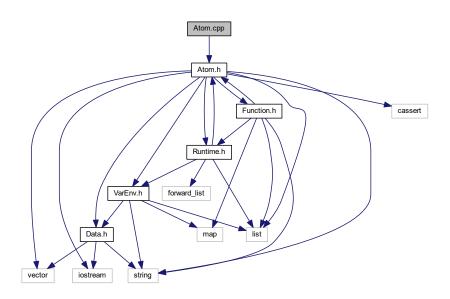
68 类说明

Chapter 5

文件说明

5.1 Atom.cpp 文件参考

#include "Atom.h" Atom.cpp 的引用 (Include) 关系图:



函数

- void printSpace (int i)
- bool isBlank (const char &achar)
- std::string getWord (const std::string &str, unsigned long &pos)

5.1.1 函数说明

5.1.1.1 getWord()

函数调用图:

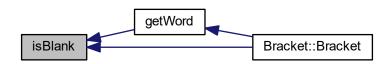


这是这个函数的调用关系图:



5.1.1.2 isBlank()

这是这个函数的调用关系图:

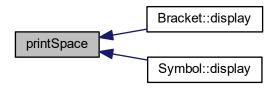


5.2 Atom.h 文件参考 71

5.1.1.3 printSpace()

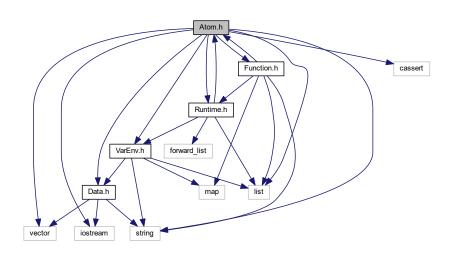
```
\begin{tabular}{ll} \beg
```

这是这个函数的调用关系图:

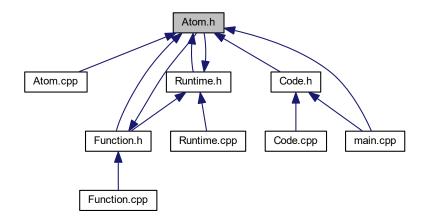


5.2 Atom.h 文件参考

```
#include <vector>
#include <iostream>
#include <string>
#include <list>
#include <cassert>
#include "Data.h"
#include "Runtime.h"
#include "Function.h"
#include "VarEnv.h"
Atom.h 的引用 (Include) 关系图:
```



此图展示该文件直接或间接的被哪些文件引用了:

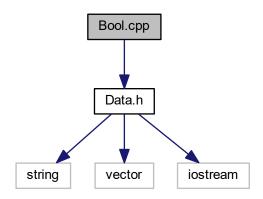


类

- class Atom
- · class Bracket
- class Symbol

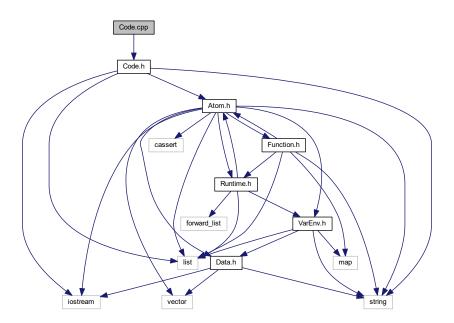
5.3 Bool.cpp 文件参考

#include "Data.h" Bool.cpp 的引用 (Include) 关系图:



5.4 Code.cpp 文件参考

#include "Code.h" Code.cpp 的引用 (Include) 关系图:



函数

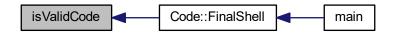
• bool isValidCode (std::string &str)

5.4.1 函数说明

5.4.1.1 isValidCode()

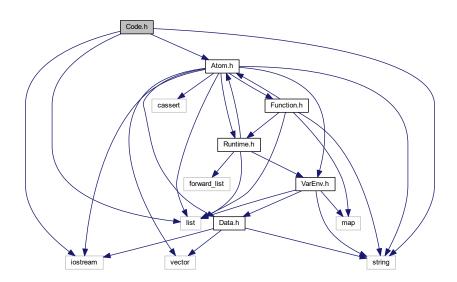
```
bool isValidCode ( std::string \ \& \ str \ )
```

这是这个函数的调用关系图:

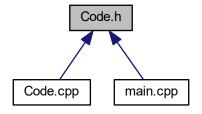


5.5 Code.h 文件参考

#include <string> #include <list> #include <iostream> #include "Atom.h" Code.h 的引用 (Include) 关系图:



此图展示该文件直接或间接的被哪些文件引用了:

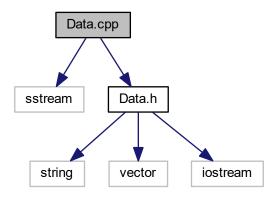


类

• class Code

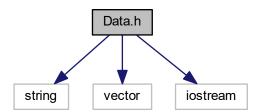
5.6 Data.cpp 文件参考

#include <sstream> #include "Data.h" Data.cpp 的引用 (Include) 关系图:

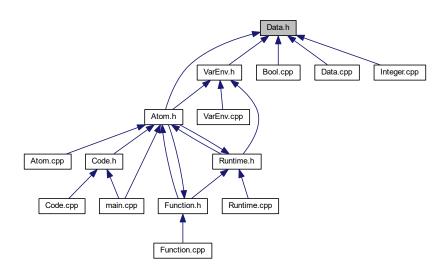


5.7 Data.h 文件参考

#include <string> #include <vector> #include <iostream> Data.h 的引用 (Include) 关系图:



此图展示该文件直接或间接的被哪些文件引用了:

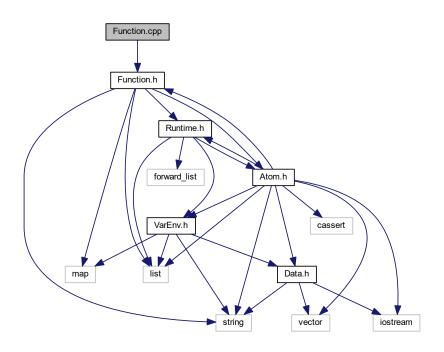


类

- class Data
- class Integer
- class Bool

5.8 Function.cpp 文件参考

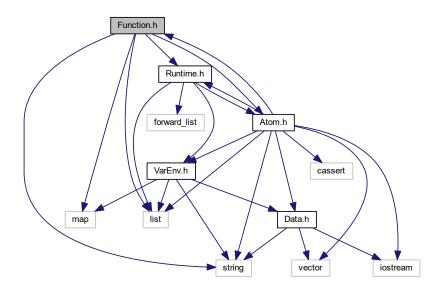
Function.cpp 的引用 (Include) 关系图:



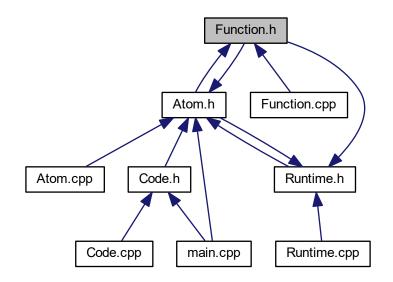
5.9 Function.h 文件参考

```
#include <list>
#include <string>
#include <map>
#include "Runtime.h"
#include "Atom.h"
```

Function.h 的引用 (Include) 关系图:



此图展示该文件直接或间接的被哪些文件引用了:

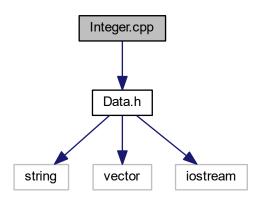


类

• class Function

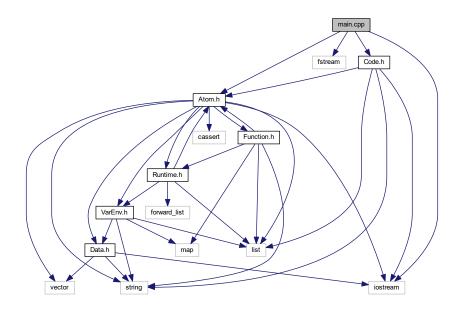
5.10 Integer.cpp 文件参考

#include "Data.h" Integer.cpp 的引用 (Include) 关系图:



5.11 main.cpp 文件参考

#include <iostream> #include <fstream> #include "Code.h" #include "Atom.h" main.cpp的引用 (Include) 关系图:



函数

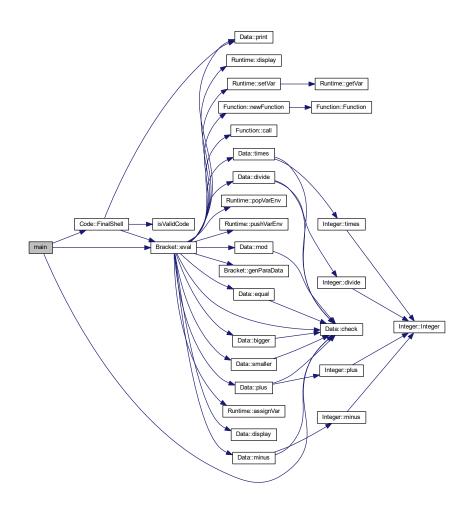
• int main (int argc, char *argv[])

5.11.1 函数说明

5.11.1.1 main()

```
int main (
          int argc,
          char * argv[] )
```

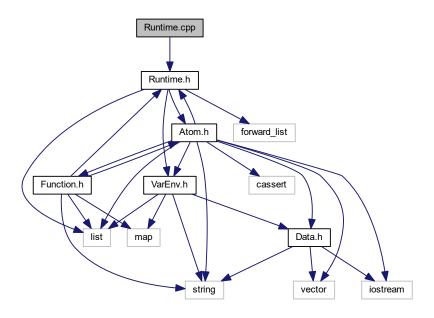
函数调用图:



5.12 Runtime.cpp 文件参考

#include "Runtime.h"

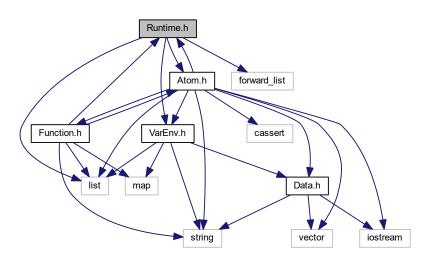
Runtime.cpp 的引用 (Include) 关系图:



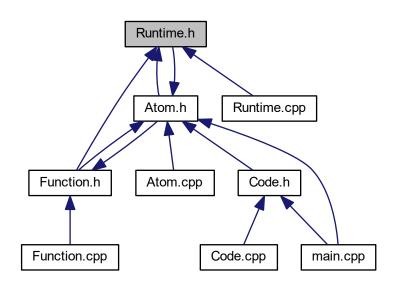
5.13 Runtime.h 文件参考

```
#include <list>
#include <forward_list>
#include "VarEnv.h"
#include "Atom.h"
```

Runtime.h 的引用 (Include) 关系图:



此图展示该文件直接或间接的被哪些文件引用了:

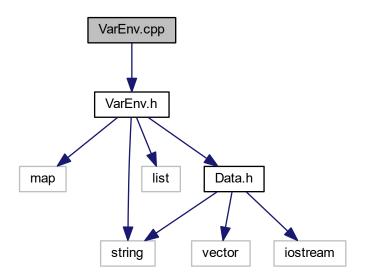


类

• class Runtime

5.14 VarEnv.cpp 文件参考

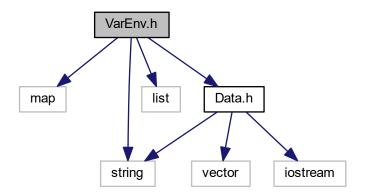
VarEnv.cpp 的引用 (Include) 关系图:



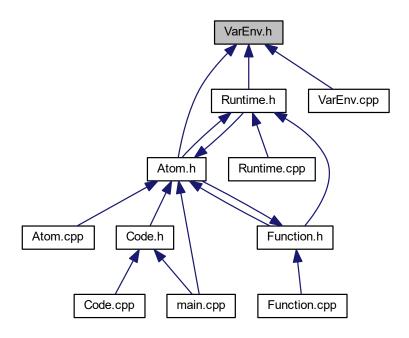
5.15 VarEnv.h 文件参考

```
#include <map>
#include <string>
#include <list>
#include "Data.h"

VarEnv.h 的引用 (Include) 关系图:
```



此图展示该文件直接或间接的被哪些文件引用了:



类

• class VarEnv

Index

~Atom, 8 ~Bool Bool, 13 ~Bracket Bracket, 19 ~Data Data, 28 ~Function Function, 43 ~Integer Integer, 50 ~Runtime Runtime, 56 ~Symbol Symbol, 62	Bracket, 15 ~Bracket, 19 Bool, 15 Bracket, 18 copy, 19 display, 19 eval, 20 func, 22 Function, 22 genParaData, 21 para, 22 Runtime, 22 Symbol, 64 brackets Code, 25
~VarEnv	call
	Function, 43
VarEnv, 65	check
assignVar	Data, 30
Runtime, 56	cite
VarEnv, 65	Data, 31
assignVarToTop	cited
Runtime, 57	Data, 41
Atom, 7	Code, 23
~Atom, 8	brackets, 25
Atom, 8	FinalShell, 24
copy, 9	Code.cpp, 73
display, 9	isValidCode, 73
eval, 10	Code.h, 74
Atom.cpp, 69	сору
getWord, 70	Atom, 9
isBlank, 70	Bracket, 19
printSpace, 70	Symbol, 62
Atom.h, 71	D
higgor	Data, 25 ∼Data, 28
bigger	,
Data, 29 body	bigger, 29 Bool, 15, 41
Function, 45	check, 30
Bool. 11	cite, 31
~Bool, 13	cited, 41
Bool, 13	Data, 28
Bracket, 15	display, 31, 32
Data, 15, 41	divide, 32
display, 14	equal, 33
print, 14	falseData, 41
Runtime, 15	Integer, 41, 54
value, 15	minus, 34
Bool.cpp, 72	mod, 35
• •	•

86 INDEX

newData, 36 plus, 37 print, 38 smaller, 39 times, 40 trueData, 42 uncite, 41 VarEnv, 41 Data, 47 Data, 47 Data, 47 Data, 31, 32 Integer, 50 Data, 41 Data, 31, 32 Integer, 49 Data, 31, 32 Integer, 50 Data, 31 Data, 31 Puntime, 57 Symbol, 62 VarEnv, 66 Data, 33 eval Data, 33 eval Atom, 10 Bracket, 20 Function, 44 Symbol, 63 Facket, 22 Funcer Function, 45 Function, 47 Funcer, 45 Function, 47 Funcer, 45 Function, 48 Function, 49 Function, 49 Function, 40 Function, 40 Function, 41 Funcer, 55 Function, 42 Puncer, 45 Function, 45 Function, 45 Function, 45 Function, 47 Function, 47 Function, 48 Function, 49 Function, 49 Function, 49 Function, 40 Function, 41 Funcer, 55 Function, 41 Funcer, 55 Function, 69		
print, 38	newData, 36	Atom.cpp, 70
mailer, 39 times, 40 trueData, 42 uncite, 41 Uncite, 42 Uncite, 41 Uncite, 41 Uncite, 52 Uncite, 41 Uncite, 41 Uncite, 41 Uncite, 52 Uncite, 42 Uncite, 41 Uncite, 50 Uncite, 50 Uncite, 41 Uncite, 51 Uncite, 42 Uncite, 42 Uncite, 43 Uncite, 50 Uncite, 44 Uncite, 50 Uncite, 45 Uncite, 50 Uncite, 5	plus, 37	
times, 40 trueData, 42 uncite, 41 VarEnv, 41 Data. cpp, 75 minus, 51 Data. ph, 75 display Atom, 9 Bool, 14 Bracket, 19 Data, 31, 32 Integer, 50 Runtime, 57 Symbol, 62 VarEnv, 66 Data, 33 eval Atom, 10 Bracket, 20 Function, 44 Symbol, 63 Bracket, 22 Functenv Function, 45 Function, 42 ParaBata Bracket, 22 Call, 43 eval, 44 FuncEnv, 45 Function, 43 body, 45 Function, 43 bracket, 20 Function, 44 FuncEnv, 45 Function, 43 bracket, 20 Function, 42 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 47 Function, 48 Function, 49 Function, 59 Function, 50 Function, 50 Function, 50 Function, 50 Function, 50 Function, 5	•	
trueData, 42 uncite, 41 VarEnv, 41 lnteger, 49 Data.cpp, 75 Data.h, 75 Data.h, 75 Data.h, 75 Diata.h, 75 Bool, 14 Diata.ga, 21 Diata.ga, 31, 32 Diata.ga, 31 Diata.ga, 32 Diata.ga, 32 Diata.ga, 32 Diata.ga, 33 Diata.ga, 34 Diata.ga, 36 Diata.ga, 36 Diata.ga, 37 Diata.ga, 36 Diata.ga, 36 Diata.ga, 37 Diata.ga, 36 Diata.ga, 37 Diata.ga, 36 Diata.ga, 37 Diata.ga, 36 Diata.ga, 37 Diata.ga, 36 Diata.ga, 36 Diata.ga, 37 Diata.ga, 36 Diata.ga, 37 Diata.ga, 36 Diat		_
uncite, 41 VarEnv, 41 Data.cpp, 75 Data.h, 76 Data.h, 7		
VarEnv, 41 Data.cpp, 75 Data.cpp, 75 Data.h, 75 display Atom, 9 Bool, 14 Bracket, 19 Data, 31, 32 Integer, 50 Runtime, 57 Symbol, 62 VarEnv, 66 Code.cpp, 73 divide Data, 32 Integer, 50 main.cpp, 80 main.cpp, 70 squal Data, 33 minus Data, 33 eval Atom, 10 Bracket, 20 Function, 44 Symbol, 63 Code, 24 Function, 44 FuncEnv Function, 45 Bracket, 22 call, 43 call, 43 call, 44 FuncEnv Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv Function, 45 Function, 45 Function, 46 Bracket, 22 call, 43 eval, 44 FuncEnv Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 46 Bracket, 27 call, 49 popVarEnv Runtime, 58 Function, 49 poplyarEnv Runtime, 58 Function, 59 Function, 77 pushVarEnv genParaData Bracket, 21 getVar Runtime, 55 Funttime, 56 cassignVar, 56		
Data.cpp, 75 minus, 51 Data.h, 75 plus, 52 display print, 53 Atom, 9 Runtime, 54 Bool, 14 times, 53 Bracket, 19 value, 54 Data, 31, 32 Integer.cpp, 79 Integer, 50 Atom.cpp, 70 Symbol, 62 isValidCode VarEnv, 66 Code.cpp, 73 divide main Data, 32 main Integer, 50 main.cpp, 70 main.cpp, 80 main.cpp, 80 main.cpp, 79 main.go equal main.go Data, 33 minus eval Data, 34 Atom, 10 Integer, 51 Bracket, 20 mod Function, 44 Data, 35 Symbol, 63 name falseData Symbol, 64 Data, 35 name falseData Symbol, 64 Data, 35 name falseData Symbol, 64 Data, 36 newFunction		
Data.n., 75 plus, 52 display print, 53 Atom, 9 Runtime, 54 Bool, 14 times, 53 Bracket, 19 value, 54 Data, 31, 32 Integer.cpp, 79 Integer, 50 isBlank Runtime, 67 Atom.cpp, 70 Symbol, 62 validCode VarEnv, 66 Code.cpp, 73 divide main Data, 32 main Integer, 50 main.cpp, 79 main, 80 main.cpp, 70 main, 80 main.cpp, 80 main, 80 main.cpp, 80 main, 80 main.cpp, 80 main, 80 main.cpp, 80 main, 80		_
display print, 53 Atom, 9 Runtime, 54 Bool, 14 times, 53 Bracket, 19 value, 54 Data, 31, 32 Integer.cpp, 79 Integer, 50 isBlank Runtime, 57 Atom.cpp, 70 Symbol, 62 isValidCode VarErw, 66 Code.cpp, 73 divide main.cpp, 79 Data, 32 main Integer, 50 main.cpp, 79 equal main.cpp, 79 equal main.cpp, 79 equal main.gpp, 70 paras mod Lata, 33 minus eval paras Function, 44 paras Function, 45 paras Function, 45 paras	• •	
Atom, 9 Bool, 14 Bracket, 19 Data, 31, 32 Integer, 50 Runtime, 57 Symbol, 62 VarEnv, 66 Code.cpp, 73 divide Data, 32 Integer, 50 main Data, 32 Integer, 50 main.cpp, 70 isValidCode Code.cpp, 73 divide Data, 32 Integer, 50 main.cpp, 80 main.cpp, 79 main, 80 minus Data, 33 eval Atom, 10 Bracket, 20 Function, 44 Symbol, 63 falseData Data, 41 FinalShell Code, 24 func Bracket, 22 Function, 42 Function, 45 Bracket, 22 Function, 42 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Bracket, 22 call, 43 eval, 44 Funcenv, 45 Function, 43 popVarEnv Function, 44 Funcenv, 45 Function, 43 print Bracket, 22 popVarEnv Function, 43 print Runtime, 58 Function, 66 Function, 77 pushVarEnv Runtime, 58 Function, 77 pushVarEnv Runtime, 58 Function, 77 pushVarEnv Runtime, 58 Function, 57 VarEnv, 66 Runtime, 55 Runtime, 56 assignVar, 56		•
Bool, 14 Bracket, 19 Data, 31, 32 Integer, 50 Integer, 50 Integer, 50 Runtime, 57 Symbol, 62 VarEnv, 66 Code.cpp, 73 divide Data, 32 Integer, 50 main.cpp, 80 main.cpp, 79 main, 80 main.cpp, 70 push VarEnv Runtime, 58 Function, 43 popVarEnv Runtime, 55 Runtime, 55 Runtime, 57 VarEnv, 66 assignVar, 56	• •	-
Bracket, 19 Data, 31, 32 Integer, 50 Integer, 50 Runtime, 57 Symbol, 62 VarEnv, 66 Data, 32 Integer, 50 equal Data, 32 Integer, 50 main.cpp, 70 equal Data, 33 eval Adom, 10 Bracket, 20 Function, 44 Symbol, 63 falseData Data, 41 FinalShell Code, 24 Funcenv Function, 45 Function, 45 Bracket, 22 Function, 42 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 Funcenv, 45 Function, 43 newFunction, 43 print Bracket, 22 call, 43 eval, 44 Funcenv, 45 Function, 43 newFunction, 45 Function, 43 newFunction, 44 Funcenv, 45 Function, 43 newFunction, 43 newFunction, 43 newFunction, 44 paras, 45 Symbol, 64 Function, 55 Function, 77 pushVarEnv Runtime, 58 Function, 57 Quentime, 56 Runtime, 57 VarEnv, 66 Runtime, 57 VarEnv, 66 Integer, 73 Integer, 76 Runtime, 56 assignVar, 56		
Data, 31, 32		
Integer, 50 Runtime, 57 Symbol, 62 VarEnv, 66 VarEnv, 66 Data, 32 Integer, 50 main.cpp, 79 equal main.cpp, 79 equal main.cpp, 79 main, 80 main.cpp, 80 main.cpp, 79 main, 80 main.cpp, 79 main, 80 main.cpp, 80 main.cpp, 80 main.cpp, 80 main.cpp, 80 main.cpp, 79 main.cpp, 70 push yation main.cpp, 80 main.cpp, 70 push yation main.cpp main.cpp main.exity main.cpp main.exity main.cpp main.exity ma		
Runtime, 57 Symbol, 62 VarEnv, 66 Code.cpp, 73 divide Data, 32 Integer, 50 main.cpp, 80 main.cpp, 79 equal Data, 33 eval Data, 34 Atom, 10 Bracket, 20 Function, 44 Symbol, 63 falseData Data, 41 Paracket, 20 Code, 24 Code, 24 Code, 24 Code, 24 Function, 45 Function, 45 Bracket, 22 Function, 42 Symbol, 45 Bracket, 22 call, 43 body, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Function, 45 Function, 45 Bracket, 20 call, 43 eval, 44 FuncEnv, 45 Function, 47 Runtime, 58 Function.h, 77 pushVarEnv genParaData Bracket, 21 getVar Runtime, 55 Runtime, 57 VarEnv, 66		
Symbol, 62 isValidCode VarEnv, 66 Code.cpp, 73 divide main Data, 32 main.cpp, 80 Integer, 50 main.cpp, 79 equal main, 80 Data, 33 minus eval Data, 34 Atom, 10 Integer, 51 Bracket, 20 mod Function, 44 Data, 35 Symbol, 63 name falseData Symbol, 64 Data, 35 name falseData Symbol, 64 Data, 35 name falseData Symbol, 64 Data, 35 name falseData Data, 35 Data, 36 newFunction Code, 24 newFunction func Function, 44 Bracket, 22 para Function, 45 para Function, 45 plus Bracket, 22 para, 37 call, 43 popVarEnv Function, 44 popVarEnv Runtime, 58 printSpace Function.cpp, 76 printSpace <td></td> <td></td>		
VarEnv, 66 divide Data, 32 Integer, 50 main Data, 33 eval Atom, 10 Bracket, 20 Function, 44 Symbol, 63 FinalShell Code, 24 Function, 45 Function, 42 Function, 42 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Function, 42 Paras Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Function, 46 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Function, 46 Function, 47 paras Function, 48 eval, 44 popVarEnv Runtime, 58 Function, 58 Function, 59 Function, 77 pushVarEnv genParaData Bracket, 21 getVar Runtime, 55 Runtime, 55 Runtime, 55 Runtime, 56 assignVar, 56		
divide Data, 32 Integer, 50 main.cpp, 80 main.cpp, 79 equal Data, 33 eval Atom, 10 Bracket, 20 Function, 44 Symbol, 63 falseData Data, 36 Code, 24 func Bracket, 22 Function, 45 body, 45 body, 45 Bracket, 22 call, 43 eval, 44 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 Function, 45 Bracket, 25 Function, 45 Bracket, 26 Bracket, 27 Function, 48 body, 45 Bracket, 29 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 43 popVarEnv Function, 43 print Bool, 14 paras, 45 Symbol, 64 Integer, 53 print Bool, 14 paras, 45 Symbol, 64 Function.cpp, 76 Function.h, 77 Atom.cpp, 70 pushVarEnv genParaData Bracket, 21 getVar Runtime, 57 VarEnv, 66 assignVar, 56	-	
Data, 32 Integer, 50 Integer, 50 equal Data, 33 eval Atom, 10 Bracket, 20 Function, 44 Symbol, 63 falseData Data, 36 Code, 24 func Bracket, 22 Functon, 45 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 43 eval Enunction, 45 Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 popVarEnv Function, 43 newFunction, 44 print Bool, 14 paras, 45 Symbol, 64 Function.pp, 76 Function.pp, 76 Function, 77 pushVarEnv Runtime, 58 Function, 79 pushVarEnv Runtime, 58 Function.pp, 70 pushVarEnv Runtime, 58 Runtime, 58 Runtime, 57 VarEnv, 66 Runtime, 56 assignVar, 56		Code.cpp, 73
Integer, 50 Integer, 50 Integer, 50 Equal Data, 33 Eval Atom, 10 Bracket, 20 Function, 44 Symbol, 63 Iname falseData Data, 36 Code, 24 Function, 45 Function, 45 Function, 43 body, 45 Bracket, 22 Function, 43 Eval		main
equal main.cpp, 79 equal main, 80	•	
equal main, 80 Data, 33 eval Data, 34 Atom, 10 Bracket, 20 Function, 44 Symbol, 63 falseData Symbol, 64 Data, 41 FinalShell Data, 36 Code, 24 func Function, 45 Function, 42 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Functon, 45 Functon, 45 Functon, 45 Functon, 47 Bracket, 28 Function, 49 Bracket, 29 call, 41 FuncEnv, 45 Function, 45 Function, 45 Bracket, 26 Bracket, 27 Call, 48 eval, 44 FuncEnv, 45 Function, 45 Function, 45 Function, 45 Function, 46 Function, 47 Runtime, 58 Function, 48 Bool, 14 Data, 38 Integer, 52 Function, 49 Function, 49 Function, 49 Function, 40 Function, 41 Bool, 14 Bool,	integer, 50	
Data, 33 minus eval Data, 34 Atom, 10 Integer, 51 Bracket, 20 mod Function, 44 Data, 35 Symbol, 63 name falseData Symbol, 64 Data, 41 newData FinalShell Data, 36 Code, 24 newFunction func Function, 44 Bracket, 22 para Function, 45 Bracket, 22 Function, 45 plus Bracket, 22 paras ~Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 pushVarEnv Runtime, 58 Parasign Runtime, 58	equal	
eval Data, 34 Atom, 10 Bracket, 20 Function, 44 Symbol, 63 rame falseData Symbol, 64 Data, 41 FinalShell Data, 36 Code, 24 func Bracket, 22 Function, 45 Function, 45 Bracket, 22 Function, 42 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 Funcenv, 45 Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 Funcenv, 45 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 Funcenv, 45 Function, 45 Function, 45 Function, 47 Funcenv, 48 Function, 49 popVarEnv Runtime, 58 Function, 49 print Bool, 14 paras, 45 Symbol, 64 Function.cpp, 76 Function.cpp, 76 Function.h, 77 Runtime, 58 Function.h, 77 Runtime, 58 Function.h, 77 Runtime, 58 Function, 57 Runtime, 57 VarEnv, 66 Runtime, 56 assignVar, 56	·	
Atom, 10 Bracket, 20 Function, 44 Symbol, 63 falseData Data, 35 Symbol, 64 Data, 41 FinalShell Code, 24 func Bracket, 22 FuncEnv Function, 45 Function, 43 body, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 43 body, 45 Function, 45 Function, 45 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 43 bracket, 25 Function, 45 Function, 45 Function, 45 Function, 46 Function, 47 Funcenv, 48 Function, 49 Funcenv, 49 Funcenv, 40 Funcenv, 40 Funcenv, 41 Funcenv, 42 Funcenv, 43 Function, 43 Function, 43 Function, 44 Funcenv, 45 Function, 45 Function, 46 Function, 47 Function, 48 Function, 49 Function, 49 Function, 40 Function, 41 Function, 41 Function, 42 Function, 43 Function, 44 Function, 45 Function, 45 Function, 47 Function, 48 Function, 49 Function, 49 Function, 40 Function, 40 Function, 41 Function, 41 Function, 42 Function, 43 Function, 45 Function, 46 Function, 47 Function, 48 Function, 49 Function, 40 Fun	,	
Bracket, 20 mod Function, 44 Data, 35 Symbol, 63 name falseData Symbol, 64 Data, 41 newData FinalShell Data, 36 Code, 24 newFunction func Function, 44 Bracket, 22 para Function, 45 Bracket, 22 Function, 42 paras ∼Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv Function, 43 print newFunction, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 pushVarEnv Runtime, 58 Bracket, 21 Runtime, 55 VarEnv, 66 assignVar, 56		•
Function, 44 Symbol, 63 name falseData Symbol, 64 Data, 41 Paras, 45 Bracket, 22 Function, 42 Function, 43 Bracket, 22 Function, 45 Bracket, 22 Function, 45 Bracket, 20 Function, 45 Bracket, 20 Function, 45 Bracket, 20 Function, 40 Bracket, 20 Function, 41 Bracket, 21 Bracket, 22 Function, 45 Bracket, 20 Bracket, 21 Bracket, 22 Bracket, 21 Bracket, 21 Bracket, 21 Bracket, 21 Bracket, 22 Bracket, 22 Bracket, 22 Bracket, 22 Bracket, 22 Bracket, 21 Bracket, 22 Bracket, 2		
Symbol, 63 name falseData Symbol, 64 Data, 41 newData FinalShell Data, 36 Code, 24 newFunction func Function, 44 Bracket, 22 Function, 44 Function, 45 paras Function, 42 paras ~Function, 43 Function, 45 Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 pushVarEnv Runtime, 58 Bracket, 21 Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56		
falseData		Data, SS
Data, 41 newData FinalShell Data, 36 Code, 24 newFunction func Function, 44 Bracket, 22 Function, 45 Function, 45 Bracket, 22 Function, 42 paras ~Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv Function, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	5 , 5 5., 5 5	name
FinalShell Data, 36 Code, 24 newFunction func Function, 44 Bracket, 22 Function, 45 Function, 45 Bracket, 22 Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 58 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	falseData	Symbol, 64
Code, 24 newFunction func Function, 44 Bracket, 22 Function, 45 Function, 45 Bracket, 22 Function, 42 paras ~Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv Function, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 58 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	Data, 41	newData
func Function, 44 Bracket, 22 para Function, 45 Bracket, 22 Function, 42 paras ~Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv FuncEnv, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 58 Runtime, 57 ~Runtime, 55 VarEnv, 66 assignVar, 56	FinalShell	Data, 36
Bracket, 22 Function, 45 para Function, 42 paras ~Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv FuncEnv, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	Code, 24	newFunction
Function 45 Bracket 22 Function 42 paras ~Function 43 Function 45 body 45 plus Data 37 Bracket 22 Data 37 11 12	func	Function, 44
Function, 42 Bracket, 22 Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv FuncEnv, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 58 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	Bracket, 22	
Function, 42 paras ∼Function, 43 Function, 45 body, 45 plus Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv Function, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 pushVarEnv Runtime, 58 Bracket, 21 Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	FuncEnv	para
~Function, 43 body, 45 Bracket, 22 Call, 43 eval, 44 Function, 45 Function, 43 newFunction, 44 paras, 45 Symbol, 64 Function.cpp, 76 Function.h, 77 Atom.cpp, 70 pushVarEnv genParaData Bracket, 21 getVar Runtime, 57 VarEnv, 66 Function, 43 Function, 44 Runtime, 55 ~Runtime, 57 VarEnv, 66 Function, 45 Popular Popul	Function, 45	Bracket, 22
body, 45 Bracket, 22 Bracket, 22 call, 43 eval, 44 FuncEnv, 45 Function, 43 newFunction, 44 paras, 45 Symbol, 64 Function.cpp, 76 Function.h, 77 GenParaData Bracket, 21 getVar Runtime, 58 print Bool, 14 Data, 38 Integer, 53 PrintSpace Function.cpp, 76 Function.h, 77 Atom.cpp, 70 pushVarEnv Runtime, 58 Bracket, 21 getVar Runtime, 55 ~Runtime, 57 VarEnv, 66 Runtime, 56 assignVar, 56	Function, 42	paras
Bracket, 22 Data, 37 call, 43 Integer, 52 eval, 44 popVarEnv Function, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 pushVarEnv genParaData Runtime, 58 Bracket, 21 Runtime, 58 Runtime, 57 ∼Runtime, 56 VarEnv, 66 assignVar, 56	\sim Function, 43	Function, 45
call, 43 Integer, 52 eval, 44 popVarEnv FuncEnv, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 58 qetVar Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56		plus
eval, 44 FuncEnv, 45 Function, 43 newFunction, 44 paras, 45 Symbol, 64 Function.cpp, 76 Function.h, 77 genParaData Bracket, 21 getVar Runtime, 55 Runtime, 57 VarEnv, 66 Runtime, 56 AssignVar, 56		
FuncEnv, 45 Runtime, 58 Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 58 Runtime, 57 ~Runtime, 55 VarEnv, 66 assignVar, 56		_
Function, 43 print newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 pushVarEnv Runtime, 58 Bracket, 21 Runtime, 58 Runtime, 57 ~Runtime, 55 VarEnv, 66 assignVar, 56		
newFunction, 44 Bool, 14 paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 pushVarEnv genParaData Bracket, 21 Runtime, 58 getVar Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56		
paras, 45 Data, 38 Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 pushVarEnv genParaData Bracket, 21 Runtime, 58 getVar Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56		•
Symbol, 64 Integer, 53 Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56		
Function.cpp, 76 printSpace Function.h, 77 Atom.cpp, 70 genParaData Runtime, 58 Bracket, 21 Runtime, 55 qetVar Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	•	
Function.h, 77 Atom.cpp, 70 pushVarEnv genParaData Bracket, 21 getVar Runtime, 55 Runtime, 57 VarEnv, 66 Atom.cpp, 70 pushVarEnv Runtime, 58 Runtime, 55 ~Runtime, 55 assignVar, 56		_
genParaData pushVarEnv Bracket, 21 Runtime, 58 getVar Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	···	
genParaData Runtime, 58 Bracket, 21 Runtime, 55 getVar Runtime, 55 Runtime, 57 ~Runtime, 56 VarEnv, 66 assignVar, 56	Function.h, 77	
Bracket, 21 getVar Runtime, 55 Runtime, 57 VarEnv, 66 Runtime, 56 assignVar, 56	was David Data	
$\begin{array}{ccc} \text{getVar} & \text{Runtime, 55} \\ & \text{Runtime, 57} & \sim \text{Runtime, 56} \\ & \text{VarEnv, 66} & \text{assignVar, 56} \end{array}$	5	Runtime, 58
Runtime, 57 \sim Runtime, 56 VarEnv, 66 assignVar, 56		Dunties - FF
VarEnv, 66 assignVar, 56	_	
-		
getvvoru assignvar101op, 57		
	getword	assign var 10 10p, 5/

INDEX 87

```
Bool, 15
     Bracket, 22
     display, 57
     getVar, 57
     Integer, 54
     popVarEnv, 58
     pushVarEnv, 58
     Runtime, 56
     setVar, 58
     Symbol, 64
     varEnvs, 59
Runtime.cpp, 80
Runtime.h, 81
setVar
     Runtime, 58
smaller
     Data, 39
Symbol, 59
     \simSymbol, 62
     Bracket, 64
     copy, <mark>62</mark>
     display, 62
     eval, 63
     Function, 64
     name, 64
     Runtime, 64
     Symbol, 62
times
     Data, 40
     Integer, 53
trueData
     Data, 42
uncite
     Data, 41
value
     Bool, 15
     Integer, 54
VarEnv, 64
     \simVarEnv, 65
     assignVar, 65
     Data, 41
     display, 66
     getVar, 66
     VarEnv, 65
     vars, 66
VarEnv.cpp, 82
VarEnv.h, 83
varEnvs
     Runtime, 59
vars
     VarEnv, 66
```