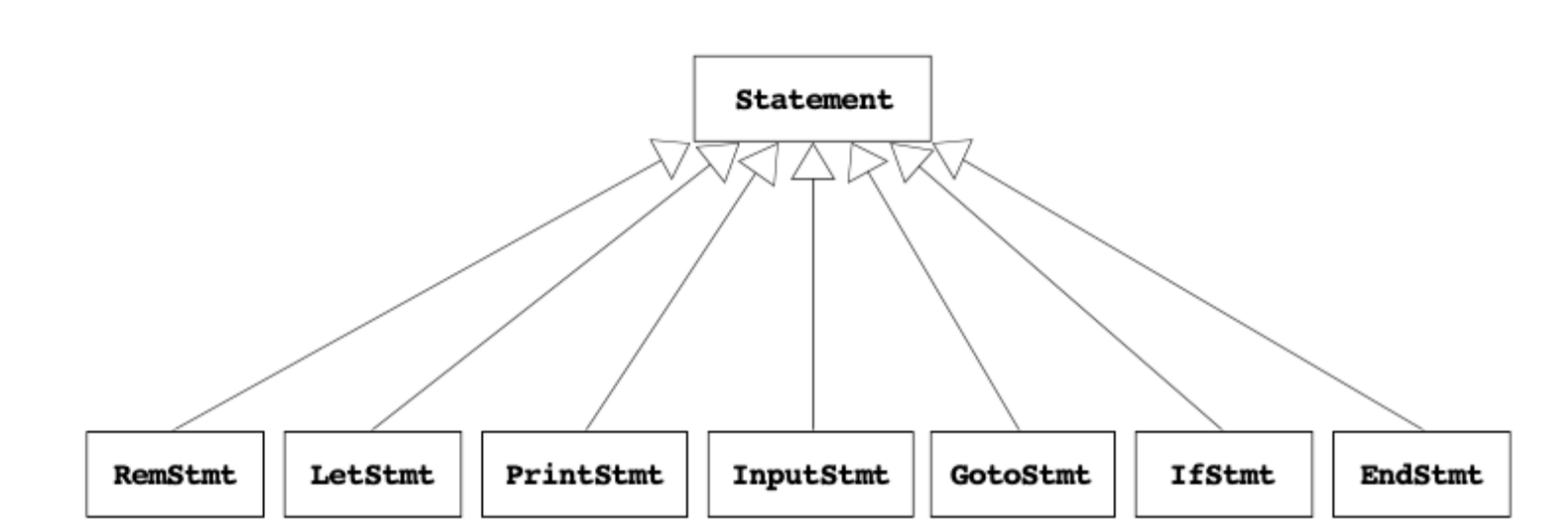
BASIC解释器设计文档

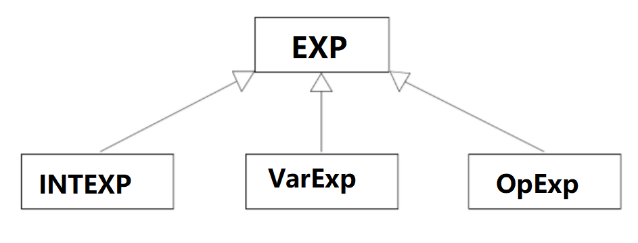
517021911052 周佳懿

**功能一览：**

1. GUI界面
2. 文件的读取和保存
3. 程序的输入
4. Basic程序的语法分析、语义分析、解释运行（要求的基础功能都有实现）
5. 语法错误提示（变量名是否规范、语句是否有行号等）
6. 命令行风格运行
7. 相应RUN、HELP、CLEAR等命令的实现

**数据结构：**

1. Codelist类是一个保存代码的链表。具有清空链表和根据行号将结点插入链表的功能。
2. Token类是保存token类型、变量名和常量整型值的类。
3. Statement是下图多个类的父类。是Parsing之后得到的语句的中间表示。
4. EXP 是下图多个类的父类。是Parsing之后得到的表达式的中间表示。



**设计架构：**

1. MainWindow类作为程序的主窗口，成员变量中有Lexer、Parser、Runner、Command（文本输入框）、Codelist的指针，便于在输入文本和解释过程中利用信号槽传递信息，实现了工程中各个模块的的解耦。
2. Command类继承自QT:: QTextEdit，实现了程序的输入。

通过重载keyPressEvent来实现对RUN、CLEAR、HELP等命令的判定，并向MainWindow发出相应的信号，交给MainWindow调用相应槽函数执行。

如果输入的是一条basic 语句，则向MainWindow发出newLineWritten信号，并把这一行的string也传递给MainWindow。

Mainwindow判断该语句有无行号。有行号则加入codelist，无行号则调用interpret\_a\_line函数。

1. Lexer是词法分析器。首先定义TOKEN\_MAP，然后做字符串的split等操作，对split得到的数组来判断是不是合法的token。同时如果是变量名则进行合法变量名的判断。如果是INT类型的TOKEN,记下INT的值，如果是ID类型的TOKEN,记下变量名。最后得到TOKEN的list。不是合法的token则返回词法错误的信息，由MainWindow调用Command显示错误信息。
2. Parser是语法分析器。对Lexer得到的TOKEN的list进行分析，然后生成Statement。

Parse表达式的时候使用了逆波兰表达式生成EXP TREE，同时对不规范表达式报错。

Mainwindow会设置Parser是有行号/无行号模式。

有行号模式下，Parser根据行号之后的第一个TOKEN进入switch语句的各个case，然后一一匹配相应的TOKEN。匹配成功就得到了STATEMENT,不成功则返回语法错误的信息，由MainWindow调用Command显示错误信息。

无行号模式下只会有LET PRINT INPUT 三种statement生成，不然也是报错。

1. Runner是解释器，对Parser生成Statement解释运行，同时维护一个SYMBOL TABLE，实现对变量的定义和变量值的存取。

由于INPUT需要对Runner进行中断，特此说明其中实现思想：INPUT的时候,先记下当前的行数，然后发出**input\_a\_val**信号，然后返回到MainWindow。

MainWindow接收到**input\_a\_val**信号会调用Command的槽函数，此时Command就知道之后的输入应该是INPUT一个整型数字。输入正确后Command发出信号，同时把整型参数转递过去，Runner的槽函数接收，继续从上次的行数运行。

**Bonus tasks**

* 语法错误提示
* 变量名是否规范
* 语句是否有行号
* 运算表达式是否合法
* 变量是否有定义
* INPUT所需要的输入是否是数字