第二部分：语法分析

1. 内容简介：

在第二部分，我们需要构建一个语法分析器，将获得的词法单元序列构建成一棵抽象语法分析树 (AST)。具体包括解析函数的声明和调用，Tensor 变量的声明以及 Tensor 的二元运算表达式等，并针对非法格式输出错误信息。

2. 功能实现：

开始本部分前请务必先用git pull 最新的repo，同时注意保存之前部分的代码

Tips：

- 在Parser.h搜索"TODO"，可以看到需要实现的相关函数以及具体要求。
- 在处理非法情况时，要求编译器在终端输出尽可能详细的错误信息。
- 在实现具体功能之前，须阅读AST.h，该文件定义了ExprAST类及其各种子类。

文件地址： /pony_compiler/pony/include/pony/Parser.h

要求实现以下功能：

① 解析变量的声明，实现成员函数parseDeclaration()并扩展成员函数parseType()，要求:

1). 语法变量必须以“var”开头，后面为变量名及tensor shape

2). 语法分析器已支持以下两种初始化形式，以一个二维矩阵为例：

- var a = [[1, 2, 3], [4, 5, 6]]
- var a<2,3> = [1, 2, 3, 4, 5, 6]

需要同学们额外支持第三种新的形式：

- var a[2][3] = [1, 2, 3, 4, 5, 6]

② 解析函数内的部分常用表达式，具体要求为：

1). 解析标识符语句，其可以是简单的变量名，也可以用于函数调用。要求实现成员函数 parseIdentifierExpr()

2). 解析矩阵的二元运算表达式，需要考虑算术符号的优先级。要求实现成员函数parseBinOpRHS()

3. 实验验证：

在对语法分析器构建完毕后，可以通过运行测试用例test_8至test_12来检查语法分析器的正确性，本次要求**验证test_8至test_12生成AST(-emit=ast)的正确性，以及test_11编译执行(-emit=jit)的正确性**。

以test_8为例，验证语法分析器功能的正确性，输出AST（-emit=ast）：

```
$ cmake --build . --target pony
$ ./bin/pony ../../test/test_8.pony -emit=ast
```

同学们可以根据输出AST的结构来判断语法分析器功能的正确性。如果执行结果如下图所示，表示语法分析器解析正确。

```
root@e062ec9d43ba:/home/workspace/pony_compiler/src/build# ./bin/pony ../../test/test_8.pony -emit=ast
  Module:
    Function
      Proto 'main' @../../test/test_8.pony:4:1
      Params: []
      Block {
        VarDecl a<> @../../test/test_8.pony:6:3
          Literal: <2, 3>[ <3>[ 1.000000e+00, 2.000000e+00, 3.000000e+00], <3>[ 4.000000e+00, 5.000000e+00, 6.000000e+00]] @../../test/test_8.pony:6:11
        VarDecl b<2, 3> @../../test/test_8.pony:7:3
          Literal: <6>[ 1.000000e+00, 2.000000e+00, 3.000000e+00, 4.000000e+00, 5.000000e+00, 6.000000e+00] @../../test/test_8.pony:7:17
        Print [ @../../test/test_8.pony:8:3
          var: a @../../test/test_8.pony:8:9
        ]
        Print [ @../../test/test_8.pony:9:3
          var: b @../../test/test_8.pony:9:9
        ]
      } // Block
```

同理，以test_9为例，验证语法分析器识别语法错误的功能。

```
$ cmake --build . --target pony
$ ./bin/pony ../../test/test_9.pony -emit=ast
```

结果如下图，显示了错误的语法。

```
root@e062ec9d43ba:/home/workspace/pony_compiler/src/build# ./bin/pony ../../test/test_9.pony -emit=ast
Parse error (5, 7): expected 'identified' after 'var' declaration but has Token 60 '<'
Parse error (5, 7): expected 'nothing' at end of module but has Token 60 '<'
```

此外，对于test_11，使用JIT编译执行，输出执行结果（-emit=jit）：

```
$ cmake --build . --target pony
$ ./bin/pony ../../test/test_11.pony -emit=jit
```

结果如下图，验证正确执行了所定义的矩阵运算。

```
root@e062ec9d43ba:/home/workspace/pony_compiler/src/build# ./bin/pony ../../test/test_11.pony -emit=jit
4.000000 16.000000 36.000000
64.000000 100.000000 144.000000
```