The Assignment

- The assignment will be to write a compiler for a language of arithmetic expressions which we call expressive.
- The compiler should generate code for a small virtual machine.
- You will also implement parts of this machine.

The Language
An Example:

average.exp:

```
print "How many data points?". printnl.
readint n.
sum:= 0. i:= n-1.
while i >= 0 {
  print "Number ". print n-i. printnl.
  readint x.
  sum := sum + x. i := i - 1.
}
average := sum/n.
print "The average is ca:". print average. printnl.
```


The Language
The Grammar

```
Program → Statements
Statements → Statement '.' Statements
  | ε
Statement -> PrintStatement
  | ReadStatement
  | LetStatement
  | CondStatement
  | WhileStatement
```


The Grammar
Statements

```
PrintStatement →
  'print' Expr | 'print' String | 'println'
ReadStatement →
  'readint' Identifier | 'readbool' Identifier
LetStatement → Identifier ':=' Expr
CondStatement → 'if'  Expr  '{' Statements '}'
  |'if'Expr'{'Statements'}''else''{'Statements'}'
WhileStatement → 'while' Expr '{'  Statements '}'
```


The Grammar
Expressions

```
Expr → Comparison
  | Arithmetic
  | BoolExpr
  |'(' Expr ')'
  |Boolean | Integer | Identifier
Comparison → Expr CompOp Expr
Arithmetic →
  Expr Op Expr | UnOp Expr
BoolExpr → Expr BoolOp Expr
```


The Grammar
Tokens

CompOp → **'<'** | **'<='** | **'='** | **'>='** | **'>'** | **'!='**
Op → **'+'** | **'-'** | **'*'** | **'/'** | **'%'** | **'^'**
UnOp → **'!'** | **'-'**
BoolOp → **'|'** | **'&'**
Boolean → **'true'** | **'false'**
Identifier → Letter (Letter | Digit)*
Integer → Digit Digit*
Letter → [a-zA-Z_]
Digit →[0-9]
String → '"' StringChars '"'
StringChars →[^"]*

The assignment

Assignment 1

- Write a scanner for the language Expressive using JLex. The scanner should print all tokens it finds.
- A framework for the assignment is available on the course web page.