Grammar for the Exp programming language

Version 7

program → main

main → (statement)+

statement → st_print | st_attrib | st_if | st_while | st_break | st_continue

st_print → **print (** expression (**,** expression)* **)**

st_attrib → **name =** expression

st_if → **if** comparison **{** (statement)+ (**}** **else** **{** (statement)+ **}**)? **}**

st_while → **while** comparison **{** (statement)+ **}**

st_break → **break**

st_continue → **continue**

comparison → expression (**==** | **!=** | **<** | **<=** | **>** | **>=**) expression

expression → term ((**+** | **-**) term)*

term → factor ((***** | **/** | **%**) factor)*

factor → **number** | **string** | **(** expression **)** | **name** | **read_int ( )** | **read_str ( )**

---

tokens: **+ - * / % ( ) = , { } == != < <= > >=**
number string name read_int read_str
print if else while break continue

bytecode instructions:

+1	**ldc**	*integer or "string"*
-1	**iadd**	
-1	**isub**	
-1	**imul**	
-1	**idiv**	
-1	**irem**	

-1	**istore** *index*	
+1	**iload** *index*	
-1	**astore** *index*	
+1	**aload** *index*	
+1	**getstatic**	
-2	**invokevirtual …/print(I)V**	
-2	**invokevirtual …/print(Ljava/lang/String;)V**	
-1	**invokevirtual …/println()V**	
+1	**invokestatic …/readInt()I**	
-2	**if_icmp??** *label*	**??** → **eq** | **ne** | **lt** | **le** | **gt** | **ge**
0	**goto** *label*	