Please note this is a language like C or Java where whitespaces have no meaning, and whitespace can be inserted between keywords, identifiers, constants, and specials to accommodate programmer style. This grammar does not include formal rules about whitespace because that would add immense clutter.

Recall **epsilon** is the empty string.

<procedure> ::= **procedure ID is** <decl-seq> **begin** <stmt-seq> **end**

<decl-seq> ::= <decl > | <decl><decl-seq>

<stmt-seq> ::= <stmt> | <stmt><stmt-seq>

<decl> ::= <decl-integer> | <decl-record>

<decl-integer> ::= **integer ID ;**

<decl-record> ::= **record ID ;**

<stmt> ::= <assign> | <if> | <loop> | <out>

<assign> ::= **id** <index> **:=** <expr> **;** | **id := new record [** <expr> **];** | **id := record id;**

<index> ::= **[** <expr> **]** | **epsilon**

<out> ::= **out (** <expr> **) ;**

<if> ::= **if** <cond> **then** <stmt-seq> **end**

| **if** <cond> **then** <stmt-seq> **else** <stmt-seq> **end**

<loop> ::= **while** <cond> **do** <stmt-seq> **end**

<cond> ::= <cmpr> | **not** <cond> | <cmpr> **or** <cond> | <cmpr> **and** <cond>

<cmpr> ::= <expr> **=** <expr> | <expr> **<** <expr>

<expr> ::= <term> | <term> **+** <expr> | <term> **–** <expr>

<term> ::= <factor> | <factor> * <term> | <factor> **/** <term>

<factor> ::= **id** | **id [** <expr> **]** | **const** | **(** <expr> **)** | **in ( )**