**­­­­**

PL/0 User’s Guide

Authors: Travis Le and Allysia Freeman

Table of Contents

Table of Contents 2

Table of Figures and Tables 3

1.0: How to Program with PL/0 4

1.1 Data Types 4

1.2 Operators 5

1.3 Expressions 6

1.4 Statements 7

1.5 Procedures 8

2.0: Build, Compile, and Execute PL/0 programs 9

2.1 Building the Compiler 9

2.2 Executing the Compiler 9

3.0: EBNF Grammar of PL/0 10

4.0: Tokens and Reserved Words 11

5.0: Instructions in PL/0 12

6.0: Error Codes 13

Table of Figures and Tables

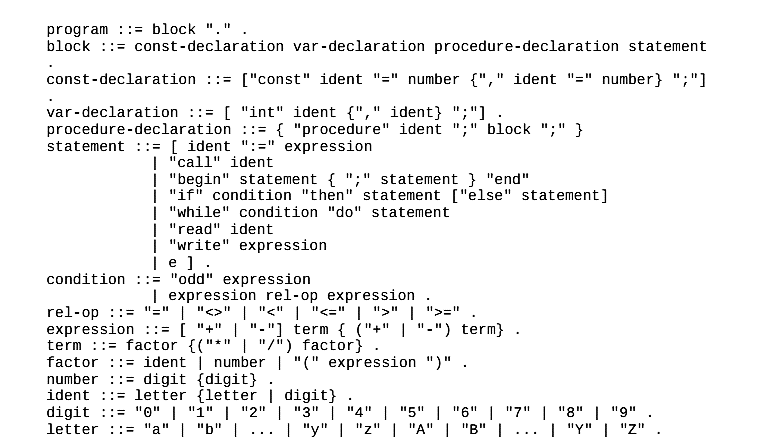
# **1.0: How to Program with PL/0**

* 1. Data Types
     1. **Variable**
     2. **Integer**
     3. **Procedure**
  2. Operators
     1. **Assignment**
     2. **Relational Operators**
     3. **Mathematical Operators**
  3. Expressions
  4. Statements
     1. **Assignments**
     2. **Read and Write**
     3. **While Do**
     4. **If, Then, Else**
  5. Procedures
     1. **Declaration**
     2. **Calling**
     3. **Recursion**
     4. **Nested Procedures**

**2.0: Build, Compile, and Execute PL/0 programs**

1. 1. Building the Compiler
   2. Executing the Compiler

**3.0: EBNF Grammar of PL/0**



Where…

**4.0: Tokens and Reserved Words**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Name** | **Value** | **Usage** |
|  | nulsym |  | n/a |
|  | identsym |  | used for constant, procedure, and variable names |
|  | numbersym |  | used for numbers |
| **+** | plussym |  | adds |
| **-** | minussym |  | subtracts |
| **\*** | multsym |  | multiplies |
| **/** | slashsym |  | divides |
| **odd** | oddsym |  | test if an expression is odd |
| **=** | eqlsym |  | constant definition or check is two expressions are equal |
| **<>** | neqsym |  | test that two expressions do not equal eachother |
| **<** | lessym |  | tests that left expression is less than right expression |
| **<=** | leqsym |  | tests that left expression is less than or equal to right expression |
| **>** | gtrsym |  | tests that left expression is greater than right expression |
| **>=** | geqsym |  | tests that left expression is greater than or equal to right expression |
| **(** | lparentsym |  | begin factor |
| **)** | rparentsym |  | end factor |
| **,** | commasym |  | separates identifiers in declarations |
| **;** | semicolonsym |  | ends a statement |
| **.** | periodsym |  | ends the program |
| **:=** | becomessym |  | assigns a value to a variable |
| **begin** | beginsym |  | begins a block of statements |
| **end** | endsym |  | ends a block of statements |
| **if** | ifsym |  | begins if statement, followed by condition |
| **then** | thensym |  | follows then, followed by statement |
| **while** | whilesym |  | begins while loop, followed by condition |
| **do** | dosym |  | follows while, followed by statement |
| **call** | callsym |  | calls a procedure |
| **const** | constsym |  | begins a constant declaration |
| **var** | varsym |  | begins a variable declaration |
| **procedure** | procsym |  | begins a procedure declaration |
| **write** | writesym |  | prints a value to the screen |
| **read** | readsym |  | asks the user to enter a value |
| **else** | elsesym |  | may follow if statements |

**5.0: Instructions in PL/0**

**6.0: Error Codes**

|  |  |  |
| --- | --- | --- |
| Error Number | Error Message | Explanation |
| **0** | Program is syntactically correct. | n/a |
| **1** | Invalid file input | Make sure the input file is named correctly or edit header.h's filename definitions |
| **2** | Use “=” not “:=” | Don’t use := when you’re not assigning a value to a variable. |
| **3** | Use “:=” not “=” | Always use := when assigning a value to a variable. |
| **4** | “=” expected after const declaration | When declaring a constant, it must be given a value. |
| **5** | Number expected after “=” with const | Constant declarations must be followed by =. |
| **6** | "then" expected after "if" | “then” must follow after an “if” statement. |
| **7** | "do" expected after "while" | “do” must follow after a “while” statement. |
| **8** | const, var, and procedure must be followed by an identifier | You must give a name to every constant, varable, and procedure when declaring it. |
| **9** | “:=” expected after identifier | Missing “:=” after a variable name. |
| **10** | Ident expected after “call” | The proper syntax is “call name;” |
| **11** | Relational operator expected | Missing a relational test. |
| **12** | Assignment to constants and procedures not allowed | You cannot assign a value to procedure names or constants. They are static. |
| **13** | Semicolon needed between statements | Missing a semicolon after a statement. |
| **14** | Cannot begin with this symbol | Check your program’s syntax. |
| **15** | Undeclared variable detected | Declare the variable before using it. |
| **16** | Unclosed parenthesis detected | All parenthesis must be closed. |
| **17** | Invalid operator | See operators section for correct implementation. |
| **18** | Invalid symbol | The symbol is not supported by PL/0. |
| **19** | ";" expected | Missed a “;” in code |
| **20** | Number too long | A number is too long. Make it shorter. |
| **21** | Identifier too long | The name of a variable is too long. Make it shorter. |
| **22** | Generated code too long | Your program is too complicated for the compiler. Shorten or simplify your code. |
| **23** | Compiler has run out of memory | n/a |
| **24** | Period expected. | A period must be at the end of the program. |