Curtin University – Department of Computing

Assignment Cover Sheet / Declaration of Originality

Complete this form if/as directed by your unit coordinator, lecturer or the assignment specification.

Last name:	Beardsmore	Student ID:	15504319
Other name(s):	Connor		
Unit name:	Programming Languages	Unit ID:	COMP2007
Lecturer / unit coordinator:	Stefan Prandl	Tutor:	Stefan Prandl
Date of submission:	03/11/2017	Which assignment?	(Leave blank if the unit has only one assignment.)

I declare that:

- The above information is complete and accurate.
- The work I am submitting is *entirely my own*, except where clearly indicated otherwise and correctly referenced.
- I have taken (and will continue to take) all reasonable steps to ensure my work is *not accessible* to any other students who may gain unfair advantage from it.
- I have *not previously submitted* this work for any other unit, whether at Curtin University or elsewhere, or for prior attempts at this unit, except where clearly indicated otherwise.

I understand that:

- Plagiarism and collusion are dishonest, and unfair to all other students.
- Detection of plagiarism and collusion may be done manually or by using tools (such as Turnitin).
- If I plagiarise or collude, I risk failing the unit with a grade of ANN ("Result Annulled due to Academic Misconduct"), which will remain permanently on my academic record. I also risk termination from my course and other penalties.
- Even with correct referencing, my submission will only be marked according to what I have done myself, specifically for this assessment. I cannot re-use the work of others, or my own previously submitted work, in order to fulfil the assessment requirements.
- It is my responsibility to ensure that my submission is complete, correct and not corrupted.

Signature: _____ Date of signature: 03/11/2017

(By submitting this form, you indicate that you agree with all the above text.)

PL200 Report

Bison and Flex Parser

Connor Beardsmore - 15504319

Curtin University
Science and Engineering
Perth, Australia
November 2017

EBNF Specification

The full EBNF specification for *QUENYALGOL* is as listed below.
This EBNF follows the ISO BNF standard.

⟨*ident*⟩ ::= [a..z] { ⟨*ident*⟩ }

⟨*inumber*⟩ ::= [0..9] { ⟨*number*⟩ }

⟨*id_num*⟩ ::= [⟨*ident*⟩ | ⟨*number*⟩]

⟨*term*⟩ ::= ⟨*id_num*⟩ { ('`*`' | '`/`') ⟨*id_num*⟩ }

⟨*expression*⟩ ::= ⟨*term*⟩ { ('`+`' | '`-`') ⟨*term*⟩ }

⟨*statement_loop*⟩ ::= ⟨*statement*⟩ { '`;`' ⟨*statement*⟩ }

⟨*compound_statement*⟩ ::= '`BEGIN`' ⟨*statement_loop*⟩ '`END`'

⟨*for_statement*⟩ ::= '`FOR`' ⟨*ident*⟩ '`:=`' ⟨*expression*⟩ '`DO`' ⟨*statement_loop*⟩ '`END`' '`FOR`'

⟨*do_statement*⟩ ::= '`DO`' ⟨*statement_loop*⟩ '`WHILE`' ⟨*expression*⟩ '`END`' '`DO`'

⟨*while_statement*⟩ ::= '`WHILE`' ⟨*expression*⟩ '`DO`' ⟨*statement_loop*⟩ '`END`' '`WHILE`'

⟨*if_statement*⟩ ::= '`IF`' ⟨*expression*⟩ '`THEN`' ⟨*statement*⟩ '`END`' '`IF`'

⟨*procedure_call*⟩ ::= '`CALL`' ⟨*ident*⟩

⟨*assignment*⟩ ::= ⟨*ident*⟩ '`:=`' ⟨*expression*⟩

⟨*statement*⟩ ::= ⟨*assignment*⟩
| ⟨*procedure_call*⟩
| ⟨*if_statement*⟩
| ⟨*while_statement*⟩
| ⟨*do_statement*⟩
| ⟨*for_statement*⟩
| ⟨*compound_statement*⟩

⟨*implementation_part*⟩ ::= ⟨*statement*⟩

⟨*function_declaration*⟩ ::= ‘`FUNCTION`’ ⟨*ident*⟩ ‘`;`’ ⟨*block*⟩ ‘`;`’

⟨*procedure_declaration*⟩ ::= ‘`PROCEDURE`’ ⟨*ident*⟩ ‘`;`’ ⟨*block*⟩ ‘`;`’

⟨*specification_part*⟩ ::= {}
| ‘`CONST`’ ⟨*constant_declaration*⟩
| ‘`VAR`’ ⟨*variable_declaration*⟩
| ⟨*procedure_declaration*⟩
| ⟨*function_declaration*⟩

⟨*block*⟩ ::= ⟨*specification_part*⟩ ⟨*implementation_part*⟩

⟨*implementation_unit*⟩ ::= ‘`IMPLEMENTATION`’ ‘`OF`’ ⟨*ident*⟩ ⟨*block*⟩ ‘`.`’

⟨*range*⟩ ::= ⟨*number*⟩ ‘`..`’ ⟨*number*⟩

⟨*array_type*⟩ ::= ‘`ARRAY`’ ⟨*ident*⟩ ‘`[`’ ⟨*range*⟩ ‘`]`’ ‘`OF`’ ⟨*type*⟩

⟨*range_type*⟩ ::= ‘`[`’ ⟨*range*⟩ ‘`]`’

⟨*enumerated_type*⟩ ::= ‘`{`’ ⟨*ident*⟩ { ‘`,`’ ⟨*ident*⟩ } ‘`}`’

⟨*basic_type*⟩ ::= ⟨*ident*⟩
| ⟨*enumerated_type*⟩
| ⟨*range_type*⟩

⟨*type*⟩ ::= ⟨*basic_type*⟩
| ⟨*array_type*⟩

⟨*variable_declaration*⟩ ::= ⟨*ident*⟩ ‘`:`’ ⟨*ident*⟩ { ‘`,`’ ⟨*ident*⟩ ‘`:`’ ⟨*ident*⟩ } ‘`;`’

⟨*constant_declaration*⟩ ::= ⟨*ident*⟩ ‘`=`’ ⟨*number*⟩ { ‘`,`’ ⟨*ident*⟩ ‘`:`’ ⟨*number*⟩ } ‘`;`’

⟨*formal_parameters*⟩ ::= ‘`(`’ ⟨*ident*⟩ { ‘`;`’ ⟨*ident*⟩ } ‘`)`’

⟨*type_declaration*⟩ ::= ‘`TYPE`’ ⟨*ident*⟩ ‘`:`’ ⟨*type*⟩ ‘`;`’

⟨*function_interface*⟩ ::= '`FUNCTION`' ⟨*ident*⟩ [⟨*formal_parameters*⟩]

⟨*procedure_interace*⟩ ::= '`PROCEDURE`' ⟨*ident*⟩ [⟨*formal_parameters*⟩]

⟨*declaration_unit*⟩ ::= '`DECLARATION`' '`OF`' ⟨*ident*⟩ ['`CONST`' ⟨*constant_declaration*⟩] ['`VAR`' ⟨*variable_declaration*⟩] [⟨*type_declaration*⟩] [⟨*procedure_interface*⟩] [⟨*function_interface*⟩]

⟨*basic_program*⟩ ::= ⟨*declaration_unit*⟩ ⟨*implemenetation_unit*⟩

Parser Implementation

bjhbjhb

Lex

sdsdsd

Yacc

sdsdsd

Design Decisions

sdasasdadadsa

References

Levine, John, Tony Mason, and Doug Brown. 1992. *Lex & Yacc*. 2nd. O'Reilly.

Sebesta, Robert W. 2016. *Concepts of Programming Languages*. 11th. USA: Addison-Wesley Publishing Company. ISBN: 0136073476.