Sonoma State University
Computer Science Department
CS460 – Fall 2023 – Watts

Semester Project - Part 2 (aka Project 2)

(Due: Thursday, 16 November 2023, 6:59 am **and** Thursday, 23 November 2023, 6:59 am)

For this project you are to write a recursive descent parser that performs syntactical analysis on PL460 source code using the grammar distributed in class and posted on our website.

Extra Credit Write a meaningful PL460 program that uses all 93 rules in our grammar.
Date Due: Thursday, 16 November 2023, 6:59 am
To turn in: A file called lastnameP2.pl460. Submit your .tgz file by copying it to ~tiawatts/cs460drop.

Specifications

The grammar can be found at: http://watts.cs. sonoma.edu/cs460f23/ProjectGrammar.pdf

Starter files for the assignment can be found in the folder Project2Framework in the course pickup directory. LexicalAnalyzer will generate the tokens identified in the table below.

The functions developed for this parser should be in a .cpp file. Necessary classes, types and prototypes should be in an associated .h file. The files SyntacticalAnalyzer.cpp and SyntacticalAnalyzer.h in Project2Framework should be used as a starting point for this project.

Input: A source code file. The file name should be accepted as a command line argument. The file name extension must be '.pl460'.

Output: A listing of the original source code with lexical and syntactical error messages (written to a listing file (filename - .pl460 + .lst)). A Project 2 file (filename - .pl460 + .p2) containing a list of the rules applied. Each time a rule is used, a line of the form "Using Rule #" should be written to the .p2 file. Sample files are in the P2Tests folder in the course pickup folder. A debugging file (filename - .pl460 + .dbg) containing (perhaps) a list of terminal symbols, non-terminal symbols, rules, and functions encountered while parsing the program (and other useful debugging information). Sample input and output for PL460 programs P2-1.pl460 and P2-2.pl460 are illustrated below.

Your makefile must create an executable called "P2.out".

Your main function must be in a file called Project2.cpp.

Date Due: Thursday, 23 November 2023, 6:59 am
To turn in: A tarred and zipped directory containing source files (headers and implementations) and a makefile. Your directory should be called *lastname*P2 and your tarred and zipped file should be called *lastname*P2.tgz. Submit your .tgz file by copying it to ~tiawatts/cs460drop.

The following tokens are generated by LexicalAnalyzer.o

Identifier	α(α|#|_)*	IDENT_T
Numeric Literals	(+|-|λ) (#⁺ | #*.#⁺ | #⁺.#* | #⁺/#⁺)	NUMLIT_T
String Literals	" . . . "	STRLIT_T
Logical Literals	*#t*	*TRUE_T*
	#f	*FALSE_T*
Key words	*(cad*r)* | *(cd⁺r)* | *(cd*ar)* | *list*	*LISTOP1_T*
	cons | *append*	*LISTOP2_T*
	if	IF_T
	cond	COND_T
	else	ELSE_T
	display	DISPLAY_T
	newline	NEWLINE_T
	and	AND_T
	or	OR_T
	not	NOT_T
	define	DEFINE_T
	let	LET_T
	read	READ_T
Predicates	number?	NUMBERP_T
	list?	LISTP_T
	zero?	ZEROP_T
	null?	NULLP_T
	eof?	EOFP_T
Arithmetic	+	PLUS_T
	-	MINUS_T
	/	DIV_T
	*	MULT_T
	modulo	MODULO_T
	round	ROUND_T
Logical/Relational	=	EQUALTO_T
	>	GT_T
	<	LT_T
	>=	GTE_T
	<=	LTE_T
Other	(	LPAREN_T
	)	RPAREN_T
	‘	SQUOTE_T
		ERROR_T
		EOF_T

```
::::::::::::::
P2-1.pl460
::::::::::::::
(define (main)
     (display "Hello World\n")
)

(main)

::::::::::::::
P2-1.lst
::::::::::::::
Input file: P2-1.pl460
   1: (define (main)
   2:     (display "Hello World\n")
   3: )
   4:
   5: (main)
0 errors found in input file

::::::::::::::
P2-1.p1
::::::::::::::
     LPAREN_T        (
     DEFINE_T        define
     LPAREN_T        (
     IDENT_T         main
     RPAREN_T        )
     LPAREN_T        (
     DISPLAY_T       display
     STRLIT_T        "Hello World\n"
     RPAREN_T        )
     RPAREN_T        )
     LPAREN_T        (
     IDENT_T         main
     RPAREN_T        )
     EOF_T

::::::::::::::
P2-1.p2
::::::::::::::
Using Rule 1
Using Rule 4
Using Rule 20
Using Rule 9
Using Rule 55
Using Rule 7
Using Rule 11
Using Rule 6
Using Rule 3
Using Rule 6
```

```
::::::::::::::
P2-2.pl460
::::::::::::::
(define (square n)
     (* n n)
)

(square 45)
::::::::::::::
P2-2.lst
::::::::::::::
Input file: P2-2.pl460
   1: (define (square n)
   2:     (* n n)
   3: )
   4:
   5: (square 45)
0 errors found in input file
::::::::::::::
P2-2.p1
::::::::::::::
     LPAREN_T        (
     DEFINE_T        define
     LPAREN_T        (
     IDENT_T         square
     IDENT_T         n
     RPAREN_T        )
     LPAREN_T        (
     MULT_T          *
     IDENT_T         n
     IDENT_T         n
     RPAREN_T        )
     RPAREN_T        )
     LPAREN_T        (
     IDENT_T         square
     NUMLIT_T        45
     RPAREN_T        )
     EOF_T
::::::::::::::
P2-2.p2
::::::::::::::
Using Rule 1
Using Rule 4
Using Rule 19
Using Rule 20
Using Rule 9
Using Rule 46
Using Rule 5
Using Rule 8
Using Rule 5
Using Rule 8
Using Rule 6
Using Rule 6
Using Rule 3
Using Rule 5
Using Rule 7
Using Rule 10
Using Rule 6
```