Compiler Design Lab – Mini Project

Context Free Grammar

Constructs:

1) **Do..while**
2) **Switch..case**

Language:

PERL Programming Language

Lexer:

SNO	PATTERN	TOKEN NAME	TOKEN
1.	if,print,do,while,switch,else,case	Keyword or key	<T_key,entryinst>
2.	[$|@][letter|_](letter|_|digit)*	Identifier	<T_id,install_id()>
3.	digit*(\.digit)?([Ee][+-]?digit)?	Number	<T_num,value/entryinst>
4.	> | >= | < | <= | == | !=	Relational operator	<T_relop,operator>
5.	+ | - | * | / | ** | %	Arithmetic operator	<T_arithop,operator>
6.	\n | \t| ’ ‘	Whitespace	<;>
7.	=	Assignment operator	<T_assignop,=>
8.	#| \=.*|\=cut	comment	<;>
9.	(|) | { | } | [|]	;	<(> <)> <{> <}> <[> <]>
10.	.	Binary Operator	<T_binop, .>
11.	#!/usr/bin/perl	Shebang	<;>

Grammar:

P->Shebang S

S -> Declaration;S | Assignment expr;S | do{S} while(cond);S | us; S; switch(arg) {st} S |print “string”;S | B;S | if(cond) {S} | ue;S |ArrayDecl;S|ε

ArrayDecl -> @id=() | @id=(D)

D -> num,D|string,D|num|string

Declaration -> L

B -> $id.$id

L -> L,X|X

X -> $id|Assignment expr

Assignment expr -> $id=E

Cond -> Cond || C | C

C -> C&&D | D

D -> not D | M

M -> (cond)| relexp | true | false

relexp -> relexp relop E|E|id|num

relop -> < | > | <= | >= | == | !=

E -> E+T | E-T | T | ue

T -> T*F | T/F | F

F -> N**F | N

N ->$id | num|(E)

us -> use Switch;

arg -> $id | num

st -> case Y O

Y -> K {S}

K -> num | $id | ”character”

O -> else {S}| st

ue -> $id++ |$id--| ++$id| --$id