Bilkent University | Department of Computer Engineering

CS 315: Programming Languages

Lexical Analyser for a Programming Language for an Integer Language

Language: ***HazardCat***

13.10.2034

Section: 2

Group: 14?

- Musa Yiğit Yayla. 22003108
- Maryam Azimli. 22101528

Instructor:

Aynur Dayanik

1. <main > =: execute begin <program> end
2. <program> :: = <statements>
3. <statements>::= <statement>|< statements> +<statement>
4. < statement>:: =<cond_statement> | <loop> | <single_statement>|<statement>
5. <cond_statement>::= if (<exprs>){<statements>}| else_if (<exprs>){< statements>}|else{<statements>}
6. <loop>::= <for>|<while>
7. <for>::=for (<varName>=<expr>;<cond_statement>){<statements>};
8. <while>::= while (<expr>){statements};
9. <single_statement>::=<varDeclaration>|<return_st>|< arrDec >| <varAssign >| < constIntDecAssign>| < constStringDecAssign >| <varDecAssign>|<func_call>
10. <varDeclaration >::=let<varName>=<expr>;
11. <varAssign >::=<varName>=<number>;
12. < varDecAssign >:= let <varAssign>;
13. < constIntDecAssign >::= const <varName>=<number>;
14. < constStringDecAssign >::=string <varName>=<string>;
15. <return_stat>::=return <exprs>;
16. <arrDec>::= list <varName >;
17. <arraySizeSpecifier >::= ~
18. <func_call>::= func varName(parameters){<statements><return_st>}
19. <exprs>::= <expr>+<expr>|<expr>*<expr>|expr;
20. <expr>::=<varName>|<varName>+<expr>|<constant>;
21. <varName>::=<lower_lets>|<lower_lets>+<upper_let>|<lower_lets>+<upper_lets>+<nums>;
22. <parameters>::=int<varName>|int<varName>,<parameters>

23.<read>::=read <read_sign> <exprs>;

24.<read_sign>::= <<

25.<write>::= write <write_sign> <exprs>;

26.<write_sign>::= >>

27.<string>::=<lower_lets>|upper_lets>|< digits>|<special_buts>

28.<lower_lets>::= a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z |<lower_lets>

29.<upper_let>::=
A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|<upper_let>

30.<special_buts>::=/|.|,|*|&|^|%|$|#|@!|~

31.<compare>::=< | > | <= | >= | == | !=

32.<increment>::=<varName>++|++<varName>|++

33.<decrement>::=<varName>--|--<varName>|--

34.<plusEqual><varName>+=

35.<minusEqual><varName>-=

36.<negative>::=-(number)

37.<arithmeticOperator >::= <sum>| <subtract>| <multiply>| <divide>| <mod> |
<pow>

38.<sum>::=<varName>+<varName>|<varName>+<constant>|<constant> +
<constant>

39.<subtract>::=<varName>-<varName>|<varName>-<constant>|
<constant> - <constant>

40.<multiply>::=<varName>*<varName>|<varName>*<constant>|
<constant> * <constant>

41.<divide>::=<varName>/<varName>|<varName>/<constant>|
<constant> / <constant>

42.<mod>::=<varName>%<varName>|<varName>%<constant>|
<constant> % <constant>

43. <pow>::=<varName>^<varName>|<varName>^<constant>|
    <constant> ^ <constant>
44. <boolOperator>::= <not>|<or>|<and>|<xor>
45. <not>::=!
46. <or>::= ||
47. <and>::=&&
48. <xor>::= ^^
49. <number>::=<digits><digits>|<digits>| <negative><digits>|<negative>
50. <digits>::=0|1|2|3|4|5|6|7|8|9
51. <comments>::= #<string>
52. <print>::=print(<number>|<expr>|<string>)
53. <print_line>::=print_line+<print>