

CP471

Assignment 2

Austin Bursey | 160165200

Due: Feb 14th 2020

1 Grammar

$\langle \text{program} \rangle ::= \langle \text{fdecls} \rangle \langle \text{declarations} \rangle \langle \text{statement_seq} \rangle.$
 $\langle \text{fdecls} \rangle ::= \langle \text{fdec} \rangle ; \langle \text{fdecls} \rangle \mid \epsilon$

$\langle \text{fdec} \rangle ::= \text{def } \langle \text{type} \rangle \langle \text{fname} \rangle (\langle \text{params} \rangle) \langle \text{declarations} \rangle \langle \text{statement_seq} \rangle$
 fed
 $\langle \text{params} \rangle ::= \langle \text{type} \rangle \langle \text{var} \rangle \langle \text{params-rest} \rangle \mid \epsilon$
 $\langle \text{params-rest} \rangle ::= , \langle \text{params} \rangle \mid \epsilon \mid$
 $\langle \text{fname} \rangle ::= \langle \text{id} \rangle$

$\langle \text{declarations} \rangle ::= \langle \text{decl} \rangle ; \langle \text{declarations} \rangle \mid \epsilon \mid$

$\langle \text{decl} \rangle ::= \langle \text{type} \rangle \langle \text{varlist} \rangle$
 $\langle \text{type} \rangle ::= \text{int} \mid \text{double}$
 $\langle \text{varlist} \rangle ::= \langle \text{var} \rangle \langle \text{varlist-rest} \rangle$
 $\langle \text{varlist-rest} \rangle ::= , \langle \text{varlist} \rangle \mid \epsilon$

$\langle \text{statement_seq} \rangle ::= \langle \text{statement} \rangle \langle \text{statement_seq-rest} \rangle$
 $\langle \text{statement_seq-rest} \rangle ::= ; \langle \text{statement_seq} \rangle \mid \epsilon \mid$

$\langle \text{statement} \rangle ::= \langle \text{var} \rangle = \langle \text{expr} \rangle \mid$
 $\text{if } \langle \text{bexpr} \rangle \text{ then } \langle \text{statement_seq} \rangle \langle \text{statement-rest} \rangle \text{fi} \mid$
 $\text{while } \langle \text{bexpr} \rangle \text{ do } \langle \text{statement_seq} \rangle \text{od} \mid$
 $\text{print } \langle \text{expr} \rangle \mid$
 $\text{return } \langle \text{expr} \rangle \mid \epsilon$

$\langle \text{statement-rest} \rangle ::= \text{else } \langle \text{statement_seq} \rangle \mid \epsilon$

$\langle \text{expr} \rangle ::= \langle \text{term} \rangle \langle \text{expr-rest} \rangle$
 $\langle \text{expr-rest} \rangle ::= + \langle \text{term} \rangle \langle \text{expr-rest} \rangle \mid - \langle \text{term} \rangle \langle \text{expr-rest} \rangle \mid \epsilon$

$\langle \text{term} \rangle ::= \langle \text{factor} \rangle \langle \text{term-rest} \rangle$
 $\langle \text{term-rest} \rangle ::= * \langle \text{factor} \rangle \langle \text{term-rest} \rangle \mid / \langle \text{factor} \rangle \langle \text{term-rest} \rangle \mid \% \langle \text{factor} \rangle$
 $\langle \text{term-rest} \rangle \mid \epsilon$
 $\langle \text{factor} \rangle ::= \langle \text{id} \rangle \langle \text{factor-rest} \rangle \mid \langle \text{number} \rangle \mid (\langle \text{expr} \rangle)$
 $\langle \text{factor-rest} \rangle ::= \langle \text{var-rest} \rangle (\langle \text{exprseq} \rangle)$
 $\langle \text{exprseq} \rangle ::= \langle \text{expr} \rangle \langle \text{exprseq-rest} \rangle \mid \epsilon$
 $\langle \text{exprseq-rest} \rangle ::= , \langle \text{exprseq} \rangle \mid \epsilon \mid$

$\langle \text{bexpr} \rangle ::= \langle \text{bterm} \rangle \langle \text{bexpr-rest} \rangle$
 $\langle \text{bexpr-rest} \rangle ::= \text{or } \langle \text{bterm} \rangle \langle \text{bexpr-rest} \rangle \mid \epsilon$

$\langle \text{bterm} \rangle ::= \langle \text{bfactor} \rangle \langle \text{bterm-rest} \rangle$
 $\langle \text{bterm-rest} \rangle ::= \text{and } \langle \text{bfactor} \rangle \langle \text{bterm-rest} \rangle \mid \epsilon$

$\langle \text{bfactor} \rangle ::= (\langle \text{bfactor-rest} \rangle) | \text{not } \langle \text{bfactor} \rangle |$
 $\langle \text{bfactor-rest} \rangle ::= \langle \text{bexpr} \rangle | \langle \text{expr} \rangle \langle \text{comp} \rangle \langle \text{expr} \rangle$
 $\langle \text{bfactor-rest-rest} \rangle ::=$
 $\langle \text{comp} \rangle ::= < | > | == | <= | >= | < >$

 $\langle \text{var} \rangle ::= \langle \text{id} \rangle \langle \text{var-rest} \rangle$
 $\langle \text{var-rest} \rangle ::= [\langle \text{expr} \rangle] | \epsilon$

 $\langle \text{letter} \rangle ::= \text{a} | \text{b} | \text{c} | \dots | \text{z}$
 $\langle \text{digit} \rangle ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0$
 $\langle \text{id} \rangle ::= \langle \text{letter} \rangle \langle \text{id-rest} \rangle$
 $\langle \text{id-rest} \rangle ::= \langle \text{letter} \rangle \langle \text{id-rest} \rangle | \langle \text{digit} \rangle \langle \text{id-rest} \rangle | \epsilon$

 $\langle \text{number} \rangle ::= \langle \text{integer} \rangle | \langle \text{double} \rangle$

 $\langle \text{integer} \rangle ::= \langle \text{digit} \rangle \langle \text{int shortcut} \rangle$
 $\langle \text{int shortcut} \rangle ::= \langle \text{e-int} \rangle | \langle \text{digit} \rangle \langle \text{int shortcut} \rangle | . \langle \text{lead-off} \rangle \langle \text{e-int} \rangle$
 $| \epsilon$

 $\langle \text{e-int} \rangle ::= \text{e} \langle \text{e-rest-int} \rangle \langle \text{lead-off} \rangle | \epsilon$
 $\langle \text{double} \rangle ::= \langle \text{digit} \rangle \langle \text{double shortcut} \rangle$
 $\langle \text{double shortcut} \rangle ::= . \langle \text{lead-off} \rangle \langle \text{e-double} \rangle | \langle \text{digit} \rangle \langle \text{double shortcut} \rangle |$
 $\langle \text{e-double} \rangle | \langle \text{lead-off} \rangle | \epsilon$

 $\langle \text{e-double} \rangle ::= \text{e} \langle \text{e-rest-int} \rangle \langle \text{lead-off} \rangle | \epsilon$
 $\langle \text{lead-off} \rangle ::= \langle \text{digit} \rangle \langle \text{lead-off-rest} \rangle$
 $\langle \text{lead-off-rest} \rangle ::= \langle \text{lead-off} \rangle | \epsilon$

 $\langle \text{e-rest-int} \rangle ::= + | \epsilon$

2 First

$\text{first}(\langle \text{program} \rangle) = \{\text{def, int, double, a, b, c, ... , z, if, while, print, return, .} \}$
 $\text{first}(\langle \text{fdecls} \rangle) = \text{first}(\langle \text{fdec} \rangle) = \{\text{def, } \epsilon \}$

 $\text{first}(\langle \text{declarations} \rangle) = \{\text{int, double, } \epsilon \}$

 $\text{first}(\langle \text{params} \rangle) = \{\text{int, double, } \epsilon \}$
 $\text{first}(\langle \text{type} \rangle) = \text{first}(\langle \text{decl} \rangle) = \{\text{int, double} \}$

 $\text{first}(\langle \text{params-rest} \rangle) = \{', ', \epsilon \}$

 $\text{first}(\langle \text{fname} \rangle) = \text{first}(\langle \text{id} \rangle) = \text{first}(\langle \text{letter} \rangle) = \text{first}(\langle \text{varlist} \rangle) = \text{first}(\langle \text{var} \rangle) =$
 $\{\text{a, b, c, ... , z} \}$

$\text{first}(\langle \text{comp} \rangle) = \{<, >, ==, <=, >=, < >\}$

$\text{first}(\langle \text{varlist-rest} \rangle) = \{', ', \epsilon\}$

$\text{first}(\langle \text{statement_seq} \rangle) = \text{first}(\langle \text{statement} \rangle) = \{\text{if, while, print, return, a, b, c, ... , z, } \epsilon\}$

$\text{first}(\langle \text{statement_seq} - \text{rest} \rangle) = \{;, \epsilon\}$

$\text{first}(\langle \text{statement-rest} \rangle) = \{\text{else}, \epsilon\}$

$\text{first}(\langle \text{exprseq} \rangle) = \{ (, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, \text{a}, \text{b}, \text{c}, \dots, \text{z}, \epsilon \}$

$\text{first}(\langle \text{expr} \rangle) = \text{first}(\langle \text{term} \rangle) = \text{first}(\langle \text{factor} \rangle) = \text{first}(\langle \text{factor-rest} \rangle) = \{ (, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, \text{a}, \text{b}, \text{c}, \dots, \text{z} \}$

$\text{first}(\langle \text{expr-rest} \rangle) = \{+, -, \epsilon\}$

$\text{first}(\langle \text{term-rest} \rangle) = \{*, /, \%, \epsilon\}$

$\text{first}(\langle \text{exprseq-rest} \rangle) = \{', ', \epsilon\}$

$\text{first}(\langle \text{bexpr} \rangle) = \text{first}(\langle \text{bterm} \rangle) = \text{first}(\langle \text{bfactor} \rangle) = \{ (, \text{not} \}$

$\text{first}(\langle \text{bexpr-rest} \rangle) = \{\text{or}, \epsilon\}$

$\text{first}(\langle \text{bterm-rest} \rangle) = \{\text{and}, \epsilon\}$

$\text{first}(\langle \text{bfactor-rest} \rangle) = \{ (, \text{not}, \text{a}, \text{b}, \text{c}, \dots, \text{z}, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0 \}$

$\text{first}(\langle \text{var-rest} \rangle) = \{[, \epsilon\}$

$\text{first}(\langle \text{digit} \rangle) = \text{first}(\langle \text{number} \rangle) = \text{first}(\langle \text{integer} \rangle) = \text{first}(\langle \text{double} \rangle) = \text{first}(\langle \text{lead-off} \rangle) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 0\}$

$\text{first}(\langle \text{id-rest} \rangle) = \{\text{a}, \text{b}, \text{c}, \dots, \text{z}, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, \epsilon\}$

$\text{first}(\langle \text{int shortcut} \rangle) = \{\text{e}, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, ., \epsilon\}$

$\text{first}(\langle \text{double shortcut} \rangle) = \{., 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, \text{e-}, \epsilon\}$

$\text{first}(\langle \text{lead-off-rest} \rangle) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 0, \epsilon\}$

$\text{first}(\langle \text{e-rest-int} \rangle) = \{+, \epsilon\}$

$\text{first}(\langle \text{e-int} \rangle) = \{\text{e}, \epsilon\}$

$\text{first}(\langle \text{e-double} \rangle) = \{\text{e-}, \epsilon\}$

3 Follow

follow(<program>)= {.}

follow(<fdecls>)= { int , double,.}

follow(<declarations>)= {if, while, print, return, a , b , c , ... , z, . }

follow(<fdec>) = {;}

follow(<statement_seq>)=follow(statement_seq-rest) = { . , fed , fi ,od,else }

follow(<type>)= { a , b , c , ... , z}

follow(<fname>)= { (}

follow(<params>)= follow(<params-rest>) = {) }

follow(<var>) = { " , ,) , ; , * , / , % , + , - , ; , ' , ' ,) ,] , < , > , == , <= , >= , <> , = }

follow(<decl>) = {;}

follow(<varlist>) = follow(<varlist-rest>)= {;,) }

follow(<id>)=follow(<id-rest>) { (, [, " , " ,) , ; , * , / , % , + , - ,] , < , > , == , <= , >= , <> , = , , . , fed , fi ,od,else }

follow(<statement>)= {;, . , fed , fi ,od,else}

follow(<expr>)= follow(<expr-rest>)= {;, ' , ' ,) ,] , < , > , == , <= , >= , <> , . , fed , fi ,od,else }

follow(<term>)=follow(<term-rest>)= { + , - , ; , ' , ' ,) ,] , < , > , == , <= , >= , <> , . , . , fed , fi ,od,else}

follow(<factor>)=follow(<factor-rest>)=follow(<number>)= { * , / , % , + , - , ; , ' , ' ,) ,] , < , > , == , <= , >= , <> , . , . , fed , fi ,od,else }

follow(<exprseq>)=follow(<exprseq-rest>)= { }

follow(<bexpr>)=follow(<bexpr-rest>)= { then , do ,) }

follow(<bterm>)=follow(<bterm-rest>){then, do ,) ,or }

follow(<bfactor>) = {then, do ,) ,or , and }

follow(<bfactor-rest>) = { }

follow(<comp>)= { (,1,2,3,4,5,6,7,8,9,0, a , b , c , ... , z }

follow(<letter>)= { (, [, " ,) , ; , * , / , % , + , - ,] , < , > , = , = , < = , > = , < > , = , a , b , c , ... , z , 1,2,3,4,5,6,7,8,9,0, . , fed , fi ,od,else }

follow(<double>)=follow(<e-double>)=follow(<double-shortcut>)=follow(<integer>)=follow(<int-shortcut>)=follow(<e-int>)= { * , / , % , + , - , ; , ' , ' ,) ,] , < , > , = , = , < = , > = , < > , . , fed , fi ,od,else }

follow(<lead-off>)=follow(<lead-off-rest>)= { * , / , % , + , - , ; , ' , ' ,) ,] , < , > , = , = , < = , > = , < > , e , 1,2,3,4,5,6,7,8,9,0 }

follow(<digit>)= { * , / , % , + , - , ; , ' , ' ,) ,] , < , > , = , = , < = , > = , < > , e , 1,2,3,4,5,6,7,8,9,0, . , fed , fi ,od,else }

follow(<e-rest-int>)= { 1,2,3,4,5,6,7,8,9,0, (, [, " ,) , ; , }