(a)

First (<program>): { fun }

First (<top\_lvl\_stmt>): { fun }

First (<params>): { in, boo, big, small }

First (<param>): { in, boo, big, small }

First (<stmts>): { if, while, <letter>, in, boo, big, small, <digit>,true, false, NOT, ( }

First (<stmt>): { if, while, <letter>, in, boo, big, small, <digit>,true, false, NOT, ( }

First (<declaration>): { in, boo, big, small }

First (<assignment>): { <letter>, in, boo, big, small }

First (<conditional>): { if, while }

First (<elfears>): { elf( , else }

First (<expression>): { <letter>, <digit>,true, false, NOT, ( }

First (<op>): { -, + , \*, / , %, < ,>, <=, >=, =, !=, AND, OR }

First (<literals>): { <digit>,true, false }

First (<ident>): { <letter> }

First (<type>): { in, boo, big, small }

Follow (<program>): { $ }

Follow (<top\_lvl\_stmt>): { <endl> , $}

Follow (<params>): { ) }

Follow (<param>): { “,” , ) }

Follow (<stmts>): { elf, endwhile , endfun, $ , else, endif }

Follow (<stmt>): { <endl>, elf, endwhile , endfun, $, else, endif }

Follow (<declaration>): { <endl>, elf, endwhile , endfun, $, else, endif, ← }

Follow (<assignment>): { <endl>, elf, endwhile , endfun, $, else, endif }

Follow (<conditional>): { <endl>, elf, endwhile , endfun, $, else, endif }

Follow (<elfears>): { endif }

Follow (<expression>): { ) , -, + , \*, / , %, < ,>, <=, >=, =, !=, AND, OR, $ ,<endl>, elf, endwhile , endfun, else, endif }

Follow (<op>): { <digit> , <letter> , NOT , (, true, false }

Follow (<literals>): { ) , -, + , \*, / , %, < ,>, <=, >=, =, !=, AND, OR, $ ,<endl>, elf, endwhile , endfun, else, endif }

Follow (<ident>): { ( , “,” , <endl>, elf, endwhile , endfun, $, else, endif, ← }

Follow (<type>): { <endl>, <letter> }

(b)

Predict (<program> -> <top\_lvl\_stmt>): { fun }

Predict (<top\_lvl\_stmts> -> <top\_lvl\_stmt> <endl> <top\_lvl\_stmt>): { fun }

Predict (<top\_lvl\_stmt> -> fun <ident> (<params>) as <type> \n <stmts> endfun): { fun }

Predict (<params> -> <param> <paramlist>): { in, boo, big, small }

Predict (<params> -> λ): { λ }

Predict(<paramlist>-> <comma> <param> <paramlist>){<comma>}

Predict(<paramlist>-> λ ): { λ }

Predict (<param> -> <type> <ident>): { in, boo, big, small }

Predict (<stmts> -> <stmt> <morestmts>): { if, while, <letter>, in, boo, big, small, <digit>,true, false, NOT, ( }

Predict (<morestmts> -> <endl> <stmts>): { <endl>}

Predict (<morestmts> -> λ){<endfun>, <endwhile>, <elf> , <else> }

Predict (<stmt> -> <conditional>): { if, while }

Predict (<stmt> -> λ ): { λ }

Predict (<stmt> -> toss <expression>): { toss }

Predict (<stmt> -> <varhandler>): { <letter>, in, boo, big, small }

Predict (<varhandler> -> <ident> <assignment>): { <letter>}

Predict (<varhandler> -> <declaration> <catassign>): { in, boo, big, small }

Predict (<declaration> -> <type> <ident>): { in, boo, big, small }

Predict (<assignment> -> <- <toplevelexpression> ): { <- }

Predict (<catassign> -> <assignment> ): { <- }

Predict (<catassign> -> λ ): { λ }

Predict (<conditional> -> if (<toplevelexpression>) <endl> <stmts> <elfears> endif): { if }

Predict (<conditional> -> while (<expression>) \n <stmts> endwhile): { while }

Predict (<elfears> -> elf( <expression> ) <stmts> <elfears> ): { elf }

Predict (<elfears> -> else <stmts> ): { else }

Predict (<elfears> -> λ ): { λ }

Predict (<topexpression> -> <expression><expressionlist>): { <letter> }

Predict (<topexpression> -> (<expression><expressionlist>) <expressionlist>):{ ( }

? Predict (<expressionlist> -> <op> <topexpression>): { <op> }

Predict (<expressionlist > -> λ ): { λ }

Predict (<expression> -> <ident> <funcall>): { <letter> }

Predict (<expression> -> <literals>): { <digit>, true, false }

Predict (<expression> -> NOT (<topexpression>)): { NOT }

Predict ( <funcall> -> ( <params> ) ): { ( }

Predict ( <funcall> -> λ ): { λ }

Predict (<sop>-> +): { + }

Predict (<sop>-> -): { - }

Predict (<fop>-> \*): { \* }

Predict (<fop>-> /): { / }

Predict (<fop>-> %): { % }

Predict (<fop>-> <): { < }

Predict (<fop>-> >): { > }

Predict (<fop>-> <=): { <= }

Predict (<fop>-> >=): { >= }

Predict (<fop>-> =): { = }

Predict (<fop>-> !=): { != }

Predict (<fop>-> AND): { AND }

Predict (<fop>-> OR): { OR }

Predict (<literals> -> <digit><digit>\*): { <digit> }

Predict (<literals> -> true): { true }

Predict (<literals> -> false): { false }

Predict (<ident> -> <letter><letter|digit|underscore>\*): { <letter> }

Predict (<type> -> in): { in }

Predict (<type > -> boo): { boo }

Predict (<type> -> big): { big }

Predict (<type > -> small): { small }