(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 (<val>): { ? }

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, $ }

Follow (<stmt>): { \n, elf(, endwhile , endfun, $ }

Follow (<declaration>): { $ }

Follow (<assignment>): { $ }

Follow (<conditional>): { $ }

Follow (<elfears>): { endif , $ }

Follow (<expression>): { ) , -, + , \*, / , %, < ,>, <=, >=, =, !=, AND, OR, $ }

Follow (<val>): { }

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

Follow (<literals>): { $ }

Follow (<ident>): { ( , [ , <- }

Follow (<type>): { \n, <letter> }

(b)

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

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

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

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

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

Predict ((<params> -> λ): { λ }

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

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

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

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

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

Predict (<stmt> -> < expression >): { <letter>, <digit>,true, false, NOT, ( }

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

Predict (<stmt> -> λ ): { λ }

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

Predict (<declaration> -> <type> <ident> [<digit><digit>\*] ): { in, boo, big, small }

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

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

Predict (<assignment> -> < type> <ident>[<digit><digit>\*]<- <expression >): { in, boo, big, small }

Predict (<conditional> -> if (<expression>) \n <stmts> <elfears> endif ): { if }

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

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

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

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

Predict (<elfears> -> λ ): { λ }

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

Predict (<expression> -> <ident> [<digit><digit>\*]): { <letter>}

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

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

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

Predict (<expression> -> < expression> <op> <expression>): { <letter>, <digit>,true, false, NOT, ( }

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

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

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

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

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

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

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

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

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

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

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

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

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

Predict (<op>-> 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 }