First(program) = {void, int}

First(decl) = {void, int}

First(decl’) = { [, ) }

First(var-decl) = { int }

First(fun-decl) = { ( }

First(params) = { int, void }

First(param) = { int }

First(compound-stmt) = { { }

First(local-decl) = { int, epsilon }

First(statement-list) = { (, NUM, ID, {, if, while, return, epsilon }

First(statement) = { (, NUM, ID, {, if, while, return }

First(expression-stmt) = { (, NUM, ID, ; }

First(selection-stmt) = { if }

First(iteration-stmt) = { while }

First(return-stmt) = { return }

First(expression) = { (, NUM, ID }

First(expression’) = { =, \*, /, [ }

First(expression’’) = { =, \*, / }

First(relop) = { <, >, =, ! }

First(additive-expression) = { (, NUM, ID }

First(term) = { (, NUM, ID }

First(factor) = { (, NUM, ID }

First(simple-expression’) = { \*, / }

First(additive-expression’) = { \*, / }

First(term’) = { \*, / }

First(mulop) = { \*, / }

First(addop) = { +, - }

First(args) = { (, NUM, ID, epsilon }

Follow(program) = {$}

Follow(decl) = {void, int

Follow(decl’) = {

Follow(var-decl) = {int, “(“, NUM, ID, “{“, if, while, return, epsilon,”}”

Follow(fun-decl) = {

Follow(params) = {“)” }

Follow(param) = {“,”, “)”}

Follow(compound-stmt) = {

Follow(local-decl) = {“(“, NUM, ID, “{“, if, while, return, “}” }

Follow(statement-list) = {“}”}

Follow(statement) = {“}”

Follow(expression-stmt) = {

Follow(selection-stmt) = {

Follow(iteration-stmt) = {

Follow(return-stmt) = {

Follow(expression) = {“)”, “(“, NUM, ID, if, while, return, “]”,

Follow(expression’) = {

Follow(expression’’) = {

Follow(relop) = {“(“, NUM, ID,

Follow(additive-expression) = {“<”, “>”, “=”, “!”

Follow(term) = {\*, /, +, -}

Follow(factor) = {\*, /}

Follow(var-call’) = {

Follow(simple-expression’) = {

Follow(additive-expression’) = {<, >, =, !,

Follow(term’) = {+, -, \*, /

Follow(addop) = { (, NUM, ID

Follow(mulop) = { (, NUM, ID

Follow(args) = { ),