Rule	First	Follow
Program = decls “EOF”	Int, bool, void, EOF	$
decls = typ “id” decls_prime | epsilon	Int, bool, void, epsilon	EOF
decls_prime = vdecl decls | fdecl decls	Semi, lparen	EOF
Fdecl = “lparen” formals_opt “rparen” “LBRACE” vdecl_list stmt_list “RBRACE”	lapren	Int, bool, void, EOF
formals_opt = formal_list | epsilon	Int, bool, void, epsilon	rparen
formal_list = typ “ID” formal_list_prime	Int, bool, void	rparen
Formal_list_prime = “COMMA” formal_list | epsilon	Comma, epsilon	rparen
typ = “INT” | “BOOL” | “VOID”	Int, bool, void	Id
vdecl_list = vdecl vdecl_list | “epsilon”	Epsilon, semi	RETURN, LBRACE, IF, FOR, WHILE, INTLITERAL, TRUE, FALSE, MINUS, NOT, ID, LPAREN, RBRACE
vdecl = “SEMI”	semi	“INT”, “BOOL”, “VOID”, “EOF”, “SEMI”, “RETURN”, RETURN, “LBRACE”, IF, FOR, WHILE, INTLITERAL, TRUE, FALSE, MINUS, NOT, ID, LPAREN, “RBRACE”
stmt_list = stmt stmt_list | epsilon	ε , “RETURN”, RETURN, “LBRACE”, IF, FOR, WHILE, INTLITERAL, TRUE, FALSE, MINUS, NOT, ID, LPAREN, SEMI	RBRACE
stmt = expr SEMI | “RETURN” SEMI | RETURN expr SEMI | “LBRACE”	RETURN, “LBRACE”, IF, FOR, WHILE,	RETURN, LBRACE, IF, FOR, WHILE,

stmt_list RBRACE | IF LPAREN expr RPAREN stmt stmt_prime | FOR LPAREN expr_opt SEMI expr SEMI expr_opt RPAREN stmt | WHILE LPAREN expr RPAREN stmt	INTLITERAL, TRUE, FALSE, MINUS, NOT, ID, LPAREN, SEMI	INTLITERAL, TRUE, FALSE, MINUS, NOT, ID, LPAREN, RBRACE, NOELSE, ELSE, RBRACE
Stmt_prime = NOELSE | ELSE stmt	NOELSE, ELse	RETURN, “LBRACE”, IF, FOR, WHILE, INTLITERAL, TRUE, FALSE, MINUS, NOT, ID, LPAREN, “RBRACE”, NOELSE, ELSE, RBRACE
Expr_opt = expr | “epsilon”	INTLiteral, true, false, id, minus, not, lparen, epsilon	Semi, rparen
Expr = INTLITERAL expr_prime | TRUE expr_prime | FALSE expr_prime | MINUS expr NEG expr_prime | NOT expr expr’ | ID expr_prime_prime | LPAREN expr RPAREN expr_prime	INTLiteral, true, false, id, minus, not, lparen	Semi, rparen, Plus, minus, times, divide, EQ, NEQ, LT, LEQ, GT, GEQ, AND, OR, COMMA
expr_prime = PLUS expr expr_prime | MINUS expr expr_prime | TIMES expr expr’ | DIVIDE expr expr_prime | EQ expr expr_prime | NEQ expr expr_prime | LT expr expr_prime | LEQ expr expr_prime | GT expr expr_prime | GEQ expr expr_prime | AND expr expr_prime | OR expr expr_prime | epsilon	Plus, minus, times, divide, EQ, NEQ, LT, LEQ, GT, GEQ, AND, OR, epsilon	SEMI, RPAREN, NEG, PLUS, MINUS, TIMES, DIVIDE, EQ, NEQ, LT, LEQ, GT, GEQ, AND, OR, COMMA
Expr_prime_prime = expr_prime | ASSIGN expr expr_prime | LPAREN actuals_opt RPAREN expr_prime	Plus, minus, times, divide, EQ, NEQ, LT, LEQ, GT, GEQ, AND, OR, epsilon, Assign, Lparen	SEMI, RPAREN, NEG, PLUS, MINUS, TIMES, DIVIDE, EQ, NEQ, LT, LEQ, GT, GEQ, AND, OR, COMMA
Actuals_opt = actuals_list | epsilon	INTLiteral, true, false, id, minus, not, lparen, epsilon	rparen
actuals_list = expr actuals_list_prime	intLiteral, true, false, id, minus, not, lparen	rparen
Actuals_list_prime = COMMA expr actuals_list_prime | epsilon	COMMA, epsilon	rapren