Machine Translated by Google

Compiler Design Principles Lesson Project

Phase 2: Converting syntax grammars into code

```
Main ÿ syntax | assignment | $

If ÿanother_clause (cond) {Main} if | end {Main} | ÿ

syntax ÿ clause (cond) {Main} if Main | f-loop (from number to number by cond) {Main} Main | w-loop (from
number to number) {Main} Main | Define ID (arg) {Main} Main | ID(argplus)
arg ÿ ID arg | ÿ
argplus ÿ ID argplus | number argplus | ÿ
comparison ÿ < | > |value|
cond ÿ ID comparison ID | ID comparison number number comparison ID | number comparison
number| true | false

operation ÿ |add| | |minus| | |multi| | |div| | |set|

operator ÿ ID operation ID | ID operation number operation number Operation ID number


stmnt ÿ ID = ID | ID = number| ID = operator

assignment ÿ integer stmnt | float stmnt | string stmnt
```

In non-cond, operator, and stmnt terminals, we have a first/first problem.

Our grammars after solving the first / first problem are as follows:

```
Main ÿ syntax | assignment|$

If ÿanother_clause (cond) {Main} if | end {Main} | ÿ

syntax ÿ clause (cond) {Main} if Main | f-loop (from number to number by cond) {Main} Main | w-loop (from
number to number) {Main} Main | Define ID (arg) {Main} Main | ID(argplus) Main
arg ÿ ID arg | ÿ
argplus ÿ ID argplus | number argplus | ÿ
comparison ÿ < | > |value|

cond ÿ ID comparison Y | number comparison Y | true | false
operation ÿ |add| | |minus| | |multi| | |div| | |set|

operator ÿ ID operation Y | operation number Y
Y ÿ ID | number

stmnt ÿ ID = L
L ÿ IDZ | numberZ
Z ÿ operation Y | ÿ

assignment ÿ integer stmnt | float stmnt | string stmnt
```