Compiladores
Laboratorio 12

Objetivo

Elaborar el intérprete de una gramática con sentencias de control repetitivo y operaciones booleanas utilizando el patrón de diseño Visitor.

Programa
Se tiene implementado

- Program ::= StmtList
- StmtList ::= Stmt **(;** Stmt **)***
- Stmt ::= **id =** CExp |
  print (CExp **) |**
  if CExp **then** StmtList [**else** StmtList] **endif |**
  while CExp **do** StmtList **endwhile |**
- CExp ::= Exp [(**<**|**<=**|**==**) Exp]
- Exp ::= Term ((**+** | **-**) Term)*
- Term ::= Factor ((***|**/**) Factor)*
- Factor ::= **id** | **Num** | **(** AExp **)** | **ifexp (**AExp **,** AExp **,** AExp **)**

Problema 1
Implementar la gramática:

- Program ::= StmtList
- StmtList ::= Stmt **(;** Stmt **)***
- Stmt ::= **id =** AExp |
  print (AExp **) |**
  if AExp **then** StmtList [**else** StmtList] **endif |**
  while AExp **do** StmtList **endwhile |**
  for(AExp **,** AExp **,** AExp**)** StmtList **endfor**
- AExp ::= BExp [(**and**|**or**) BExp]
- BExp ::= CExp| **not** CExp
- CExp ::= Exp [(**<**|**<=**|**==**) Exp]
- Exp ::= Term ((**+** | **-**) Term)*
- Term ::= Factor ((***|**/**) Factor)*
- Factor ::= **id** | **Num** | **(** AExp **)** | **ifexp (**AExp **,** AExp **,** AExp **)**