Prog ::= Declaracao | Prog |
Att | Prog |
Cond | Prog |
Repeticao | Prog | ε

<Att> ::= Id = <expr>; | ε

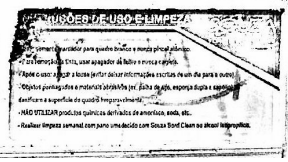
<Declaracao> ::= <tipo> Id ; | <tipo> <Att> ;

<tipo> ::= Int | float | char ;

<Repeticao> ::= while (<exp>) { <bloco> }

<exp> ::= (<expressio>) ;

<bloco> ::= Prog ;



Prog ::= Declaracao | Prog |
Att | Prog |
Cond | Prog |
Repeticao | Prog | ε

<Att> ::= Id = <expr>; | ε

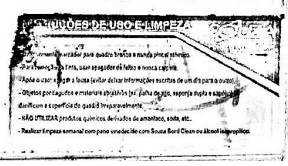
<Declaracao> ::= <tipo> Id ; | <tipo> <Att> ;

<tipo> ::= Int | float | char ;

<Repeticao> ::= while (<exp>) { <bloco> }

<exp> ::= (<expressio>) ;

<bloco> ::= Prog ;



Prog → First{ tipo, id, if, while }

Att → First{ id, ; }

Decl → int, float, char;

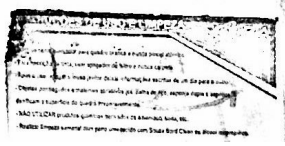
tipo → int, float, Char;

Rep → while

bloco → tipo, id, if, while

First

Follow ??



```
Prog(){
    Simb= get_next()

    if( Simb == tipo){ Decl();}
    else if( Simb == Id){ Att();}
    else if( Simb == While){ repeticao();}
    else if( Simb == if){ Cond();}
    else { Erro();}
}
```

```
Att_() {
    simb = next_simb();
  if( simb == Id ){
      simb = next_simb();
    if( simb == "=" ){
        simb = next_simb();
        exp();
    }
  }
}
```

```
Decl(){
    simb= next_simb();
    if(simb== Int || simb== float || simb==char)
        Simb= next_simb();
        if(simb== Id){
         Simb= next_simb();
          if(simb != ";"){
              erro();
}   }   }
```

```
tipo(){
    simb = next_simb();
  if(simb == int || simb == float || simb == char)
    return 0;
  }else{
      error();
      return -1;
}
```

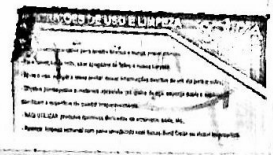
```
RepC1{
    Simb= next_simbC1;
if( simb == "while"){
    Simb= next_simbC1;
    expressaoC1;
    Simb= next_simbC1;
    if(simb== "{"){
        blocoC1;
        Simb= next_simbC1;
        if(simb != "}"){
} } } } erroC1;
```

<Bloco> ::= {programa}

```
bloco(){
    simb = next_simb();
    if(simb == "{"){
        programa();
      simb=next_simb();
    if(simb != "}"){ erro();
    }
}
```

<Exp> ::= (expr)

Seguir o modelo Aritmético.



- Prog();
- Att();
- Declaração();
- Repetição();
- Condição();

Condição ::= if(exp) bloco |
if(exp) bloco else
bloco.

```
Cond(){
  Simb= next_simb();
if( Simb== "if"){
        exp();
        bloco();
      Simb= next_simb();
      if(Simb== "else") { bloco();
}
```