Relatório do Segundo Laboratório de CES-41
Compiladores

30 de março de 2018

Disciplina: CES-41

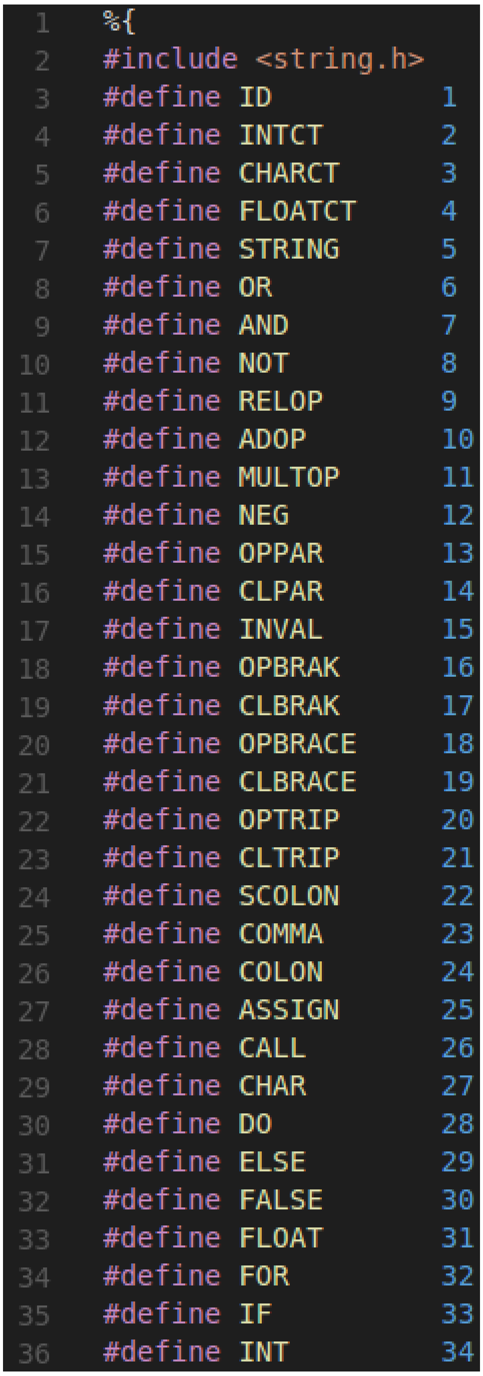
Estudante: Felipe Guimarães

Turma 19.3

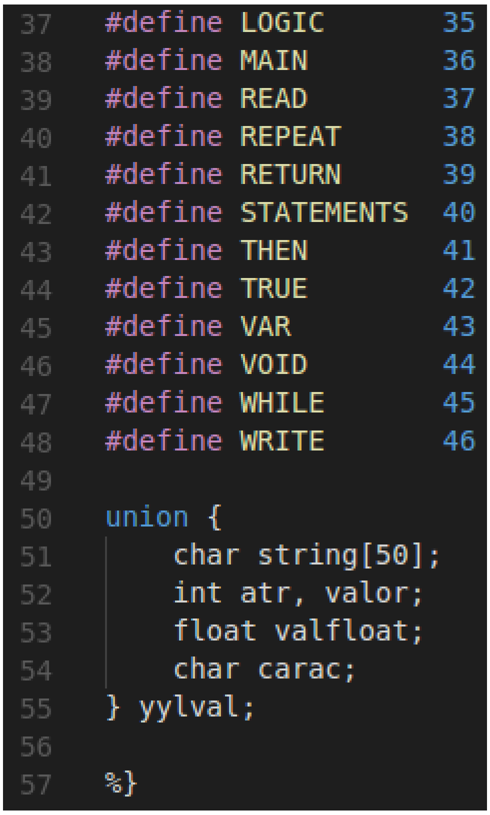
Instituto Tecnológico de Aeronáutica

I. Questão 1

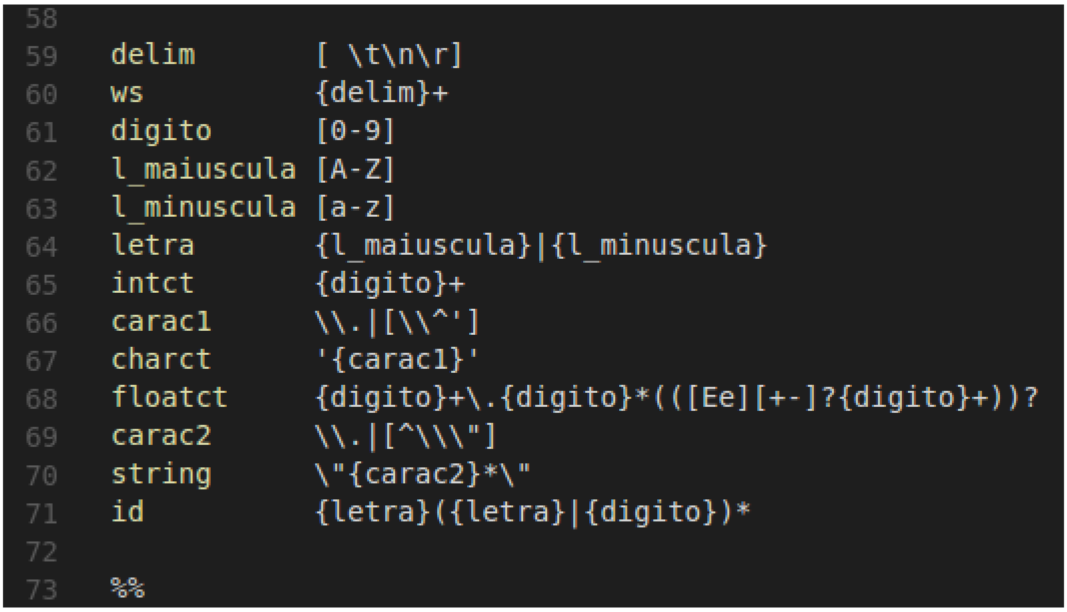
Código:



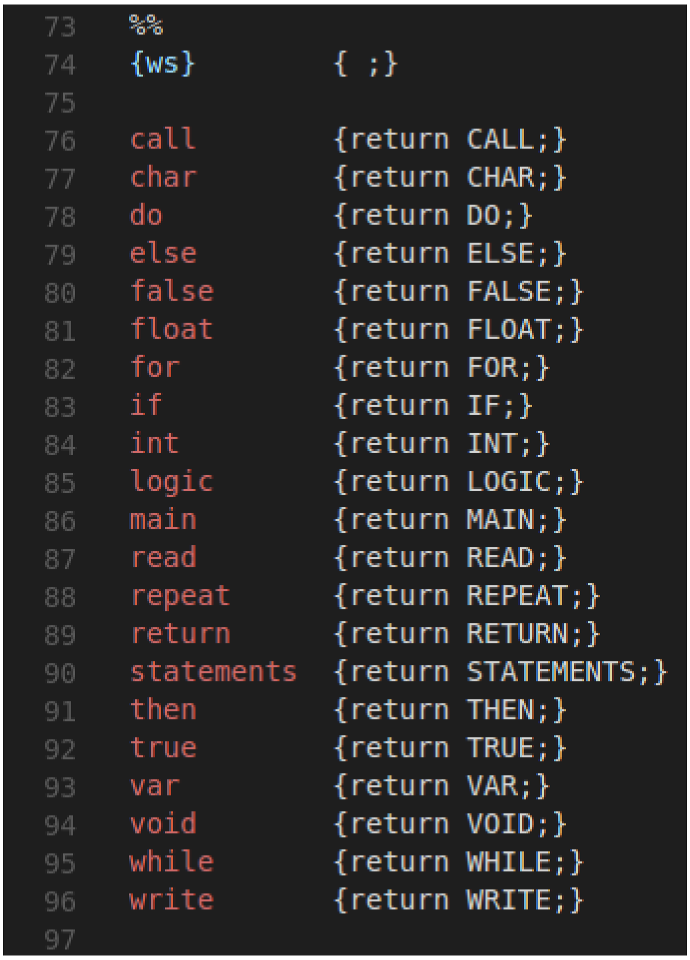
```
%{
#include <string.h>
#define ID          1
#define INTCT       2
#define CHARCT      3
#define FLOATCT     4
#define STRING      5
#define OR          6
#define AND         7
#define NOT         8
#define RELOP       9
#define ADOP        10
#define MULTOP      11
#define NEG         12
#define OPPAR       13
#define CLPAR       14
#define INVAL       15
#define OPBRAK      16
#define CLBRAK      17
#define OPBRACE     18
#define CLBRACE     19
#define OPTRIP      20
#define CLTRIP      21
#define SCOLON      22
#define COMMA       23
#define COLON       24
#define ASSIGN      25
#define CALL        26
#define CHAR        27
#define DO          28
#define ELSE        29
#define FALSE       30
#define FLOAT       31
#define FOR         32
#define IF          33
#define INT         34
```



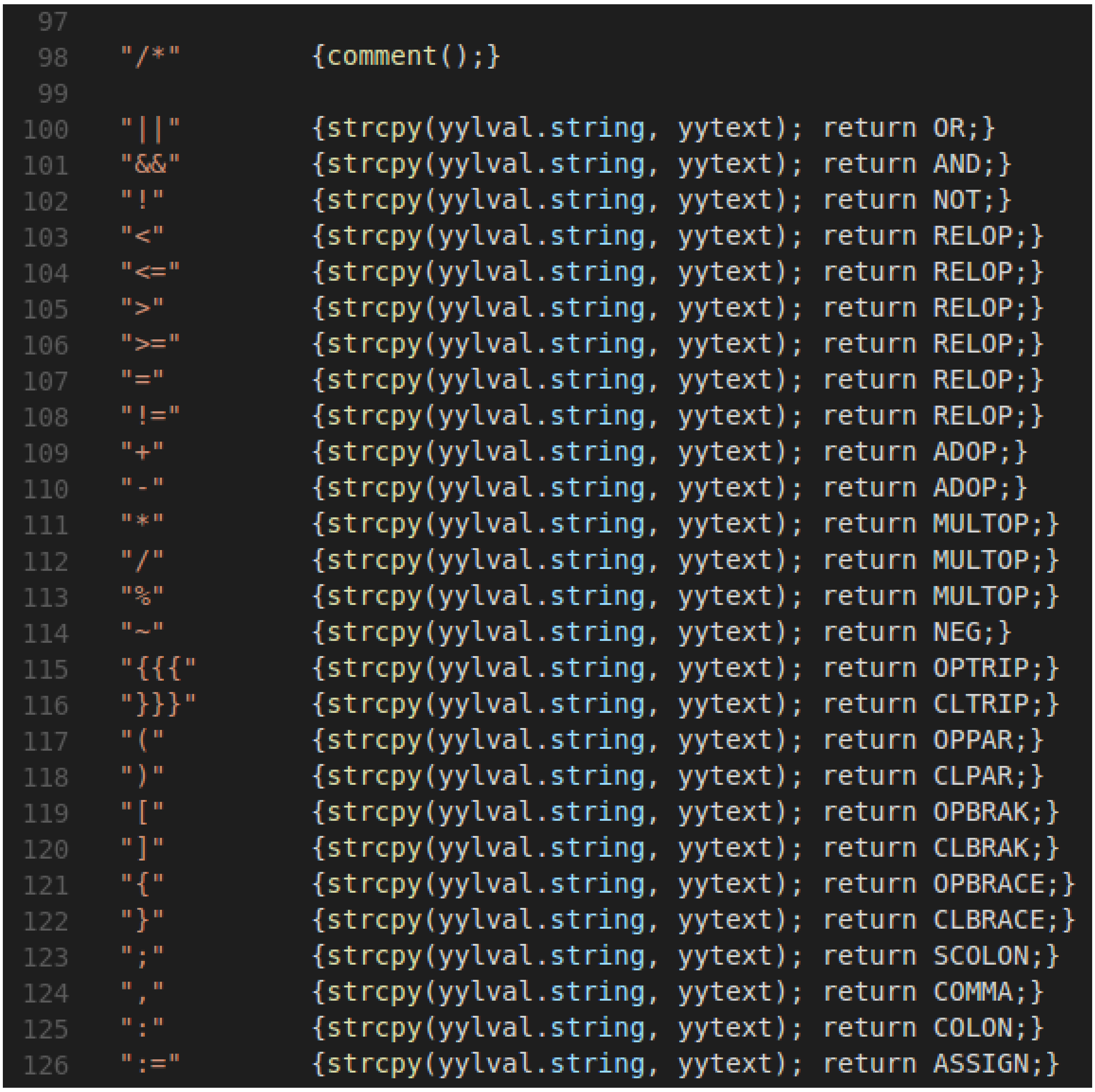
```
#define LOGIC       35
#define MAIN        36
#define READ        37
#define REPEAT      38
#define RETURN      39
#define STATEMENTS  40
#define THEN        41
#define TRUE        42
#define VAR         43
#define VOID        44
#define WHILE       45
#define WRITE       46

union {
    char string[50];
    int atr, valor;
    float valfloat;
    char carac;
} yylval;

%}
```



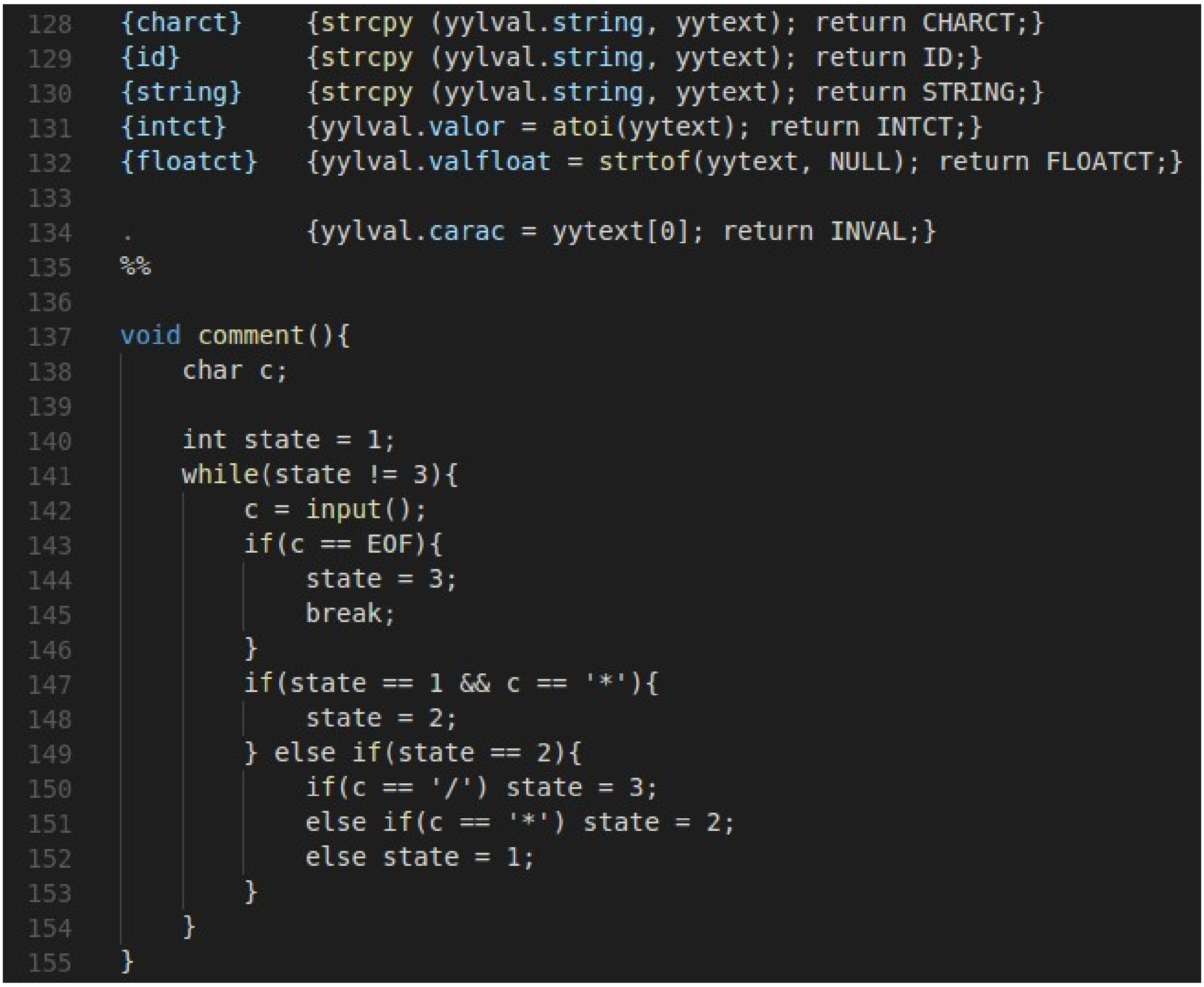
```

delim       [ \t\n\r]
ws          {delim}+
digito      [0-9]
l_maiuscula [A-Z]
l_minuscula [a-z]
letra       {l_maiuscula}|{l_minuscula}
intct       {digito}+
carac1      \\.|[\\^']
charct      '{carac1}'
floatct     {digito}+\.{digito}*(([Ee][+-]?{digito}+))?
carac2      \\.|[^\\\"]
string      \"{carac2}*\"
id          {letra}({letra}|{digito})*

%%
```



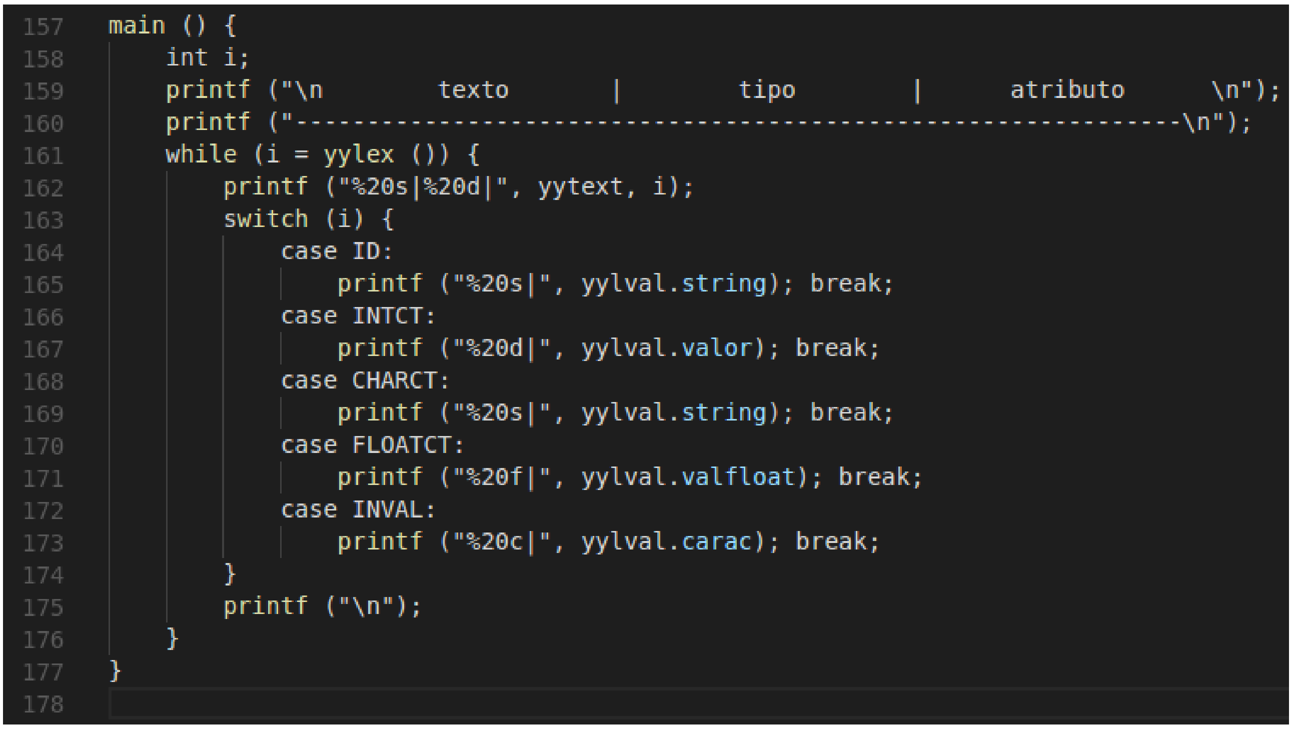
```
%%
{ws}        { ;}

call        {return CALL;}
char        {return CHAR;}
do          {return DO;}
else        {return ELSE;}
false       {return FALSE;}
float       {return FLOAT;}
for         {return FOR;}
if          {return IF;}
int         {return INT;}
logic       {return LOGIC;}
main        {return MAIN;}
read        {return READ;}
repeat      {return REPEAT;}
return      {return RETURN;}
statements  {return STATEMENTS;}
then        {return THEN;}
true        {return TRUE;}
var         {return VAR;}
void        {return VOID;}
while       {return WHILE;}
write       {return WRITE;}

```



```

"/*"        {comment();}

"||"        {strcpy(yylval.string, yytext); return OR;}
"&&"        {strcpy(yylval.string, yytext); return AND;}
"!"         {strcpy(yylval.string, yytext); return NOT;}
"<"         {strcpy(yylval.string, yytext); return RELOP;}
"<="        {strcpy(yylval.string, yytext); return RELOP;}
">"         {strcpy(yylval.string, yytext); return RELOP;}
">="        {strcpy(yylval.string, yytext); return RELOP;}
"="         {strcpy(yylval.string, yytext); return RELOP;}
"!="        {strcpy(yylval.string, yytext); return RELOP;}
"+"         {strcpy(yylval.string, yytext); return ADOP;}
"-"         {strcpy(yylval.string, yytext); return ADOP;}
"*"         {strcpy(yylval.string, yytext); return MULTOP;}
"/"         {strcpy(yylval.string, yytext); return MULTOP;}
"%"         {strcpy(yylval.string, yytext); return MULTOP;}
"~"         {strcpy(yylval.string, yytext); return NEG;}
"{{{"       {strcpy(yylval.string, yytext); return OPTRIP;}
"}}}"       {strcpy(yylval.string, yytext); return CLTRIP;}
"("         {strcpy(yylval.string, yytext); return OPPAR;}
")"         {strcpy(yylval.string, yytext); return CLPAR;}
"["         {strcpy(yylval.string, yytext); return OPBRAK;}
"]"         {strcpy(yylval.string, yytext); return CLBRAK;}
"{"         {strcpy(yylval.string, yytext); return OPBRACE;}
"}"         {strcpy(yylval.string, yytext); return CLBRACE;}
";"         {strcpy(yylval.string, yytext); return SCOLON;}
","         {strcpy(yylval.string, yytext); return COMMA;}
":"         {strcpy(yylval.string, yytext); return COLON;}
":="        {strcpy(yylval.string, yytext); return ASSIGN;}
```



```
{charct}    {strcpy (yylval.string, yytext); return CHARCT;}
{id}        {strcpy (yylval.string, yytext); return ID;}
{string}    {strcpy (yylval.string, yytext); return STRING;}
{intct}     {yylval.valor = atoi(yytext); return INTCT;}
{floatct}   {yylval.valfloat = strtof(yytext, NULL); return FLOATCT;}

.           {yylval.carac = yytext[0]; return INVAL;}
%%

void comment(){
    char c;

    int state = 1;
    while(state != 3){
        c = input();
        if(c == EOF){
            state = 3;
            break;
        }
        if(state == 1 && c == '*'){
            state = 2;
        } else if(state == 2){
            if(c == '/') state = 3;
            else if(c == '*') state = 2;
            else state = 1;
        }
    }
}
```



```
main () {
    int i;
    printf ("\n        texto       |       tipo        |     atributo      \n");
    printf ("--------------------------------------------------------------\n");
    while (i = yylex ()) {
        printf ("%20s|%20d|", yytext, i);
        switch (i) {
            case ID:
                printf ("%20s|", yylval.string); break;
            case INTCT:
                printf ("%20d|", yylval.valor); break;
            case CHARCT:
                printf ("%20s|", yylval.string); break;
            case FLOATCT:
                printf ("%20f|", yylval.valfloat); break;
            case INVAL:
                printf ("%20c|", yylval.carac); break;
        }
        printf ("\n");
    }
}
```

Análise:

Foram usadas duas entradas para a análise do programa. Uma com comentários e a outra sem comentários. Para a perfeita execução do programa, ambas deveriam apresentar o mesmo resultado. As entradas e saídas se encontram em anexo e o programa gerou saídas iguais como era esperado.

O código do programa também se encontra em anexo assim como o link para o repositório do mesmo no github.