



Trabalho 2 - Parte 1

UNIVERSIDADE DE SÃO PAULO

Departamento de Ciências de Computação e Estatística

SCC - 205 Teoria da Computação e Linguagens Formais

PROFA: Sandra Aluísio

Alunos

Bruno Pinto Ferraz Fabbri, 4154844

Henrique Pasquini Santos, 8532252

Marcelo Foresto Porto da Costa, 8782557

São Carlos, São Paulo
23 de Outubro de 2015

Sumário

Parte 1

| | |
|--------------------------------------|---|
| 1. Programas de teste e saídas | 2 |
|--------------------------------------|---|

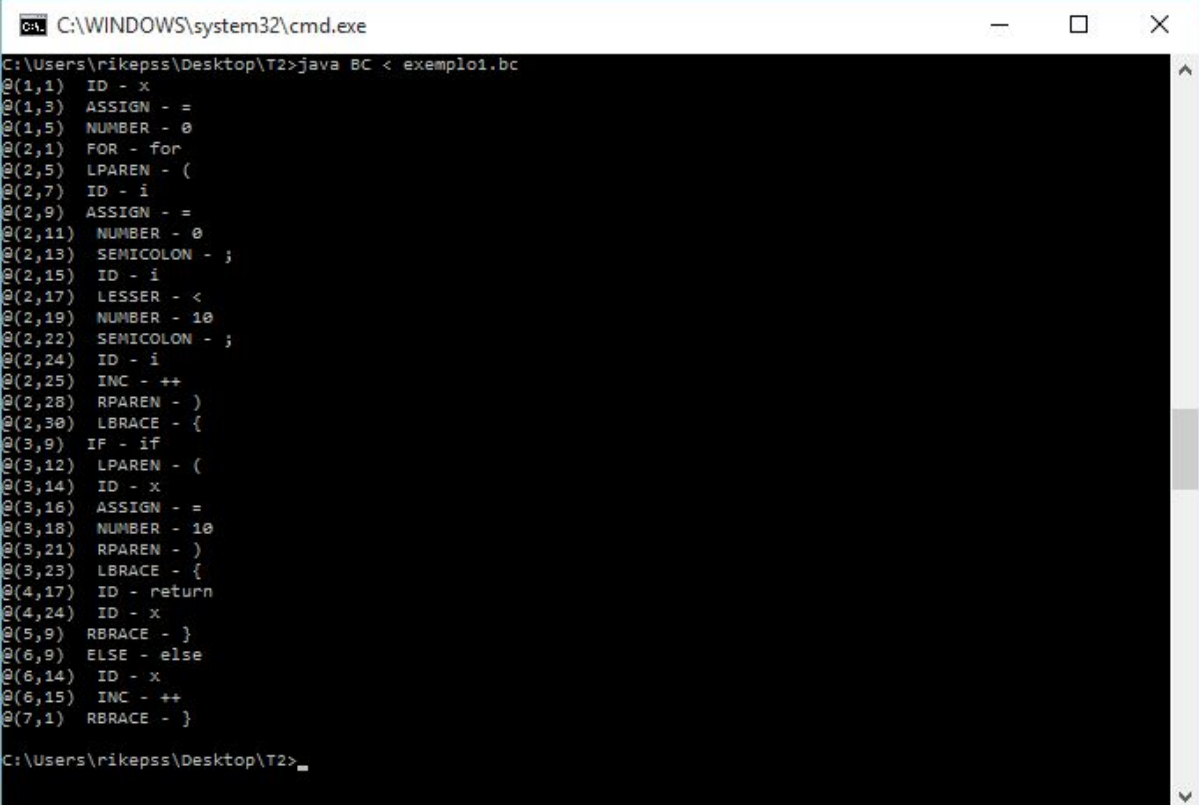
Parte 1

1. Programas de teste

Teste 1

Programa:

```
x = 0
for ( i = 0 ; i < 10 ; i++ ) {
    if ( x = 10 ) {
        return x
    }
    else x++
}
```



The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The user has entered the command `java BC < exemplo1.bc`. The output of the program is a list of tokens and their positions in the code, formatted as `@(line,column) ID - token`. The tokens include identifiers, keywords, literals, and operators, along with their respective line and column numbers in the source code.

```
C:\WINDOWS\system32\cmd.exe
C:\Users\rikepss\Desktop\T2>java BC < exemplo1.bc
@(1,1) ID - x
@(1,3) ASSIGN - =
@(1,5) NUMBER - 0
@(2,1) FOR - for
@(2,5) LPAREN - (
@(2,7) ID - i
@(2,9) ASSIGN - =
@(2,11) NUMBER - 0
@(2,13) SEMICOLON - ;
@(2,15) ID - i
@(2,17) LESSER - <
@(2,19) NUMBER - 10
@(2,22) SEMICOLON - ;
@(2,24) ID - i
@(2,25) INC - ++
@(2,28) RPAREN - )
@(2,30) LBRACE - {
@(3,9) IF - if
@(3,12) LPAREN - (
@(3,14) ID - x
@(3,16) ASSIGN - =
@(3,18) NUMBER - 10
@(3,21) RPAREN - )
@(3,23) LBRACE - {
@(4,17) ID - return
@(4,24) ID - x
@(5,9) RBRACE - }
@(6,9) ELSE - else
@(6,14) ID - x
@(6,15) INC - ++
@(7,1) RBRACE - }
```

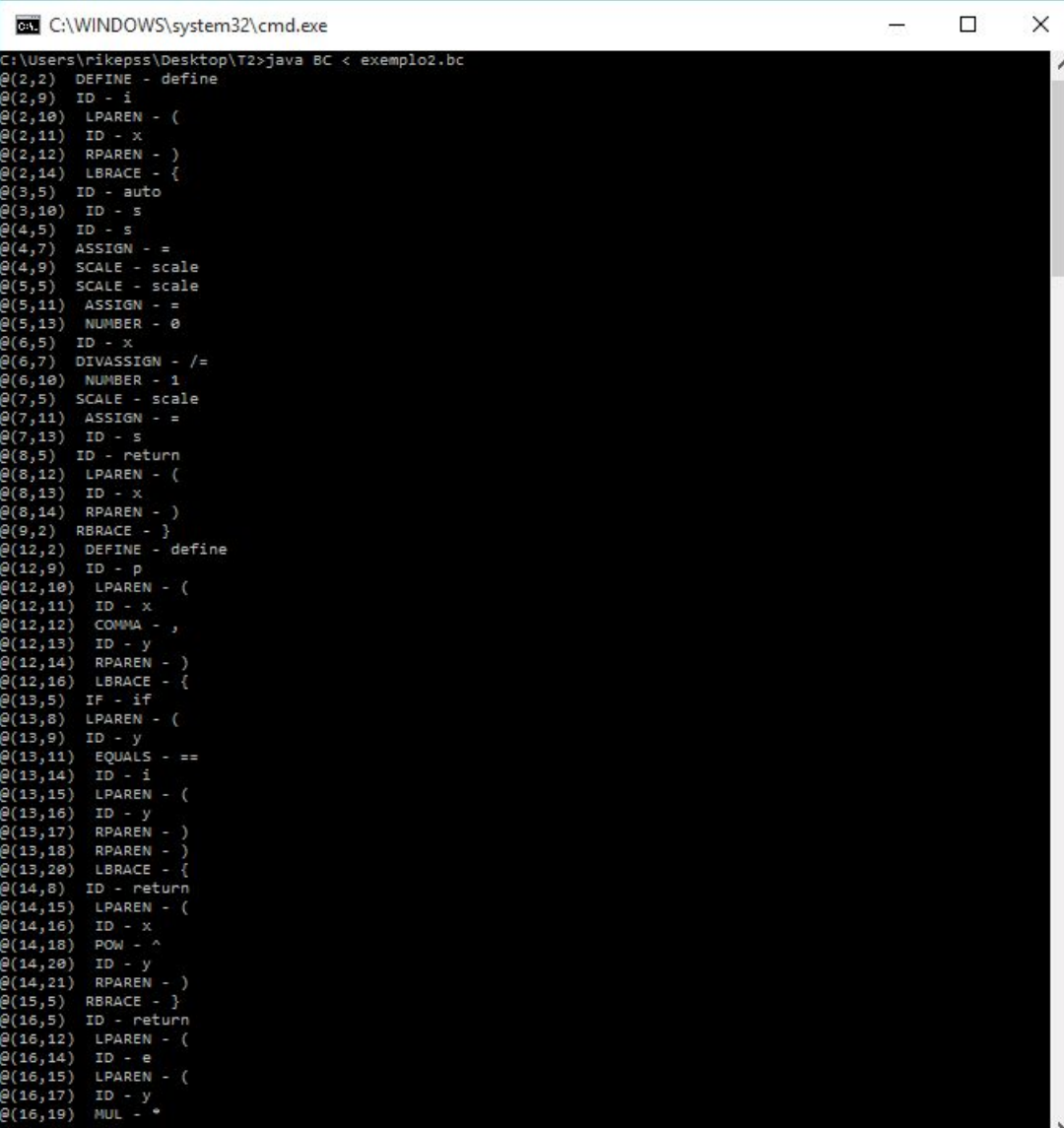
C:\Users\rikepss\Desktop\T2>

Teste 2

Programa:

```
/* A function to return the integer part of x */
define i(x) {
    auto s
    s = scale
    scale = 0
    x /= 1 /* round x down */
    scale = s
    return (x)
}

/* Use the fact that  $x^y == e^{(y \cdot \log(x))}$  */
define p(x,y) {
    if (y == i(y)) {
        return (x ^ y)
    }
    return ( e( y * l(x) ) )
}
```

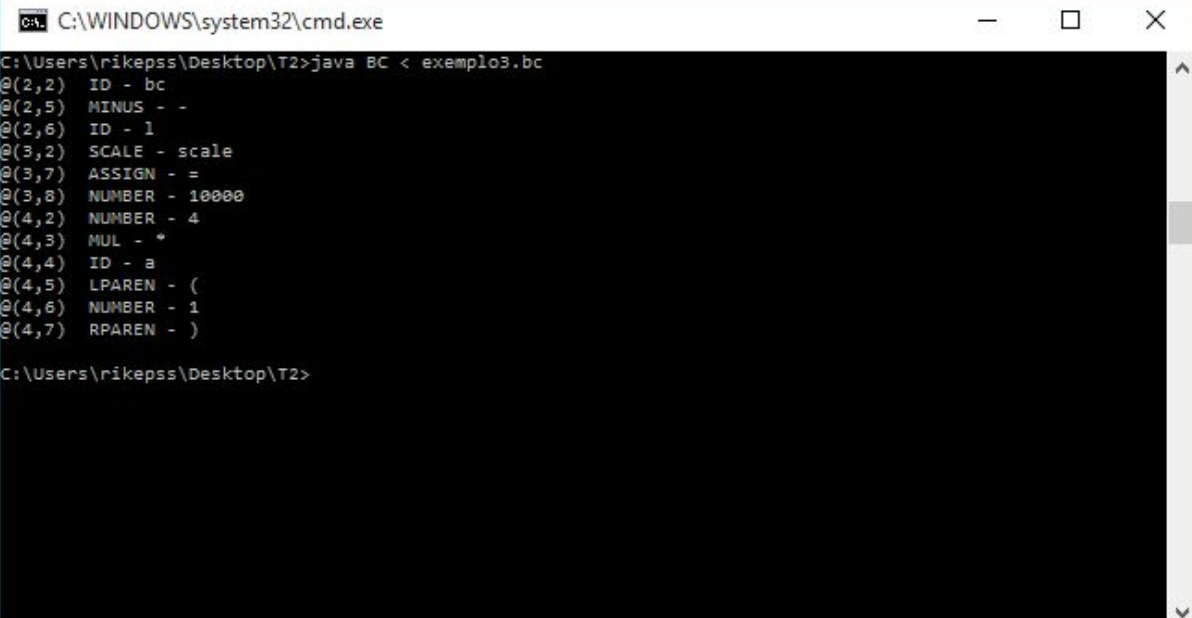


```
C:\WINDOWS\system32\cmd.exe
C:\Users\rikepss\Desktop\T2>java BC < exemplo2.bc
@ (2,2)  DEFINE - define
@ (2,9)  ID - i
@ (2,10) LPAREN - (
@ (2,11) ID - x
@ (2,12) RPAREN - )
@ (2,14) LBRACE - {
@ (3,5)  ID - auto
@ (3,10) ID - s
@ (4,5)  ID - s
@ (4,7)  ASSIGN - =
@ (4,9)  SCALE - scale
@ (5,5)  SCALE - scale
@ (5,11) ASSIGN - =
@ (5,13) NUMBER - 0
@ (6,5)  ID - x
@ (6,7)  DIVASSIGN - /=
@ (6,10) NUMBER - 1
@ (7,5)  SCALE - scale
@ (7,11) ASSIGN - =
@ (7,13) ID - s
@ (8,5)  ID - return
@ (8,12) LPAREN - (
@ (8,13) ID - x
@ (8,14) RPAREN - )
@ (9,2)  RBRACE - }
@ (12,2) DEFINE - define
@ (12,9) ID - p
@ (12,10) LPAREN - (
@ (12,11) ID - x
@ (12,12) COMMA - ,
@ (12,13) ID - y
@ (12,14) RPAREN - )
@ (12,16) LBRACE - {
@ (13,5)  IF - if
@ (13,8)  LPAREN - (
@ (13,9)  ID - y
@ (13,11) EQUALS - ==
@ (13,14) ID - i
@ (13,15) LPAREN - (
@ (13,16) ID - y
@ (13,17) RPAREN - )
@ (13,18) RPAREN - )
@ (13,20) LBRACE - {
@ (14,8)  ID - return
@ (14,15) LPAREN - (
@ (14,16) ID - x
@ (14,18) POW - ^
@ (14,20) ID - y
@ (14,21) RPAREN - )
@ (15,5)  RBRACE - }
@ (16,5)  ID - return
@ (16,12) LPAREN - (
@ (16,14) ID - e
@ (16,15) LPAREN - (
@ (16,17) ID - y
@ (16,19) MUL - *
```

Teste 3

Programa:

```
1 // The atan of 1 is 45 degrees, which is pi/4 in radians.  
2 bc -l  
3 scale=10000  
4 4*a(1)  
5 // This may take several minutes to calculate.
```

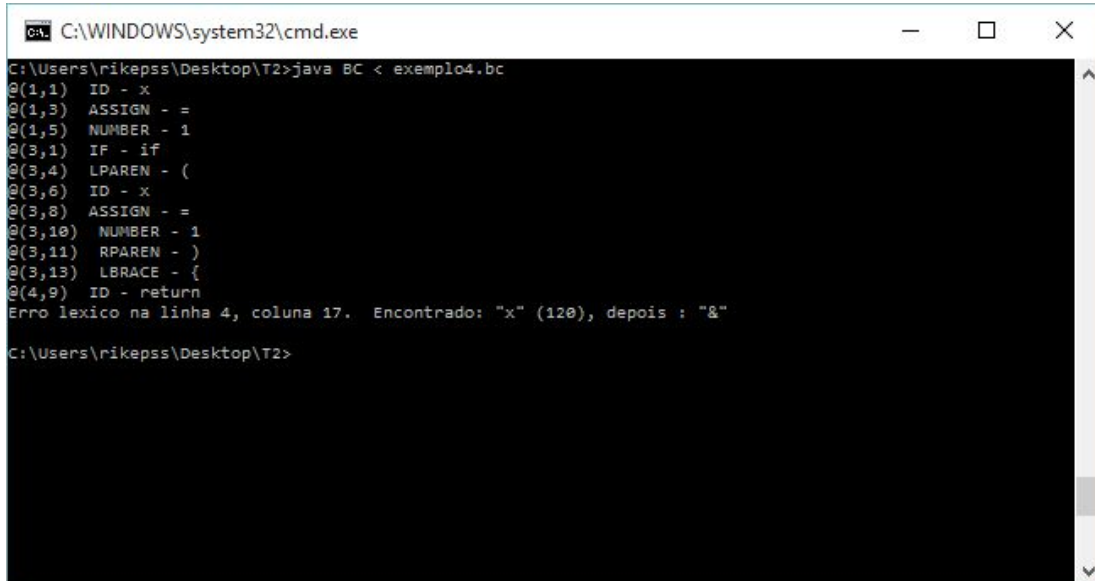


The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The user has executed the command `C:\Users\rikepss\Desktop\T2>java BC < exemplo3.bc`. The output of the program is a list of tokens from the `bc` calculator, each preceded by a label in parentheses: `@(2,2) ID - bc`, `@(2,5) MINUS - -`, `@(2,6) ID - 1`, `@(3,2) SCALE - scale`, `@(3,7) ASSIGN - =`, `@(3,8) NUMBER - 10000`, `@(4,2) NUMBER - 4`, `@(4,3) MUL - *`, `@(4,4) ID - a`, `@(4,5) LPAREN - (`, `@(4,6) NUMBER - 1`, and `@(4,7) RPAREN -)`. The prompt `C:\Users\rikepss\Desktop\T2>` is visible at the bottom.

Teste 4

Programa:

```
1 x = 1
2
3 if ( x = 1) {
4     return &x;
5 }
6 else return 0;
```



```
C:\WINDOWS\system32\cmd.exe
C:\Users\rikepss\Desktop\T2>java BC < exemplo4.bc
@ (1,1) ID - x
@ (1,3) ASSIGN - =
@ (1,5) NUMBER - 1
@ (3,1) IF - if
@ (3,4) LPAREN - (
@ (3,6) ID - x
@ (3,8) ASSIGN - =
@ (3,10) NUMBER - 1
@ (3,11) RPAREN - )
@ (3,13) LBRACE - {
@ (4,9) ID - return
Erro lexico na linha 4, coluna 17. Encontrado: "x" (120), depois : "&"
C:\Users\rikepss\Desktop\T2>
```