

Algoritmos

Strings, Cores, Decremento e Incremento de Variáveis

Prof. Dr. Osmar Betazzi Dordal

4

Strings, Cores, Incremento e Decremento de Variáveis

Sobre *Strings* – Cadeia de Caracteres

- A *string* em Pascal é na verdade uma sequência de caracteres com uma especificação de **tamanho opcional**.
- Os caracteres podem ser numéricos, letras, em branco, caracteres especiais ou uma combinação de todos.
- A linguagem Pascal fornece vários tipos de objetos de *string*, dependendo do sistema e implementação.
- Discutiremos tipos mais comuns de *strings* usados em algoritmos.

Tamanho da *String*

- Temos uma função no Pascal para retornar o tamanho de uma *string*.
- Esta função se chama *length(x)*. Onde o x é uma variável do tipo *string*.

```
1 Program cadeia_caracteres;  
2 var  
3   s: string;  
4  
5 Begin  
6  
7   s := '123456789';  
8   writeln('a string é: ', s);  
9   writeln('o tamanho da string é: ', length(s));  
10  
11 End.
```

```
a string é: 123456789  
o tamanho da string é: 9
```

Tamanho da *String*

```
1 Program cadeia_caracteres;  
2 var  
3   s, entrada : string;  
4  
5 Begin  
6  
7   s := '123456789';  
8   writeln('a string é: ', s);  
9   writeln('o tamanho da string é: ', length(s));  
10  
11  writeln('digite uma cadeia de caracteres:');  
12  readln(entrada);  
13  writeln('a string digitada é: ', entrada);  
14  writeln('o tamanho da string é: ', length(entrada));  
15  
16 End.
```

```
a string é: 123456789  
o tamanho da string é: 9  
digite uma cadeia de caracteres:  
algoritmos em pascal  
a string é: algoritmos em pascal  
o tamanho da string é: 20
```

Tamanho da *String*

```
1 Program cadeia_caracteres;  
2 var  
3   s, entrada : string[10];  
4  
5 Begin  
6  
7   s := '123456789';  
8   writeln('a string é: ', s);  
9   writeln('o tamanho da string é: ', length(s));  
10  
11  writeln('digite uma cadeia de caracteres:');  
12  readln(entrada);  
13  writeln('a string digitada é: ', entrada);  
14  writeln('o tamanho da string é: ', length(entrada));  
15  
16 End.
```

```
a string é: 123456789  
o tamanho da string é: 9  
digite uma cadeia de caracteres:  
algoritmos em pascal  
a string digitada é: algoritmos  
o tamanho da string é: 10
```

Tamanho da *String*

```
1 Program cadeia_caracteres;  
2 var  
3   s, entrada : string[10];  
4  
5 Begin  
6  
7   s := '123456789';  
8   writeln('a string é: ', s);  
9   writeln('o tamanho da string é: ', length(s));  
10  
11  writeln('digite uma cadeia de caracteres:');  
12  readln(entrada);  
13  writeln('a string digitada é: ', entrada);  
14  writeln('o tamanho da string é: ', length(entrada));  
15  
16 End.
```

a string é: 123456789
o tamanho da string é: 9
digite uma cadeia de caracteres:
algoritmos em pascal
a string digitada é: algoritmos
o tamanho da string é: 10

String com tamanho declarado

Concatenação de *Strings* – *Concat*

- A concatenação de duas ou mais *strings*, ocorre por meio da função *concat(x,y,...,n)*.

```
1 Program concatenacao;  
2  
3 var  
4   str1, str2, str3, str4, str5, str6: string;  
5  
6 Begin  
7  
8   str1 := '1';  
9   str2 := '_2_';  
10  str3 := 'Olá';  
11  str4 := 'Mundo';  
12  str5 := '!';  
13  str6 := ' ';  
14  
15  writeln(concat(str1, str2));  
16  
17  
18  
19  
20 End.
```

1_2_

Concatenação de *Strings* – Concat

- A concatenação de duas ou mais *strings*, ocorre por meio da função *concat(x,y,...,n)*.

```
1 Program concatenacao;  
2  
3 var  
4   str1, str2, str3, str4, str5, str6: string;  
5  
6 Begin  
7  
8   str1 := '1';  
9   str2 := '_2_';  
10  str3 := 'Olá';  
11  str4 := 'Mundo';  
12  str5 := '!!';  
13  str6 := ' ';  
14  
15  writeln(concat(str1, str2));  
16  writeln(concat(str3, str4));  
17  
18  
19  
20 End.
```

1_2_
OláMundo

Concatenação de *Strings* – Concat

- A concatenação de duas ou mais *strings*, ocorre por meio da função *concat(x,y,...,n)*.

```
1 Program concatenacao;  
2  
3 var  
4   str1, str2, str3, str4, str5, str6: string;  
5  
6 Begin  
7  
8   str1 := '1';  
9   str2 := '_2_';  
10  str3 := 'Olá';  
11  str4 := 'Mundo';  
12  str5 := '!';  
13  str6 := ' ';  
14  
15  writeln(concat(str1, str2));  
16  writeln(concat(str3, str4));  
17  writeln(concat(str3, str4, str5));  
18  
19  
20 End.
```

1_2_
OláMundo
OláMundo!

Concatenação de *Strings* – Concat

- A concatenação de duas ou mais *strings*, ocorre por meio da função *concat(x,y,...,n)*.

```
1 Program concatenacao;  
2  
3 var  
4   str1, str2, str3, str4, str5, str6: string;  
5  
6 Begin  
7  
8   str1 := '1';  
9   str2 := '_2_';  
10  str3 := 'Olá';  
11  str4 := 'Mundo';  
12  str5 := '!';  
13  str6 := ' ';  
14  
15  writeln(concat(str1, str2));  
16  writeln(concat(str3, str4));  
17  writeln(concat(str3, str4, str5));  
18  writeln(concat(str3, str6, str4, str5));  
19  
20 End.
```

```
1_2_  
OláMundo  
OláMundo!  
Olá Mundo!
```

Caixa alta de *Strings* – *uppercase*

- *Uppcase* é uma função que recebe como parâmetro uma *string* e a converte em caixa alta. *uppercase(x)*

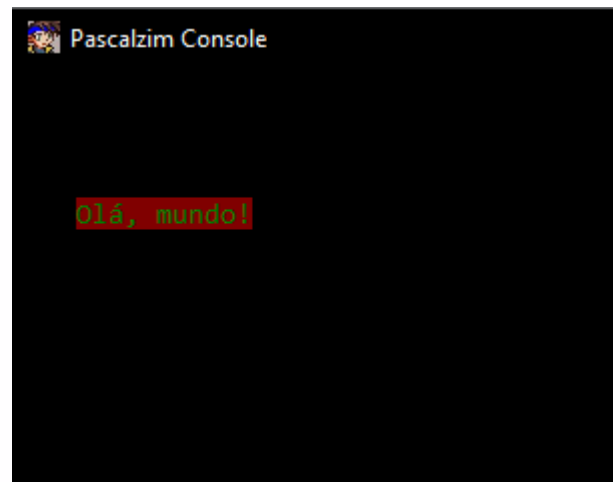
```
1 Program concatenacao;  
2  
3 var  
4   str1, str2, str3, str4, str5, str6, aux: string;  
5  
6  
7 Begin  
8  
9   str1 := '1';  
10  str2 := '_2_';  
11  str3 := 'Olá';  
12  str4 := 'Mundo';  
13  str5 := '!';  
14  str6 := ' ';  
15  
16  writeln(concat(str1, str2));  
17  writeln(concat(str3, str4));  
18  writeln(concat(str3, str4, str5));  
19  writeln(concat(str3, str6, str4, str5));  
20  
21  aux := concat(str3, str6, str4, str5);  
22  writeln(uppercase(aux));  
23  
24 End.
```

1_2_
OláMundo
OláMundo!
Olá Mundo!
OLÁ MUNDO!

Cores e Posicionamento

- *gotoxy*(coluna, linha).
- *textcolor*(cor)
- *textbackground*(cor)

```
1 Program cores;  
2  
3 Begin  
4  
5     gotoxy(5,5);  
6     textcolor(green);  
7     textbackground(red);  
8     write('Olá, mundo!');  
9  
10 End.
```



Cores e Posicionamento

```
1 Program cores;  
2  
3 Begin  
4  
5     textbackground(white);  
6  
7     textcolor(black);  
8     writeln('Olá, mundo!');  
9  
10    textcolor(green);  
11    writeln('Olá, mundo!');  
12  
13    textcolor(yellow);  
14    writeln('Olá, mundo!');  
15  
16 End.
```

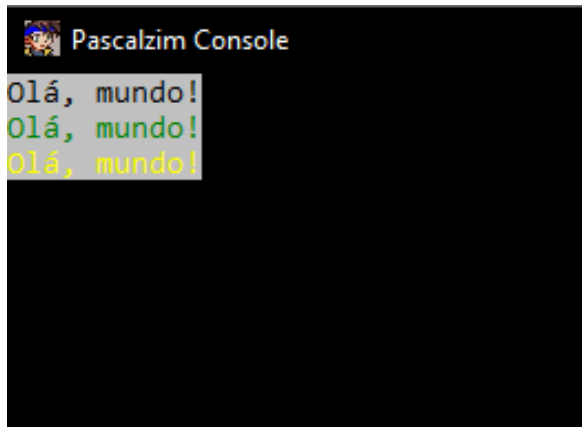


Pascalzim Console

```
Olá, mundo!  
Olá, mundo!  
Olá, mundo!
```

Cores e Posicionamento

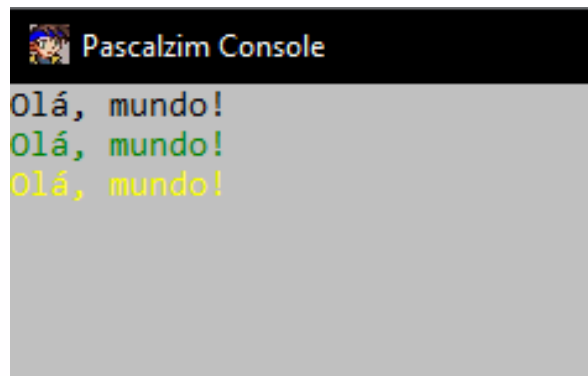
```
1 Program cores;
2
3 Begin
4
5     textbackground(white);
6
7     textcolor(black);
8     writeln('Olá, mundo!');
9
10    textcolor(green);
11    writeln('Olá, mundo!');
12
13    textcolor(yellow);
14    writeln('Olá, mundo!');
15
16 End.
```



Pascalzim Console

```
Olá, mundo!
Olá, mundo!
Olá, mundo!
```

```
1 Program cores;
2
3 Begin
4
5     textbackground(white);
6     clrscr;
7
8     textcolor(black);
9     writeln('Olá, mundo!');
10
11
12    textcolor(green);
13    writeln('Olá, mundo!');
14
15
16    textcolor(yellow);
17    writeln('Olá, mundo!');
18
19 End.
```

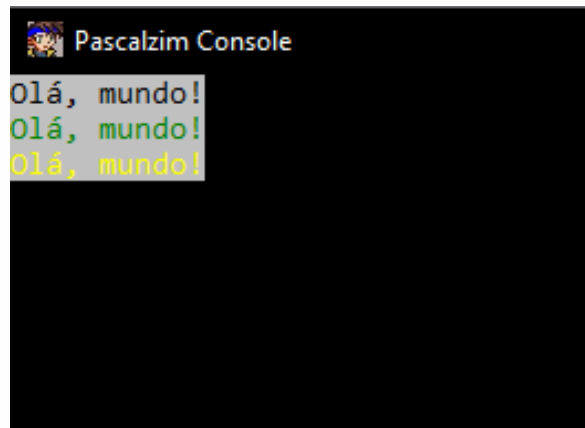


Pascalzim Console

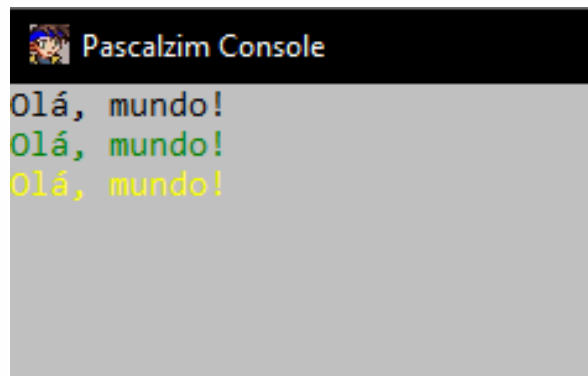
```
Olá, mundo!
Olá, mundo!
Olá, mundo!
```

Cores e Posicionamento

```
1 Program cores;
2
3 Begin
4
5     textbackground(white);
6
7     textcolor(black);
8     writeln('Olá, mundo!');
9
10    textcolor(green);
11    writeln('Olá, mundo!');
12
13    textcolor(yellow);
14    writeln('Olá, mundo!');
15
16 End.
```



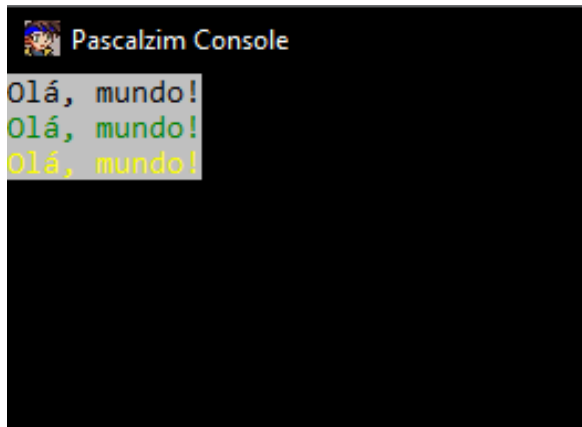
```
1 Program cores;
2
3 Begin
4
5     textbackground(white);
6     clrscr;
7
8     textcolor(black);
9     writeln('Olá, mundo!');
10
11
12    textcolor(green);
13    writeln('Olá, mundo!');
14
15
16    textcolor(yellow);
17    writeln('Olá, mundo!');
18
19 End.
```



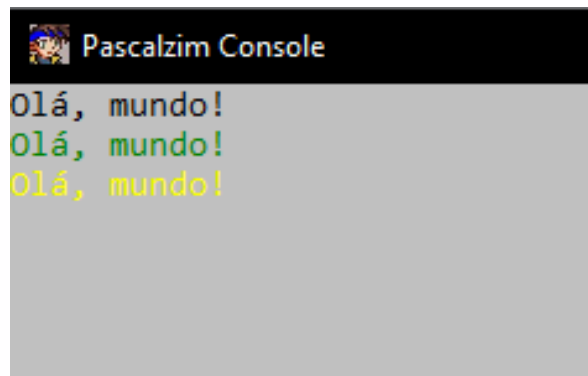
```
1 Program cores;
2
3 Begin
4
5     writeln('Olá pessoal...');
6     delay(1000);
7     clrscr;
8
9     writeln('...ainda estou executando...');
10    delay(1000);
11    clrscr;
12
13    writeln('...vou agora terminar');
14    delay(1000);
15    clrscr;
16
17    writeln('Fim...');
18    delay(1000);
19    clrscr;
20
21    writeln('Tchau!!!!');
22    delay(1000);
23
24 End.
```


Cores e Posicionamento

```
1 Program cores;
2
3 Begin
4
5     textbackground(white);
6
7     textcolor(black);
8     writeln('Olá, mundo!');
9
10    textcolor(green);
11    writeln('Olá, mundo!');
12
13    textcolor(yellow);
14    writeln('Olá, mundo!');
15
16 End.
```



```
1 Program cores;
2
3 Begin
4
5     textbackground(white);
6     clrscr;
7
8     textcolor(black);
9     writeln('Olá, mundo!');
10
11
12    textcolor(green);
13    writeln('Olá, mundo!');
14
15
16    textcolor(yellow);
17    writeln('Olá, mundo!');
18
19 End.
```

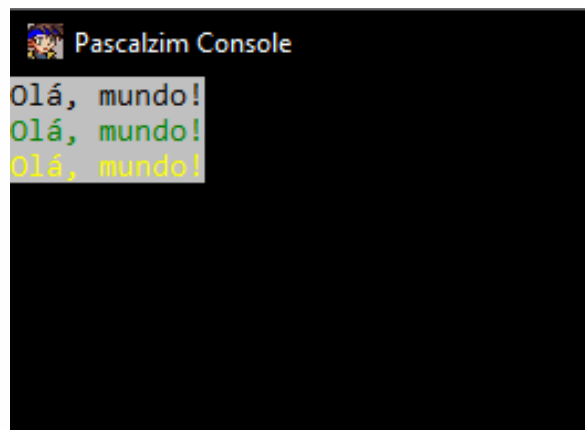


```
1 Program cores;
2
3 Begin
4
5     writeln('Olá pessoal...');
6     delay(1000);
7     clrscr;
8
9     writeln('...ainda estou executando...');
10    delay(1000);
11    clrscr;
12
13    writeln('...vou agora terminar');
14    delay(1000);
15    clrscr;
16
17    writeln('Fim...');
18    delay(1000);
19    clrscr;
20
21    writeln('Tchau!!!!');
22    delay(1000);
23
24 End.
```

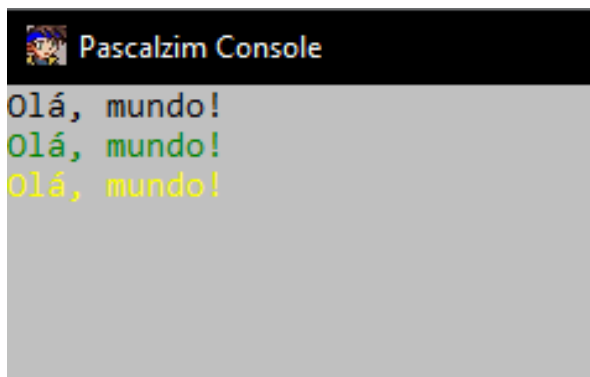
Execute com e sem o comando *clrscr*;

Cores e Posicionamento

```
1 Program cores;
2
3 Begin
4
5     textbackground(white);
6
7     textcolor(black);
8     writeln('Olá, mundo!');
9
10    textcolor(green);
11    writeln('Olá, mundo!');
12
13    textcolor(yellow);
14    writeln('Olá, mundo!');
15
16 End.
```



```
1 Program cores;
2
3 Begin
4
5     textbackground(white);
6     clrscr;
7
8     textcolor(black);
9     writeln('Olá, mundo!');
10
11    textcolor(green);
12    writeln('Olá, mundo!');
13
14    textcolor(yellow);
15    writeln('Olá, mundo!');
16
17    textcolor(yellow);
18    writeln('Olá, mundo!');
19 End.
```



```
1 Program cores;
2
3 Begin
4
5     gotoxy(12,2);
6     writeln('Olá pessoal...');
7     delay(1000);
8     clrscr;
9
10    gotoxy(4,4);
11    writeln('...ainda estou executando...');
12    delay(1000);
13    clrscr;
14
15    gotoxy(6,6);
16    writeln('...vou agora terminar');
17    delay(1000);
18    clrscr;
19
20    gotoxy(8,8);
21    writeln('Fim...');
22    delay(1000);
23    clrscr;
24
25    gotoxy(10,10);
26    writeln('Tchau!!!!');
27    delay(1000);
28
29 End.
```

Números Aleatórios – *Random*

- Recebe como parâmetro um inteiro x e retorna um número n no intervalo $0 \leq n < x$
- *random(x):integer;*

```
1 Program numero_aleatorio;  
2 var  
3     n:integer;  
4 Begin  
5     n := random(10);  
6     writeln (n);  
7 End.
```

Neste código, a cada execução, o sorteio será no intervalo: $0 \leq n < 10$

Decremento e Incremento – *dec* e *inc*.

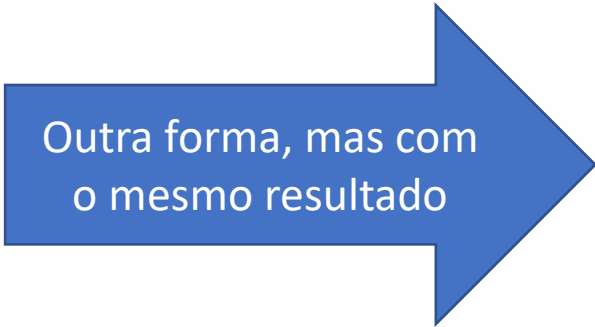
- Decremento de um **número inteiro** *dec*(*x*) Incremento de um **número inteiro** *inc*(*x*)

```
1 Program dec_inc_inteiro;  
2 var  
3   numero: integer;  
4  
5 Begin  
6  
7   numero := 10;  
8  
9   dec(numero);  
10  writeln(numero);  
11  dec(numero);  
12  writeln(numero);  
13  dec(numero);  
14  writeln(numero);  
15  inc(numero);  
16  writeln(numero);  
17  inc(numero);  
18  writeln(numero);  
19  inc(numero);  
20  writeln(numero);  
21  
22  
23 End.
```

Decremento e Incremento – *dec* e *inc*.

- Decremento de um **número inteiro** *dec*(*x*) Incremento de um **número inteiro** *inc*(*x*)

```
1 Program dec_inc_inteiro;  
2 var  
3   numero: integer;  
4  
5 Begin  
6  
7   numero := 10;  
8  
9   dec(numero);  
10  writeln(numero);  
11  dec(numero);  
12  writeln(numero);  
13  dec(numero);  
14  writeln(numero);  
15  inc(numero);  
16  writeln(numero);  
17  inc(numero);  
18  writeln(numero);  
19  inc(numero);  
20  writeln(numero);  
21  
22  
23 End.
```



Outra forma, mas com
o mesmo resultado

```
1 Program dec_inc_inteiro_outro;  
2 var  
3   numero: integer;  
4  
5 Begin  
6  
7   numero := 10;  
8  
9   numero := numero - 1;  
10  writeln(numero);  
11  numero := numero - 1;  
12  writeln(numero);  
13  numero := numero - 1;  
14  writeln(numero);  
15  numero := numero + 1;  
16  writeln(numero);  
17  numero := numero + 1;  
18  writeln(numero);  
19  numero := numero + 1;  
20  writeln(numero);  
21  
22  
23 End.
```

Decremento e Incremento – *dec* e *inc*.

- Decremento de um **número inteiro** *dec*(*x*) Incremento de um **número inteiro** *inc*(*x*)

```
1 Program dec_inc_inteiro;  
2 var  
3   numero: integer;  
4  
5 Begin  
6  
7   numero := 10;  
8  
9   dec(numero);  
10  writeln(numero);  
11  dec(numero);  
12  writeln(numero);  
13  dec(numero);  
14  writeln(numero);  
15  inc(numero);  
16  writeln(numero);  
17  inc(numero);  
18  writeln(numero);  
19  inc(numero);  
20  writeln(numero);  
21  
22  
23 End.
```

9
10
11
12
13
14
15
16
17
18
19

9
8
7
8
9
10

```
1 Program dec_inc_inteiro_outro;  
2 var  
3   numero: integer;  
4  
5 Begin  
6  
7   numero := 10;  
8  
9   numero := numero - 1;  
10  writeln(numero);  
11  numero := numero - 1;  
12  writeln(numero);  
13  numero := numero - 1;  
14  writeln(numero);  
15  numero := numero + 1;  
16  writeln(numero);  
17  numero := numero + 1;  
18  writeln(numero);  
19  numero := numero + 1;  
20  writeln(numero);  
21  
22  
23 End.
```

Decremento e Incremento – *dec* e *inc*.

- Decremento de um **caractere** *dec(x)* Incremento de um **caractere** *inc(x)*

```
1 Program dec_inc_inteiro_outro;  
2 var  
3   caractere: char;  
4  
5 Begin  
6  
7   caractere := 'd';  
8  
9   dec(caractere);  
10  writeln(caractere);  
11  dec(caractere);  
12  writeln(caractere);  
13  dec(caractere);  
14  writeln(caractere);  
15  inc(caractere);  
16  writeln(caractere);  
17  inc(caractere);  
18  writeln(caractere);  
19  inc(caractere);  
20  writeln(caractere);  
21  
22  
23 End.
```

9	c
10	b
11	a
12	b
13	c
14	d

Decremento e Incremento – *dec* e *inc*.

- Decremento de um **caractere** *dec*(*x*) Incremento de um **caractere** *inc*(*x*)

```
1 Program dec_inc_inteiro_outro;  
2 var  
3   caractere: char;  
4  
5 Begin  
6  
7   caractere := 'd';  
8  
9   dec(caractere);  
10  writeln(caractere);  
11  dec(caractere);  
12  writeln(caractere);  
13  dec(caractere);  
14  writeln(caractere);  
15  inc(caractere);  
16  writeln(caractere);  
17  inc(caractere);  
18  writeln(caractere);  
19  inc(caractere);  
20  writeln(caractere);  
21  
22  
23 End.
```

9	c
10	
11	b
12	
13	a
14	
15	b
16	
17	c
18	
19	d

~~caractere := caractere - 1;~~

~~caractere := caractere + 1;~~

Curiosidade – *readkey*

- *readkey* pega o código ASCII que foi digitado no teclado.

```
1 Program numero_aleatorio;  
2 var  
3   c: char;  
4  
5 Begin  
6   writeln( 'Please press a key' );  
7   c := Readkey;  
8   writeln( ' Você pressionou ', c, ', cujo valor ASCII é ', ord(c), '.' ) ;  
9 End.
```

Se digitarmos a letra 'a' o código ASCII desta letra será: 97.

Se digitarmos a letra 'A' o código ASCII desta letra será: 65.

Referências

- FORBELLONE, André L. **Lógica de Programação**. Prentice Hall Brasil, 3ª edição, 2005.
- VELOSO, Paulo; et al. **Estrutura de dados**. Rio de Janeiro: Campus, 4ª edição, 1996.
- LAGES & GUIMARAES. **Algoritmos e Estrutura de dados**. Ed. LTC, 1994.
- FARRER, H. **Algoritmos estruturados**. Rio de Janeiro: Guanabara Koogan, 3ª edição, 1989.
- LUIZ, Jaime. **Estrutura de dados e seus algoritmos**. Editora LTC.
- GUEDES, S. **Lógica de Programação Algorítmica**. Editora Pearson, 2014.
- MANZANO, José Augusto N. G. **Algoritmos lógica para desenvolvimento de programação de computadores**. Ed. 1, São Paulo, Erica 2016.
- MANZANO, José Augusto N. G. **Algoritmos técnicas de programação**. Ed 2, São Paulo, Erica, 2016.