

## Curso Lógica de Programação e Algoritmos Expert

[devsuperior.com.br](http://devsuperior.com.br)

**Prof. Dr. Nelio Alves**

### Capítulo: Linguagem C

#### Atenção

Este material foi elaborado para alunos do curso **Lógica de Programação e Algoritmos Expert** do professor Nelio Alves.

A didática do conteúdo deste material está adaptada para quem já fez a parte do referido curso sobre construção de algoritmos na linguagem do VisualG.

Para mais informações:  
[devsuperior.com.br](http://devsuperior.com.br)

#### Instalação das ferramentas

**Compilador GCC (para Windows: MinGW)**

<http://www.mingw.org/>

Variável de ambiente Path: C:\MinGW\bin

**IDE : Code Blocks**

<http://www.codeblocks.org/downloads>

- Download the binary release
- Escolher pacote **codeblocks-xx.xx-setup.exe**

#### Primeiro programa em C

| VisualG                | C                                     |
|------------------------|---------------------------------------|
| Algoritmo "primeiro"   | <code>#include &lt;stdio.h&gt;</code> |
| Var                    | <code>int main() {</code>             |
| Inicio                 | <code>printf("Ola mundo!\n");</code>  |
| escreval("Ola mundo!") | <code>return 0;</code>                |
| Fimalgoritmo           | <code>}</code>                        |

## TIPOS DE DADOS E VARIÁVEIS

| Significado               | Tipo VisualG | Tipo C         | Valor padrão                    | Observação   |
|---------------------------|--------------|----------------|---------------------------------|--|
| número inteiro            | inteiro      | <b>int</b>     | não atribuído (lixo de memória) | <b>int</b> : -32767 a 32767<br><b>long int</b> : -2147483648 a 2147483647<br><b>long long int</b> : -9223372036854775807 a 9223372036854775807 |
| número de ponto flutuante | real         | <b>double</b>  | não atribuído (lixo de memória) | <b>float</b> : precisão simples<br><b>double</b> : precisão dupla  |
| um único caractere        | caractere    | <b>char</b>    | não atribuído (lixo de memória) | Na linguagem C, para se representar um único caractere usa-se o tipo char. Valores literais devem ter aspas simples. Exemplo: 'F'              |
| texto                     | caractere    | <b>char[ ]</b> | não atribuído (lixo de memória) | Na linguagem C, para se representar um texto, usa-se um <b>vetor</b> de char. Valores literais devem ter aspas duplas. Exemplo: "Maria"        |
| valor lógico              | logico       | <b>int</b>     | não atribuído (lixo de memória) | Na linguagem C, o valor falso é representado pelo número 0, e o valor verdadeiro é representado por um número diferente de 0.                  |

### Lista completa de tipos de dados:

<http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1256.pdf> (OFICIAL)

[https://pt.wikibooks.org/wiki/Programar\\_em\\_C/Tipos\\_de\\_dados](https://pt.wikibooks.org/wiki/Programar_em_C/Tipos_de_dados)

[https://en.wikipedia.org/wiki/C\\_data\\_types](https://en.wikipedia.org/wiki/C_data_types)

## Exemplo: declaração e atribuição de variáveis

| VisualG   | C   |
|---|---|
| <p>Algoritmo "teste"</p> <p>Var</p> <p>idade : inteiro<br/>salario, altura : real<br/>genero : caractere<br/>nome : caractere</p> <p>Inicio</p> <p>idade &lt;- 20<br/>salario &lt;- 5800.5<br/>altura &lt;- 1.63<br/>genero &lt;- "F"<br/>nome &lt;- "Maria Silva"</p> <p>escreval("IDADE = ", idade)<br/>escreval("SALARIO = ", salario:4:2)<br/>escreval("ALTURA = ", altura:4:2)<br/>escreval("GENERO = ", genero)<br/>escreval("NOME = ", nome)</p> <p>Fimalgoritmo</p> | <pre>#include &lt;stdio.h&gt; #include &lt;string.h&gt;  int main() {      int idade;     double salario, altura;     char genero;     char nome[50];      idade = 20;     salario = 5800.5;     altura = 1.63;     genero = 'F';     strcpy(nome, "Maria Silva");      printf("IDADE = %d\n", idade);     printf("SALARIO = %.2lf\n", salario);     printf("ALTURA = %.2lf\n", altura);     printf("GENERO = %c\n", genero);     printf("NOME = %s\n", nome);      return 0; }</pre> |

**NOTA:** A linguagem C aceita atribuição diretamente na declaração da variável, inclusive para texto. Por exemplo:

```
int idade = 20;
char nome[50] = "Maria Silva";
```

## OPERADORES EM C

### Operadores aritméticos

| Operador | Significado              |
|----------|--------------------------|
| +        | adição                   |
| -        | subtração                |
| *        | multiplicação            |
| /        | divisão                  |
| %        | resto da divisão ("mod") |

### Operadores comparativos

| Operador | Significado    |
|----------|----------------|
| <        | menor          |
| >        | maior          |
| <=       | menor ou igual |
| >=       | maior ou igual |
| ==       | igual          |
| !=       | diferente      |

### Operadores lógicos

| Operador | Significado |
|----------|-------------|
| &&       | e           |
|          | ou          |
| !        | não         |

## SAÍDA DE DADOS EM C

| Comando no VisualG | Comando em C | Biblioteca                            |
|--------------------|--------------|---------------------------------------|
| escreva / escreval | printf       | <code>#include &lt;stdio.h&gt;</code> |

| Tipo                                   | Placeholder de formatação          |
|--|------------------------------------|
| <b>int</b> (inteiro 16 bits)           | <code>%d</code> ou <code>%i</code> |
| <b>long int</b> (inteiro 32 bits)      | <code>%li</code>                   |
| <b>long long int</b> (inteiro 64 bits) | <code>%lli</code>                  |
| <b>float</b> (real precisão simples)   | <code>%f</code>                    |
| <b>double</b> (real precisão dupla)    | <code>%lf</code>                   |
| <b>char</b> (um único caractere)       | <code>%c</code>                    |
| <b>char[ ]</b> (texto)                 | <code>%s</code>                    |

## SAÍDA DE DADOS EM C

| Exemplo VisualG  | Exemplo C  | Resultado na tela  |
|--|--|--|
| <pre>escreva("Bom dia") escreva("Boa noite")</pre>   | <pre>printf("Bom dia"); printf("Boa noite");</pre>   | Bom diaBoa noite   |
| <pre>escreval("Bom dia") escreval("Boa noite")</pre>   | <pre>printf("Bom dia\n"); printf("Boa noite\n");</pre>   | Bom dia<br>Boa noite   |
| <pre>x, y : inteiro x &lt;- 10 y &lt;- 20 escreval(x) escreval(y)</pre>  | <pre>int x, y; x = 10; y = 20; printf("%d\n", x); printf("%d\n", y);</pre>   | 10<br>20   |
| <pre>x : real x &lt;- 2.3456 escreval(x:4:2)</pre>   | <pre>double x; x = 2.3456; printf("%.2lf\n", x);</pre>   | 2.35   |
| <pre>idade : inteiro salario : real nome : caractere sexo : caractere  idade &lt;- 32 salario &lt;- 4560.9 nome &lt;- "Maria Silva" sexo &lt;- "F"  escreval("A funcionaria ", nome, ", sexo ", sexo, ", ganha ", salario:8:2, " e tem ", idade, " anos.")</pre> | <pre>int idade; double salario; char nome[50]; char sexo;  idade = 32; salario = 4560.9; strcpy(nome, "Maria Silva"); sexo = 'F';  printf("A funcionaria %s, sexo %c, ganha %.2lf e tem %d anos\n", nome, sexo, salario, idade);</pre> | A funcionaria Maria Silva, sexo F, ganha 4560.90 e tem 32 anos |

## PROCESSAMENTO DE DADOS / CASTING EM C

| Exemplo VisualG  | Exemplo C  | Resultado na tela    |
|--|--|----------------------|
| <pre>x, y : inteiro x &lt;- 5 y &lt;- 2 * x escreval(x) escreval(y)</pre>  | <pre>int x, y; x = 5; y = 2 * x; printf("%d\n", x); printf("%d\n", y);</pre>                                       | <pre>5 10</pre>      |
| <pre>x : inteiro y : real x &lt;- 5 y &lt;- 2 * x escreval(x) escreval(y)</pre>  | <pre>int x; double y; x = 5; y = 2 * x; printf("%d\n", x); printf("%.11f\n", y);</pre>                             | <pre>5 10.0</pre>    |
| <pre>b1, b2, h, area : real b1 &lt;- 6.0 b2 &lt;- 8.0 h &lt;- 5.0 area &lt;- (b1 + b2) / 2.0 * h; escreval(area)</pre> | <pre>double b1, b2, h, area; b1 = 6.0; b2 = 8.0; h = 5.0; area = (b1 + b2) / 2.0 * h; printf("%lf\n", area);</pre> | <pre>35.000000</pre> |
| <pre>a, b, resultado : inteiro a &lt;- 5 b &lt;- 2 resultado &lt;- a \ b escreval(resultado)</pre>                     | <pre>int a, b, resultado; a = 5; b = 2; resultado = a / b; printf("%d\n", resultado);</pre>                        | <pre>2</pre>         |
| <pre>a : real b : inteiro a &lt;- 5.0 b &lt;- Int(a) escreval(b)</pre>   | <pre>double a; int b; a = 5.0; b = (int) a; printf("%d\n", b);</pre>   | <pre>5</pre>         |

## ENTRADA DE DADOS EM C

| Comando no VisualG | Comando em C  | Biblioteca                             |
|--------------------|---|--|
| leia               | <code>scanf</code> (para ler dados de tipos básicos)      | <code>#include &lt;stdio.h&gt;</code>  |
|                    | <code>fgets</code> (para ler texto até a quebra de linha) | <code>#include &lt;string.h&gt;</code> |

| Tipo   | Placeholder de formatação  |
|--|--|
| <code>int</code> (inteiro 16 ou 32 bits)     | <code>%d</code> ou <code>%i</code>   |
| <code>long int</code> (inteiro 32 bits)      | <code>%li</code>   |
| <code>long long int</code> (inteiro 64 bits) | <code>%lli</code>  |
| <code>float</code> (real precisão simples)   | <code>%f</code>  |
| <code>double</code> (real precisão dupla)    | <code>%lf</code>   |
| <code>char</code> (um único caractere)       | <code>%c</code><br><br>Se houver uma quebra de linha pendente na entrada padrão, é preciso limpar antes:<br><code>fseek(stdin,0,SEEK_END);</code>  |
| <code>char[ ]</code> (texto)                 | <code>%s</code><br><br><code>%s</code> só funciona para um texto contíguo<br><br>Para ler até o fim da linha, use: <code>fgets</code><br>* Se houver uma quebra de linha pendente na entrada padrão, é preciso limpar a entrada antes. |



## Para ler um texto de tamanho N até a quebra de linha

```
void ler_texto(char *buffer, int length) {  
    fgets(buffer, length, stdin);  
    strtok(buffer, "\n");  
}
```

### Exemplo:

```
char nomeCompleto[50];  
  
printf("Digite seu nome completo: ");  
ler_texto(nomeCompleto, 50);
```

## Comando para limpeza de buffer de entrada

```
void limpar_entrada() {  
    char c;  
    while ((c = getchar()) != '\n' && c != EOF) {}  
}
```

**QUANDO USAR:** quando você for ler um texto até a quebra de linha (ou um caractere char), mas antes o seu programa já leu algum outro dado e deixou uma quebra de linha pendente. Veja exemplo completo desta aula.

## ENTRADA DE DADOS EM C

| Exemplo VisualG  | Exemplo C   |
|--|---|
| <p>Algoritmo "teste_entrada"</p> <p>Var</p> <p>    salario1, salario2 : real<br/>         nome1, nome2 : caractere<br/>         idade : inteiro<br/>         sexo : caractere</p> <p>Inicio</p> <p>    escreva("Nome da primeira pessoa: ")<br/>         leia(nome1)<br/>         escreva("Salario da primeira pessoa: ")<br/>         leia(salario1)</p> <p>    escreva("Nome da segunda pessoa: ")<br/>         leia(nome2)<br/>         escreva("Salario da segunda pessoa: ")<br/>         leia(salario2)</p> <p>    escreva("Digite uma idade: ")<br/>         leia(idade)<br/>         escreva("Digite um sexo (F/M): ")<br/>         leia(sexo)</p> <p>    escreval("Nome 1: ", nome1)<br/>         escreval("Salario 1: ", salario1:4:2)<br/>         escreval("Nome 2: ", nome2)<br/>         escreval("Salario 2: ", salario2:4:2)<br/>         escreval("Idade: ", idade)<br/>         escreval("Sexo: ", sexo)</p> <p>Fimalgoritmo</p> | <pre>#include &lt;stdio.h&gt; #include &lt;string.h&gt;  void limpar_entrada() {     char c;     while ((c = getchar()) != '\n' &amp;&amp; c != EOF) {} }  void ler_texto(char *buffer, int length) {     fgets(buffer, length, stdin);     strtok(buffer, "\n"); }  int main() {     double salario1, salario2;     char nome1[50], nome2[50];     int idade;     char sexo;      printf("Nome da primeira pessoa: ");     ler_texto(nome1, 50);     printf("Salario da primeira pessoa: ");     scanf("%lf", &amp;salario1);      printf("Nome da segunda pessoa: ");     limpar_entrada();     ler_texto(nome2, 50);     printf("Salario da segunda pessoa: ");     scanf("%lf", &amp;salario2);      printf("Digite uma idade: ");     scanf("%d", &amp;idade);     printf("Digite um sexo (F/M): ");     limpar_entrada();     scanf("%c", &amp;sexo);      printf("Nome 1: %s\n", nome1);     printf("Salario 1: %.2lf\n", salario1);     printf("Nome 2: %s\n", nome2);     printf("Salario 2: %.2lf\n", salario2);     printf("Idade: %d\n", idade);     printf("Sexo: %c\n", sexo);      return 0; }</pre> |

## **COMO CRIAR UM PROJETO NO CODE BLOCKS**

### **Por que criar um projeto?**

- Um projeto pode conter vários arquivos relacionados
- Algumas ferramentas da IDE só funcionam em projetos: Debugger

### **Passos:**

- File -> New -> Project
- Console Application -> Go
- Next -> (escolha a linguagem) -> Next
- (dê um nome para o projeto) -> (escolha a pasta) -> Finish

### **Para abrir o projeto novamente:**

- Abra pelo arquivo .cbp

## **COMO EXECUTAR O DEBUGGER NO CODE BLOCKS**

**ATENÇÃO:** o debug NÃO FUNCIONA para arquivos isolados. Seu programa deve estar dentro de um PROJETO.

### **PASSOS PARA CONFIGURAR O DEBUGGER:**

- Settings -> Compiler -> Toolchain Executables
  - Debugger: GDB/CDB debugger: default
- Settings -> Debugger -> GDB/CDB debugger -> Default
  - Executable path: C:\MinGW\bin\gdb.exe

### **COMANDOS DO DEBUGGER:**

- Habilitar/desabilitar breakpoint: **F5**
- Iniciar o debug: **F8**
- Rodar um passo: **F7**
- Parar o debug: **SHIFT+F8**
- Mostrar variáveis: **Debug -> Debugging windows -> Watches**

## ESTRUTURA CONDICIONAL EM C

| Simple   | Composta  | Encadeamento   |
|--|---|--|
| <pre>if (condição) {     comando1     comando2 }</pre> | <pre>if (condição) {     comando1     comando2 } else {     comando3     comando4 }</pre> | <pre>if (condição1) {     comando1     comando2 } else if (condição2) {     comando3     comando4 } else {     comando5     comando6 }</pre> |

| Exemplo VisualG   | Exemplo C  |
|---|--|
| <p>Algoritmo "teste_condicional"</p> <p>Var</p> <p>    hora : inteiro</p> <p>Inicio</p> <p>    escreva("Digite uma hora do dia: ")</p> <p>    leia(hora)</p> <p>    se hora &lt; 12 entao</p> <p>        escreval("Bom dia!")</p> <p>    senao</p> <p>        escreval("Boa tarde!")</p> <p>    fimse</p> <p>Fimalgoritmo</p> | <pre>#include &lt;stdio.h&gt;  int main() {     int hora;     printf("Digite uma hora do dia: ");     scanf("%d", &amp;hora);      if (hora &lt; 12) {         printf("Bom dia!\n");     }     else {         printf("Boa tarde!\n");     }      return 0; }</pre> |

## ESTRUTURA ENQUANTO EM C

| Sintaxe  | Regra                                      |
|--|--|
| <pre>while (<b>condição</b>) {     <b>comando1</b>     <b>comando2</b> }</pre> | <p>V: executa e volta<br/>F: pula fora</p> |

| Exemplo VisualG   | Exemplo C   |
|---|---|
| <p>Algoritmo "teste_enquanto"</p> <p>Var</p> <p>    x, soma : inteiro</p> <p>Inicio</p> <p>    soma &lt;- 0</p> <p>    escreva("Digite o primeiro numero: ")</p> <p>    leia(x)</p> <p>    enquanto x &lt;&gt; 0 faca</p> <p>        soma &lt;- soma + x</p> <p>        escreva("Digite outro numero: ")</p> <p>        leia(x)</p> <p>    fimenquanto</p> <p>    escreval("SOMA = ", soma)</p> <p>Fimalgoritmo</p> | <pre><b>#include</b> &lt;stdio.h&gt;  <b>int</b> main() {     <b>int</b> x, soma;      soma = 0;     printf("Digite o primeiro numero: ");     scanf("%d", &amp;x);      <b>while</b> (x != 0) {         soma = soma + x;         printf("Digite outro numero: ");         scanf("%d", &amp;x);     }      printf("SOMA = %d\n", soma);      <b>return</b> 0; }</pre> |

## ESTRUTURA PARA EM C

### Sintaxe / regra

Executa somente  
na primeira vez

V: executa e volta  
F: pula fora

Executa toda vez depois  
de voltar

```
for ( início ; condição ; incremento) {
    comando1
    comando2
}
```

### Exemplo VisualG

Algoritmo "teste\_para"

Var

N, i, x, soma : inteiro

Inicio

escreva("Quantos numeros serao digitados? ")  
leia(N)

soma <- 0  
para i de 1 ate N faca  
    escreva("Digite um numero: ")  
    leia(x)  
    soma <- soma + x  
fimpara

escreval("SOMA = ", soma)

Fimalgoritmo

### Exemplo C

```
#include <stdio.h>

int main()
{
    int N, i, x, soma;

    printf("Quantos numeros serao digitados? ");
    scanf("%d", &N);

    soma = 0;
    for (i = 1; i <= N; i++) {
        printf("Digite um numero: ");
        scanf("%d", &x);
        soma = soma + x;
    }

    printf("SOMA = %d\n", soma);

    return 0;
}
```

**ESTRUTURA DO-WHILE EM C (variante da estrutura REPITA-ATÉ)**

| Sintaxe   | Regra  |
|---|--|
| <pre>do {<br/>    comando 1<br/>    comando 2<br/>} while (condição);</pre> | <p><b>V:</b> volta<br/><b>F:</b> pula fora</p> |

| Exemplo VisualG   | Exemplo C   |
|---|---|
| <p><b>Algoritmo</b> "exemplo_repita_ate"</p> <p><b>Var</b><br/>C, F : real<br/>resp : caractere</p> <p><b>Inicio</b></p> <p>    <b>repita</b><br/>        escreva("Digite a temperatura em Celsius: ")<br/>        leia(C)<br/>        F &lt;- 9.0 * C / 5.0 + 32.0<br/>        escreval("Equivalente em Fahrenheit: ", F:6:1)<br/>        escreva("Deseja repetir (s/n)? ")<br/>        leia(resp)<br/>    <b>ate</b> resp &lt;&gt; "s"</p> <p><b>Fimalgoritmo</b></p> | <pre>#include &lt;stdio.h&gt;<br/><br/>void limpar_entrada() {<br/>    char c;<br/>    while ((c = getchar()) != '\n' &amp;&amp; c != EOF) {}<br/>}<br/><br/>int main()<br/>{<br/>    double C, F;<br/>    char resp;<br/><br/>    do {<br/>        printf("Digite a temperatura em Celsius: ");<br/>        scanf("%lf", &amp;C);<br/>        F = 9.0 * C / 5.0 + 32.0;<br/>        printf("Equivalente em Fahrenheit: %.1lf\n", F);<br/>        printf("Deseja repetir (s/n)? ");<br/>        limpar_entrada();<br/>        scanf("%c", &amp;resp);<br/>    } while (resp == 's');<br/><br/>    return 0;<br/>}</pre> |



## VETORES EM C

| Exemplo VisualG  | Exemplo C  |
|--|--|
| <p>Algoritmo "teste_vetor"</p> <p>Var</p> <p>vet: vetor [0..9] de real<br/>N, i : inteiro</p> <p>Inicio</p> <p>escreva("Quantos numeros voce vai digitar? ")<br/>leia(N)</p> <p>para i de 0 ate N-1 faca<br/>    escreva("Digite um numero: ")<br/>    leia(vet[i])<br/>fimpara</p> <p>escreval<br/>escreval("NUMEROS DIGITADOS:")<br/>para i de 0 ate N-1 faca<br/>    escreval(vet[i]:8:1)<br/>fimpara</p> <p>Fimalgoritmo</p> | <pre>#include &lt;stdio.h&gt;  int main() {     int N, i;      printf("Quantos numeros voce vai digitar? ");     scanf("%d", &amp;N);      double vet[N];      for (i = 0; i &lt; N; i++) {         printf("Digite um numero: ");         scanf("%lf", &amp;vet[i]);     }      printf("\nNUMEROS DIGITADOS:\n");     for (i = 0; i &lt; N; i++) {         printf("%.1lf\n", vet[i]);     }      return 0; }</pre> |

**MATRIZES EM C**

| Exemplo VisualG  | Exemplo C   |
|--|---|
| <pre>Algoritmo "teste_matriz"  Var     mat: vetor [0..4, 0..4] de inteiro     M, N, i, j : inteiro  Inicio     escreva("Quantas linhas vai ter a matriz? ")     leia(M)     escreva("Quantas colunas vai ter a matriz? ")     leia(N)      para i de 0 ate M-1 faca         para j de 0 ate N-1 faca             escreva("Elemento [", i, ",", j, "]: ")             leia(mat[i, j])         fimpara     fimpara      escreval     escreval("MATRIZ DIGITADA:")     para i de 0 ate M-1 faca         para j de 0 ate N-1 faca             escreva(mat[i, j])         fimpara     escreval     fimpara Fimalgoritmo</pre> | <pre>#include &lt;stdio.h&gt;  int main() {     int M, N, i, j;      printf("Quantas linhas vai ter a matriz? ");     scanf("%d", &amp;M);     printf("Quantas colunas vai ter a matriz? ");     scanf("%d", &amp;N);      int mat[M][N];      for (i = 0; i &lt; M; i++) {         for (j = 0; j &lt; N; j++) {             printf("Elemento [%d,%d]: ", i, j);             scanf("%d", &amp;mat[i][j]);         }     }      printf("\nMATRIZ DIGITADA:\n");     for (i = 0; i &lt; M; i++) {         for (j = 0; j &lt; N; j++) {             printf("%d ", mat[i][j]);         }         printf("\n");     }      return 0; }</pre> |