

Flávio Augusto de Freitas  
Introdução à Programação em Linguagem C/C++

<http://flavioaf.blogspot.com>

# C/C++

Tutorial 4 (usando Dev-C++ versão 4.9.9.2)



2011

# 1 INTRODUÇÃO

Esta série de tutoriais foi escrita usando o **Microsoft Windows 7 Ultimate** e o **Bloodshed Dev-C++** versão 4.9.9.2, que pode ser baixada em <http://www.bloodshed.net>.

Qualquer exemplo de programa nesta série de tutoriais, implica em um novo projeto, dentro de uma nova pasta. Se não souber como criar uma nova pasta, volte e refaça o tutorial 1.

## 2 RESPOSTAS DOS EXERCÍCIOS PROPOSTOS NO TUTORIAL 3

- a) Crie um programa para ler um string e imprimir cada caractere em um linha. Exemplo: "teste" deverá ficar

t  
e  
s  
t  
e

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
```

```
    char str[20];
    int i;
```

```
    printf("String (por exemplo: \"teste\"): ");
    scanf("%s", &str);
```

```
    for(i = 0; i < strlen(str); i++)
        printf("%c\n", str[i]);
```

```
    system("PAUSE");
    return EXIT_SUCCESS;
```

```
}
```

- b) Faça um programa para ler dois strings e informar qual dos dois têm mais caracteres.

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
```

```
    char str1[20], str2[20];
    int tam1, tam2;
```

```
    printf("Informe primeiro string: ");
    scanf("%s", str1);
    printf("\n\nInforme segundo string: ");
    scanf("%s", str2);
```

```
    tam1 = strlen(str1);
    tam2 = strlen(str2);
```

```
    if(tam1 > tam2)
```

```
        printf("\n\"%s\" (%d letras) eh mais longa
que \"%s\" (%d letras)\n\n", str1, tam1, str2,
tam2);
```

```
    else if(tam2 > tam1)
```

```
        printf("\n\"%s\" (%d letras) eh mais longa
que \"%s\" (%d letras)\n\n", str2, tam2, str1,
tam1);
```

```
    else printf("\n\"%s\" (%d letras) eh tao
comprida quanto \"%s\" (%d letras)\n\n",
str1, tam1, str2, tam2);
```

```
    system("PAUSE");
    return EXIT_SUCCESS;
```

```
}
```

- c) Escreva um programa para ler um string e convertê-lo todo para minúsculo. Exemplo: “Isto eh um String” deverá ficar “isto eh um string”.

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
    char str[20];
    int i;

    printf("Informe um string em minusculo: ");
    scanf("%s", &str);

    printf("\n\nString original : %s\n", str);

    for(i = 0; i < strlen(str); i++)
        str[i] = tolower(str[i]);

    printf("String maiuscula: %s\n\n", str);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

- d) Escreva outro programa para ler um string e convertê-lo todo para maiúsculo. Exemplo: “Isto eh um string” deverá ficar “ISTO EH UM STRING”.

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
    char str[20];
    int i;

    printf("Informe um string em minusculo: ");
    scanf("%s", &str);

    printf("\n\nString original : %s\n", str);

    for(i = 0; i < strlen(str); i++)
        str[i] = toupper(str[i]);

    printf("String maiuscula: %s\n\n", str);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

- e) Crie um programa para ler um string e converter as iniciais para maiúsculo e o restante para minúsculo. Exemplo: “isto eh um string” deverá ficar “Isto Eh Um String”.

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
    char str[20];
    int i;

    printf("Informe um string: ");
    scanf("%s", &str);

    str[0] = toupper(str[0]);
    for(i = 0; i < strlen(str) - 1; i++) {
        if((str[i] == ' ') && (strlen(str) > i) &&
            (!isspace(str[i + 1])))
            str[i + 1] = toupper(str[i + 1]);
        else str[i + 1] = tolower(str[i + 1]);
    }

    printf("String com iniciais maiusculas: \n%s\n\n", str);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

- f) Programe um código para ler dois strings, cada um com exatamente 10 caracteres, que transfira o 3º, 4º e 5º caracteres do string 2 para o string 1.

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
    char str1[10], str2[10];

    strcpy(str1, "");
    while(strlen(str1) < 10) {
        printf("Insira um string com 10
caracteres: ");
        scanf("%[^\\n]", &str1);
    }

    strcpy(str2, "");
    while(strlen(str2) < 10) {
        printf("\\n\\nInsira outro string com 10
caracteres: ");
        scanf("%s", &str2);
    }

    system("CLS"); // limpar tela

    printf("\\n\\nString1 original : %s\\n", str1);
    printf("String2 original : %s\\n", str2);

    for(int i = 2; i < 5; i++) str2[i] = str1[i];
    printf("String2 modificado: %s\\n", str2);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

- g) Crie um código que leia um string e inverta-o.  
Exemplo: "roma" deverá ficar "amor".

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
    char str[20], aux;
    int i;

    printf("Informe uma string: ");
    scanf("%[^\\n]", &str);

    for(i = 0; i < strlen(str) / 2; i++) {
        aux = str[i];
        str[i] = str[strlen(str) - i - 1];
        str[strlen(str) - i - 1] = aux;
    }

    printf("String invertido: %s\\n\\n", str);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

- h) Escreva um programa que leia um string e verifique se ele é palíndromo. Exemplo: “osso” é palíndromo, pois o inverso – “osso” -é o mesmo string.

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
    char str1[20], str2[20], aux;
    int i;

    printf("Informe uma string: ");
    scanf("%[^\n]", &str1);
    strcpy(str2, str1);

    for(i = 0; i < strlen(str1) / 2; i++) {
        aux = str1[i];
        str1[i] = str1[strlen(str1) - i - 1];
        str1[strlen(str1) - i - 1] = aux;
    }

    if(!strcmp(str1, str2))
        printf("String %s eh palindromo.\n\n",
str2);
    else printf("Nao eh palindromo.\n\n");

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

### 3 OPERAÇÕES MATEMÁTICAS

Em C, fazer operações matemáticas simples é bastante fácil e intuitivo. Por exemplo, se quisermos que uma variável contenha o resultado da conta  $123 + 912$ , fazemos assim:

```
var = 123 + 912;
```

Os operadores aritméticos básicos são cinco: + (adição), - (subtração), \* (multiplicação), / (divisão) e % (resto de divisão inteira).

Outro exemplo:

```
int a = 15;
int b = 72;
int c = a * b; /* c valerá 15×72 */
```

Podemos usar mais de um operador na mesma expressão. A precedência é igual à usada na matemática comum:

```
a = 2 + 4*10; // = 42, o mesmo que 2 + (4*10)
a = 2 + 40/2 + 5; // = 27, o mesmo que 2 + (40/2) + 5
```

Você pode usar parênteses, como em expressões matemáticas normais:

```
a = (2 + 4)*10; /* retornará 60 */
a = (2 + 40)/(2 + 5); /* retornará 6 */
```

Note que uma operação entre números inteiros sempre retornará um número inteiro. Isso é evidente para a adição, subtração e multiplicação. Mas em uma divisão de inteiros, por exemplo,  $3/2$ , a expressão retornará apenas a parte inteira do resultado, ou seja, 1.

Se quisermos um resultado não-inteiro, um dos operandos deve ser não-inteiro. Nesse exemplo, poderíamos usar  $3.0/2$  ou  $3/2.0$ , ou mesmo  $3./2$  ou  $(1.0 * 3)/2$ , pois, em C, uma operação envolvendo um número não-inteiro sempre terá como resultado um número real.

Note que em C o separador decimal é o ponto e não a vírgula.

O seguinte exemplo poderia surpreender, pois o programa dirá que o valor de f continua sendo 3.

```
#include <cstdlib>
#include <iostream>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
    int i = 5;
    int j = 2;
    float f = 3.0;

    f = f + j / i;
    printf("O valor de f é %f", f);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

Mas, segundo a precedência dos operadores,  $j / i$  é calculado primeiro, e como ambos os valores são do tipo inteiro, o valor dessa expressão é zero.

#### Abreviações

Alguns tipos de atribuições são bastante comuns, e por isso foram criadas abreviações. Por exemplo, é muito comum incrementar em uma unidade o valor de uma variável (em loops, por exemplo). Em vez de escrever  $var = var + 1$ , podemos escrever simplesmente  $var++$ . Da mesma maneira, existe o operador de

decremento, que decrementa em uma unidade o valor da variável: `var--` (equivalente a `var = var - 1`).

Os operadores de decremento e incremento também podem ser utilizados antes do nome da variável. Isso significa que estas duas instruções são equivalentes:

```
var++;
++var;
```

Agora vamos supor que você use em seu programa uma variável que aumenta de 10 em 10 unidades. É claro que usar `var++` dez vezes não abreviaria nada. Em vez disso, existe a abreviação `var += 10`.

Genericamente, para qualquer dos cinco operadores aritméticos `op`, vale a abreviação:

```
var = var op num;
var op = num;
```

Ou seja, os seguintes pares são equivalentes:

```
x *= 12;      x = x * 12;
x /= 10;      x = x / 10;
x -= 2;       x = x - 2;
x %= 11;      x = x % 11;
```

Este exemplo clarifica o uso dos operadores de incremento:

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    int a, b;

    a = b = 5;
    printf("%d\n", ++a + 5);
    printf("%d\n", a);
    printf("%d\n", b++ + 5);
    printf("%d\n", b);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

O resultado que você deve obter ao executar o exemplo é:

```
11
6
10
6
```

Esse resultado mostra que `++var` e `var++` não são a mesma coisa se usados como uma expressão. Quando usamos os operadores na forma prefixal (antes do nome da variável), o valor é retornado depois de ser incrementado; na forma sufixal, o valor é retornado e depois incrementado. O mesmo vale para o operador de decremento.

*E o que aconteceria se você escrevesse algo como o seguinte?*

```
printf("%d\n", a / ++a);
```

A resposta é: não sabemos. Segundo o padrão C, o resultado disso é indefinido (o que significa que pode variar de um compilador para outro). Não existe uma regra sobre avaliar primeiro o numerador ou o denominador de uma fração. Ou seja, não use uma variável mais de uma vez numa expressão se usar operadores que a modificam.

Talvez você tenha achado estranha a linha:

```
a = b = 5;
```

Isso é possível porque atribuições são feitas da direita para a esquerda e uma instrução de atribuição é também uma expressão que retorna o valor atribuído. Ou seja, a expressão `b = 5` retornou o valor 5, que foi usado pela atribuição `a = (b = 5)`, equivalente a `a = 5`.

## 4 PROGRAMAS DE EXEMPLO

### Programa que utiliza os operadores aritméticos

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    int x,y,soma,sub;
    float modulo,div,mult;

    x = 69;
    y = 24;

    // usando os operadores aritmeticos:
    soma = x + y;
    sub = x - y;

    mult = (float) x*y; // aqui é usado
    div = (float) x/y; // o operador cast

    modulo = (x % y);
    printf("%d + %d = %d\n",x,y,soma);
    //imprime a soma dos dois números
    printf("%d - %d = %d\n",x,y,sub);
    //imprime a subtração
    printf("%d x %d = %f\n",x,y,mult);
    //imprime a multiplicação
    printf("%d / %d = %f\n",x,y,div);
    //imprime a divisão
    printf("%d mod %d = %f\n",x,y,modulo);
    //imprime o resto da divisão

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

### Operadores de Incremento Pré-fixado

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    int var, var2;

    var = 5;
    var2 = ++var;
    printf("var = %d\n", var);
    printf("var2= %d\n", var2);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

### Operadores de Incremento Pós-fixado

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    int var, var2;

    var = 5;
    var2 = var++;
    printf("var = %d\n", var);
    printf("var2= %d\n", var2);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

### Resolvendo uma expressão simples

Seja resolver a expressão  $y = \frac{x + \frac{x}{3}}{x - \frac{x}{7}}$

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    float x, y;

    printf("Forneca o valor de x: ");
    scanf("%f",&x);
    printf("\n\n");

    y = (x + x/3)/(x - x/7);
    printf("          x\n");
    printf("      x + ---\n");
    printf("          3\n");
    printf("y = ----- = %10.4f\n",y);
    printf("          x\n");
    printf("      x - ---\n");
    printf("          7\n");

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

## Resolvendo uma equação mais complexa

Seja resolver  $y = \frac{x^2 - 3x + \frac{x}{2}}{\frac{x}{2} - 3x - x^5}$

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    float x, y;

    printf("Forneca o valor de x: ");
    scanf("%f",&x);
    printf("\n\n");

    y = x*x - 3*x + x/2;
    y /= (x/2 - 3*x - x*x*x*x*x);
    printf("      2      x\n");
    printf("      x  - 3x + ---\n");
    printf("                2\n");
    printf("y = ----- = %8.4f\n",y);
    printf("      x      5\n");
    printf("    --- - 3x - x\n");
    printf("      2      \n");

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

## Porcentagens

### Exemplo 1

Suponhamos que um produto que custe R\$ 178,00 sofra um acréscimo de 15%. Qual o valor final do produto?

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    float valor = 178.00; // valor original
    float percentual = 15.0 / 100.0; // 15%
    float vfinal = valor + (percentual*valor);

    printf("Valor final = %f\n\n", vfinal);
    // 204.70

    system("PAUSE"); // pausa o programa
    return EXIT_SUCCESS;
}
```

### Exemplo 2

Um produto, cujo valor original era de R\$ 250,00, teve um desconto de 8%. Qual foi seu valor final?

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    float valor = 250.00; // valor original
    float percentual = 8.0 / 100.0; // 8%
    float vfinal = valor - (percentual*valor);

    printf("Valor final = %f\n\n", vfinal);
    // 230.00

    system("PAUSE"); // pausa o programa
    return EXIT_SUCCESS;
}
```

### Exemplo 3

Em um concurso de perguntas e respostas, um jovem acertou 72 das 90 perguntas apresentadas. Qual foi a porcentagem de acertos? E a porcentagem de erros?

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    float perguntas = 90.0;
    float acertos = 72.0;

    acertos *= 100.0 / perguntas;
    printf("% de acertos: %f%\n", acertos);
    printf("% de erros: %f%\n", 100 - acertos);
    // Os resultados serão 80% e 20%

    system("PAUSE"); // pausa o programa
    return EXIT_SUCCESS;
}
```

## 5 EXERCÍCIOS PROPOSTOS

- Elabore um programa para ler dois números inteiros e calcular sua soma.
- Elabore um programa para ler dois números inteiros e calcular seu produto.
- Crie um programa que leia dois números inteiros e calcule seu produto de forma iterativa, ou seja, usando um comando for e



somas sucessivas. Por exemplo:  $3 \times 4 = 3 + 3 + 3 + 3$ .

- d) Faça um programa que leia dois números quaisquer e a operação desejada (+, -, \*, /) e informe o resultado do cálculo escolhido pelo usuário.
- e) Elabore um programa para calcular o quadrado e o cubo dos números entre 0 e 1 em intervalos de 0,1.
- f) Sabendo-se que a fórmula dos juros simples é  $j = C.i.n$ , onde  $j$  é o valor dos juros calculados;  $C$  é o capital sobre o qual os juros são calculados;  $i$  é o valor da taxa percentual na forma decimal ( $1\% = 0,01$  em valor decimal);  $n$  é o período, crie um programa para ler o valor do capital em reais, a taxa mensal em porcentagem (por exemplo, 5,3 significa 5,3%, mas o programa deve usar 0,053) e o período em meses e calcular o valor dos juros devidos. Por exemplo:  $C = 1000$ ,  $i = 5\%$ ,  $n = 3 \Rightarrow j = 1000.0,05.3 \Rightarrow j = 150,00$ .
- g) Sabe-se que o montante é o capital acrescido dos juros, ou mais simplesmente, aplicamos a fórmula  $M = C(1 + i.n)$  para juros simples, onde  $M$  é o montante em reais;  $C$  é o capital em reais;  $i$  é o valor da taxa percentual na forma decimal ( $1,5\% = 0,015$  em valor decimal);  $n$  é o período em meses. Crie um programa para ler o capital, a taxa e o período e calcular o valor do montante a juros simples. Por exemplo:  $C = 1000$ ,  $i = 5\%$ ,  $n = 3 \Rightarrow M = 1000(1 + 0,05.3) \Rightarrow M = 1150,00$ .
- h) Escreva um código para um programa que leia um número inteiro e calcule todos os seus múltiplos de 1 até 40. Por exemplo:  $n = 5 \Rightarrow 5, 10, 15, 20, \dots, 200$ .

## 6 TERMINAMOS

Terminamos por aqui. Saia do Dev-C++ e corra para o próximo tutorial.