RONEI NUNES RIBEIRO - MSC

LÓGICA DE PROGRAMAÇÃO

**INTRODUÇÃO À LINGUAGEM C**

**CONCEITOS BÁSICOS:**

* **linguagem de programação:** Uma Linguagem de Programação é um método padronizado para expres- sar instruções para um computador. (LAUREANO, 2005, p. 4).
* **programas:** Um programa de computador é uma cole- ção de instruções que descrevem uma tarefa a ser reali- zada por um computador. (LAUREANO, 2005, p. 4).
* **compilador:** programa que traduz algoritmos cons- truídos em uma determinada linguagem de programa- ção para arquivos em linguagem de máquina, ou seja, possíveis de serem executados em computadores.

**VARIÁVEIS EM C**

Constantes e variáveis alocam espaço em memória e são utilizadas para armazenar valores necessários à execução do programa. A diferença entre constantes e variáveis está no fato de que o valor de uma constante nunca se altera, enquanto o valor de uma variável pode mudar durante a execução do programa.

A declaração de variáveis em C é muito parecida com a forma que foi aprensetaddo em Portugol, ou seja, indicamos o tipo da variável e, em seguida, o nome da mesma.

O tipo da variável define, além do tipo de dado que ela pode armazenar, o tamanho do espaço de memória que deve ser alocado para a mesma. O tamanho do espaço de memória é medido em uma unidade chamada byte. Abaixo é apresentada uma tabela que traz os tipos de variáveis existentes em C, informando para cada um o tipo de dados que pode ser armazenado (fazendo uma comparação com o Portugol) e o tamanho do espaço de memória reservado:

|  |  |  |
| --- | --- | --- |
| Tipo de Variável em C | Valores a  serem armazenados | Tamanho em Bytes |
| char | Permite armazenar um caractere alfanumérico. Equivalente ao tipo caractere de Portugol. | 1 |
| int | Permite armazenar números inteiros positivos ou negativos. Equivalente ao tipo inteiro de Portugol. | 2 |
| float | Permite armazenar valores numéricos reais, ou seja, números com ponto decimal. Equivalente ao tipo real de Portugol. | 4 |

Quanto aos nomes de variáveis, o primeiro caractere do nome deve ser uma letra e os demais podem ser letras, números ou o caractere underline (\_ ).

Vale lembrar que em C há distinção entre caracteres maiúsculos e minúsculos. Assim, caso você declare uma variável com caracteres maiúsculos no nome, e quando for utilizá-la escrever seu nome utilizando apenas caracteres minúsculos, ocorrerá um erro de compilação. Assim, para evitar erros desse tipo, aconselhamos evitar o uso de caracteres maiúsculos nos nomes das variáveis, apesar de seu uso ser permitido.

Após declarar uma variável, pode-se atribuir um valor a ela através da utilização do comando de atribuição igual (=). Em C, a atribuição pode ser feita em qualquer ponto do programa após a criação da variável, mas também é permitido fazer uma atribuição na mesma linha em que é feita a declaração.

Os valores atribuídos a variáveis do tipo char devem estar sempre entre aspas. Também é importante ressaltar que o separador decimal utilizado em variáveis do tipo float é o ponto (.) e não a vírgula (,) como costumamos utilizar no Brasil. Assim, se queremos atribuir a uma variável o valor 552,35 devemos utilizar 552.35.

Veja os exemplos:

**float salario = 552.35; //a variável salario foi declarada e recebeu o valor 552.35**

**char sexo = ‘F’; //a variável sexo do tipo char foi declarada e recebeu o valor F F (note as aspas)**

**salario = 625.23; //a variável salario recebeu o valor 625.23**

COMANDO DE SAÍDA DE DADOS – PRINTF ( )

A função printf é a função de saída de dados em C. Com o printf imprimimos mensagens na tela.

A função printf ( ) usa o caractere de percentual (%) seguido de uma letra para identificar o formato de impressão. Abaixo são exibidos os principais códigos de formatação utilizados no printf ( ):

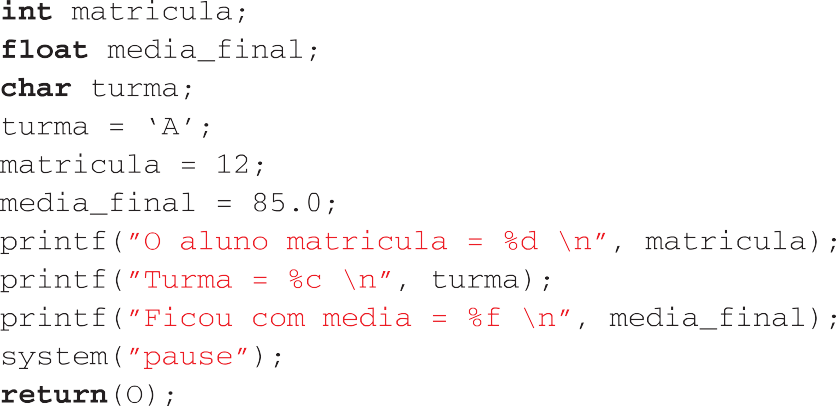
**%c: usado quando a função for exibir apenas um caractere (tipo char).**

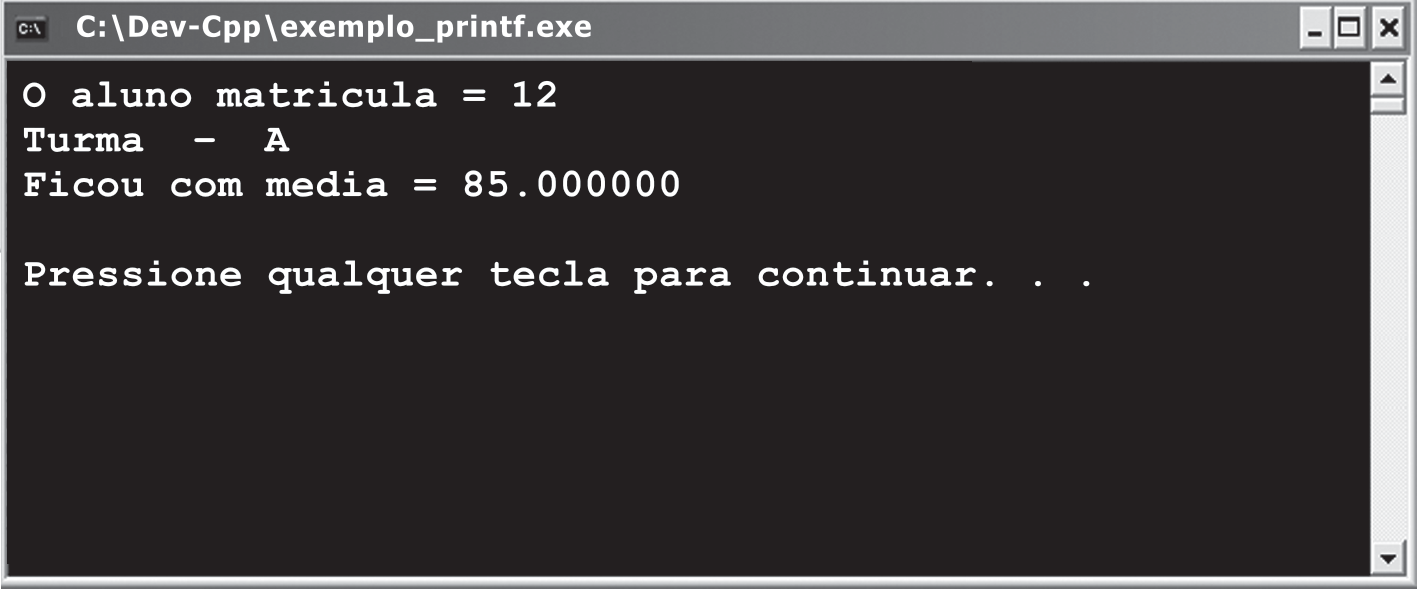
**%f: usado quando a função for exibir número com ponto flutuante (tipo float).**

**%s: usado quando a função for exibir uma cadeia de caracteres, ou seja, uma ou várias palavras (tipo char[ ]).**

**%d: usado quando a função for exibir um número inteiro (tipo int).**

Abaixo segue um exemplo com printf e diferentes tipos de dados:





Nesse exemplo utilizamos o %d quando imprimimos a variável matricula, que é do tipo int, %c para a variável turma, que é do tipo char e %f para imprimir a variável media\_final, do tipo float. Os caracteres \n que aparecem no final de cada printf são utilizados para pular uma linha; ou seja: caso não tivéssemos utilizado \n, todas as mensagens seriam impressas na mesma linha.

**COMANDO DE ENTRADA DE DADOS – SCANF ( ).**

Com o scanf ( ) lemos entradas de dados através do teclado.

A exemplo do printf ( ), o scanf ( ) também utiliza os códigos de formatação. Enquanto no printf ( ) esses códigos eram utilizados para indicar o formato dos dados a serem escritos, no scanf ( ) esses mesmos códigos indicam o formato dos dados a serem lidos. A tabela abaixo exibe os códigos de formatação utilizados no scanf ( ). Note a semelhança com a tabela de códigos do printf ( ).

%c: usado quando a função for armazenar um caractere (tipo char).

%f: usado quando a função for armazenar um número com ponto flutuante, aquele valor com vírgula (tipo float).

%s: usado quando a função for armazenar uma cadeia de caracteres, ou seja, uma ou várias palavras (tipo char[ ]).

%d: usado quando a função for armazenar um número inteiro (tipo int).

COMENTÁRIOS

Quando desenvolvemos programas, devemos colocar textos que expliquem o raciocínio seguido durante seu desenvolvimento para que outras pessoas, ou nós mesmos, ao ler o programa mais tarde, não tenhamos dificuldades em entender sua lógica. Esses textos são chamados de comentários.

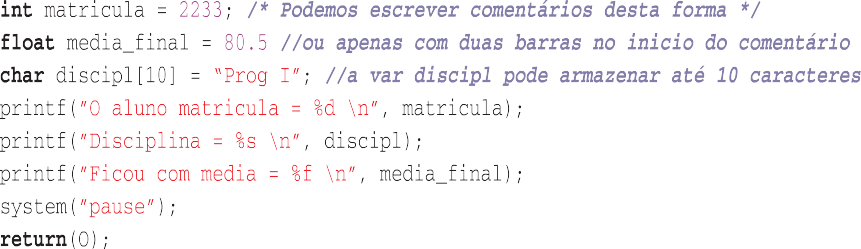
Os comentários podem aparecer em qualquer lugar do programa. Em C, há dois tipos de comentários: os comentários de linha e os comentários de bloco.

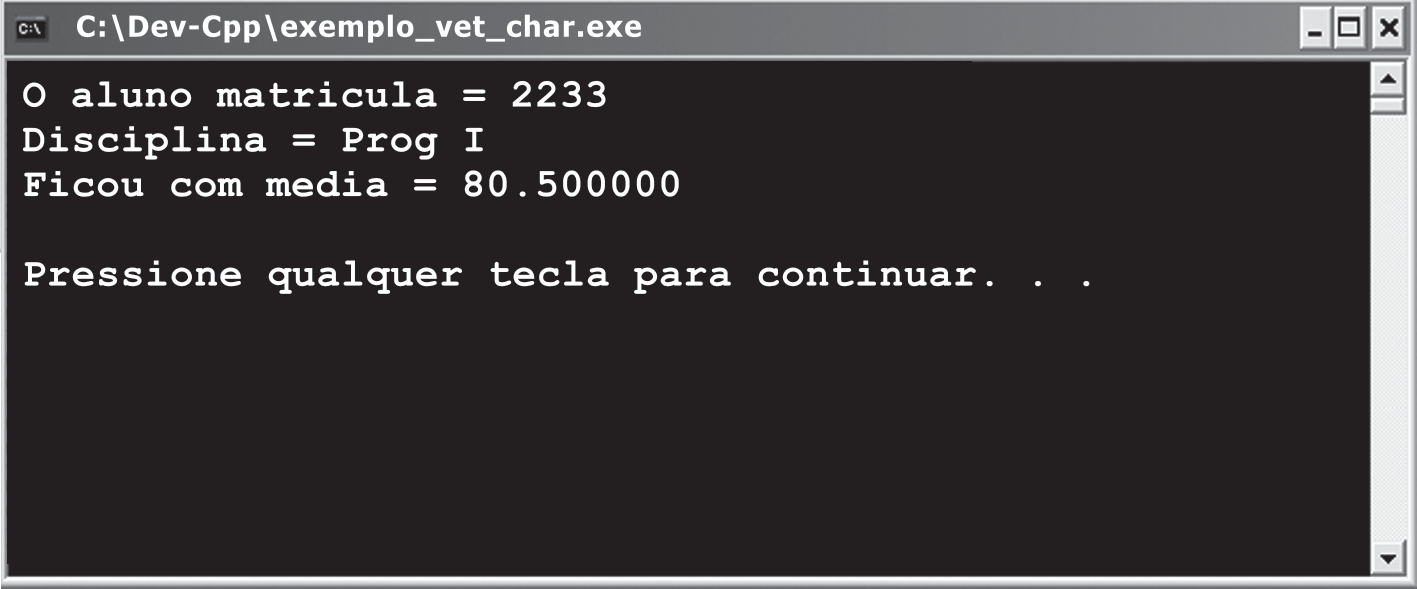
Os comentários de linha são identificados pelo uso de //. Assim, quando usamos // em uma linha, tudo o que estiver nessa linha depois do // são considerados comentários.

Os comentários de bloco são iniciados por /\* e finalizados por \*/. Tudo o que estiver entre esses dois símbolos são considerados comentários. Os comentários de bloco podem ocupar várias linhas.

Veja o exemplo da Figura 10:







**EXPRESSÕES ARITMÉTICAS**

Como estudamos no capítulo 2, os operadores aritméticos são símbolos que representam operações aritméticas, ou seja, as operações matemáticas básicas. A maior parte dos operadores aritméticos de C são os mesmos que vimos em Portugol. Conforme podemos ver na tabela abaixo, apenas acrescentamos o incremento unário (++) e o decremento unário (--):

+ : Soma

-: Subtração

\*: Multiplicação

/: Divisão

- -: Decremento Unário

++: Incremento Unário

%: Resto da Divisão Inteira

O operador de incremento unário (++) incrementa de 1 o seu operando. Ou seja, se eu quiser incrementar em 1 o valor de uma variável x, posso fazer x=x+1; ou fazer simplesmente x++;.

De forma análoga, o operador de decremento unário (--) decrementa de 1 o seu operando. Ou seja, se eu quiser decrementar de 1 o valor de uma variável x, posso fazer x=x-1; ou fazer simplesmente x--;.

A ordem de precedência entre os operadores em expressões aritméticas é a mesma já estudada, ou seja, primeiro as multiplicações e divisões e só depois as somas e subtrações. Em C também podemos utilizar os parênteses em expressões aritméticas.

**Exercicios.**

Faça um programa que peça ao usuário para digitar um número inteiro, armazene esse número numa variável chamada num1, peça ao usuário para digitar outro número inteiro, armazene esse número numa variável chamada num2, some os valores e guarde o resultado numa variável chamada soma. Exiba o resultado.

Faça um programa que peça ao usuário para digitar um número inteiro, armazene esse úmero numa variável chamada A, peça ao usuário para digitar outro número inteiro, armazene esse número numa variável chamada C. Exiba os valores digitados.

**ESTRUTURAS DE DECISÃO EM LINGUAGEM C**

**EXPRESSÕES LÓGICAS**

As expressões lógicas são expressões formadas a partir do uso de variáveis e constantes, operadores relacionais e operadores lógicos. As expressões lógicas são avaliadas e retornam sempre um valor lógico, em outras palavras, retornam sempre verdadeiro ou falso.

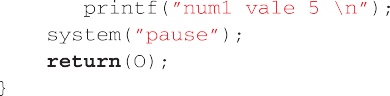
Abaixo temos uma tabela que exibe a representação dos operadores lógicos em C:

|  |  |
| --- | --- |
| OPERADOR LÓGICO | REPRESENTAÇÃO EM C |
| E &&  OU || (duas barras verticais) NÃO ! (exclamação) | |

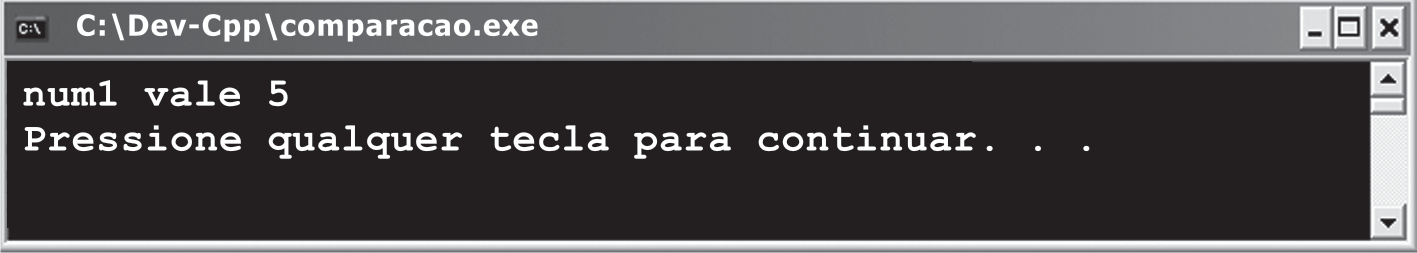
|  |  |
| --- | --- |
| DESCRIÇÃO | SÍMBOLO |
| igual a == (dois sinais de igual) maior que >  menor que <  maior ou igual a >=  menor ou igual a <=  diferente de != | |

Em C, O operador igual é representado por dois sinais de =, ou seja, por ==. Isso acontece para diferenciar o operador relacional do comando de atribuição.





Exibe o resultado da execução desse programa.



**ESTRUTURAS DE SELEÇÃO**

Ao desenvolver programas deparamos com situações nas quais o fluxo de execução do programa depende de determinadas condições, ou seja, parte do nosso programa só é executada se a condição para essa execução for verdadeira. Para isso existem os comandos de seleção ou decisão.

Para realizar essa tomada de decisão na linguagem C temos os comandos de seleção if e switch.

**COMANDO IF**

O comando if deve ser utilizado quando a execução de uma ou mais instruções do programa depender de uma ou mais condições simples.

**Sintaxe: if *(expressão de teste*) *instrução*;**

**Como Funciona?**

Se a *expressão de teste* que se encontra entre os parênteses for verdadeira, a *instrução* da linha subsequente será executada; caso contrário, não será.

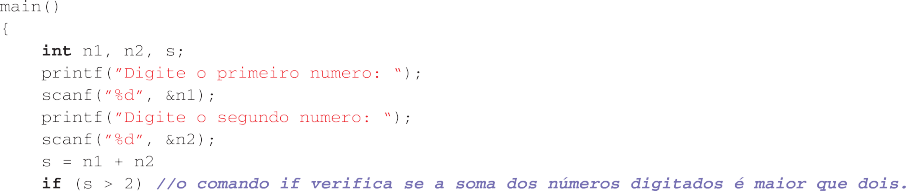
No caso de termos mais de uma instrução que dependa do resultado da condição para ser executada, essas instruções devem ficar entre chaves, conforme exibido na sintaxe abaixo:

**if *(expressão de teste*)**

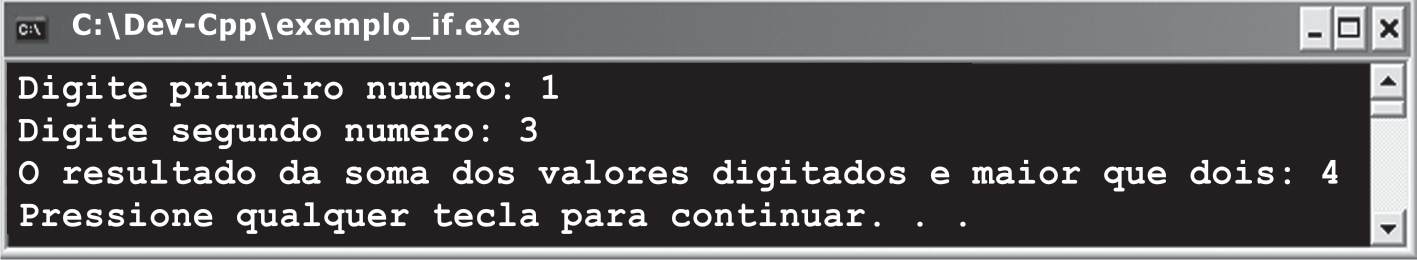
**{**

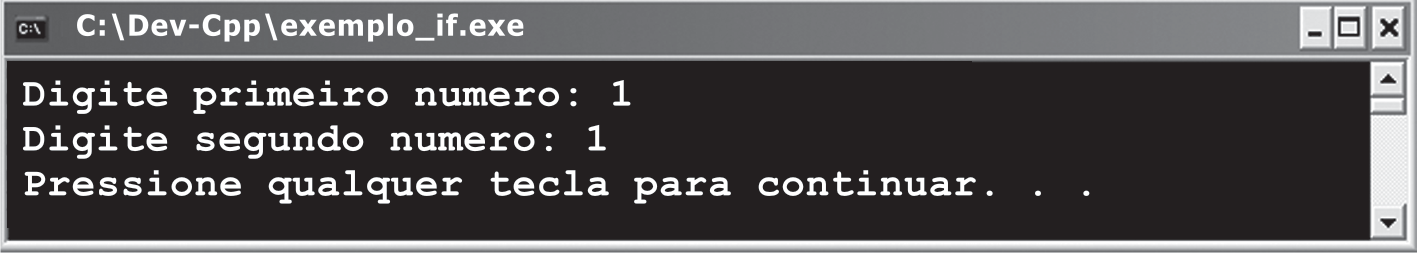
***instrução 1*;**

**}**



O resultado desse programa em um caso em que a soma dos números é maior que 2 e a Figura exibe o resultado da execução quando a soma não é maior que 2.



****

**Exercicios:**

Desenvolva um programa que leia a matrícula e a nota final de cada aluno de uma escola. Se a nota final for maior ou igual a 60, o programa deve exi- bir a mensagem “Aluno aprovado”.

Faça um programa que leia o sexo do usuário e apresente a mensagem “O sexo é válido”, se o ca- ractere digitado for ‘M’ ou ‘F’.

Faça um programa que leia um número dado como entrada e apresente a mensagem “O número está na faixa de 20 a 90” se o valor fornecido estiver entre 20 e 90.

Faça um programa que leia o valor do salário bruto de um funcionário. Se o salário for menor ou igual a R$ 500,00, o programa deve aplicar um aumento de 0.10 (10%).

**COMANDO IF-ELSE**

Como vimos, o comando if deve ser utilizado em situações nas quais um bloco de instruções só deve ser executado se uma determinada situação for verdadeira. Mas, muitas vezes deparamos com situações nas quais o programa deve seguir um fluxo caso uma determinada condição seja verdadeira e um outro fluxo caso essa condição seja falsa. Nessas situações, devemos utilizar o comando if – else.

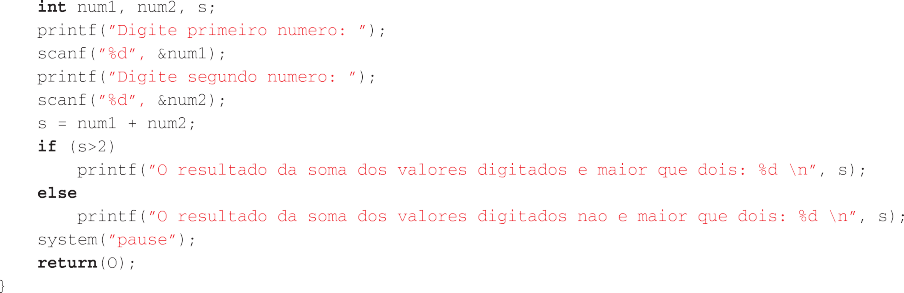
**Como Funciona?**

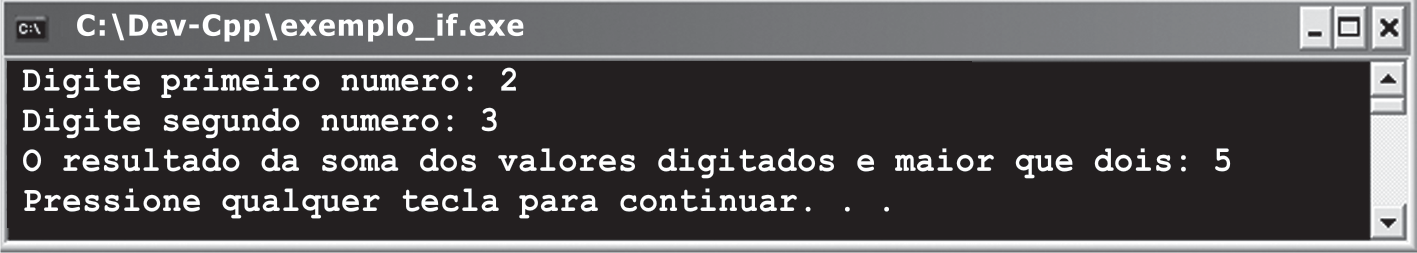
**Se a expressão de teste que se encontra entre os parênteses for verdadeira, o bloco de instruções logo abaixo do if será executado. Caso contrário, o bloco de instruções do else é que será executado.**

**Caso tenhamos apenas uma instrução no bloco do if ou no bloco do else, as chaves poderão ser omitidas.**

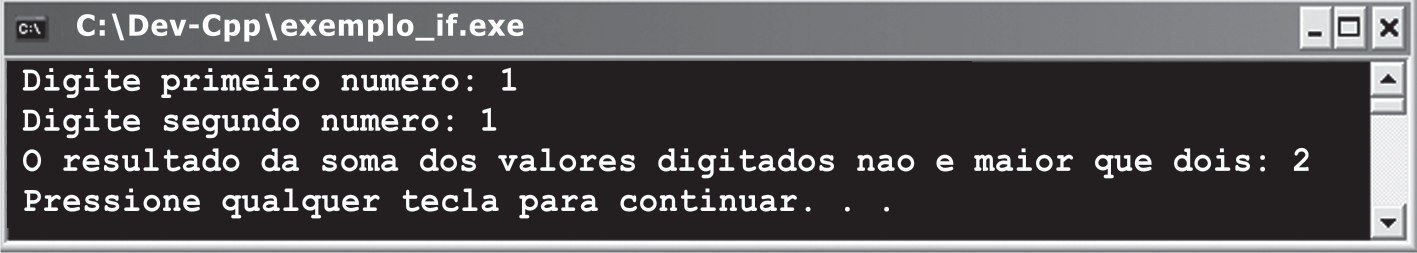
Anteriormente foi apresentado um programa que exibe uma mensagem caso a soma de dois números seja maior que 2. Caso a soma não atenda a essa condição, nenhuma ação é exercida pelo programa. Agora, vamos alterar aquele exemplo, utilizando o if – else. Em nosso novo exemplo, vamos efetuar a soma e, caso a soma seja maior que 2, será exibida uma mensagem informando isso. Caso contrário, será exibida uma mensagem informando que a soma não é maior que 2. Ou seja, vamos apenas acrescentar uma cláusula else ao nosso exemplo anterior.

Veja o exemplo anterior abaixo.



Exibe o resultado desse programa em um caso onde a soma dos números é maior que 2.

Execução do Programa para uma soma maior que 2.

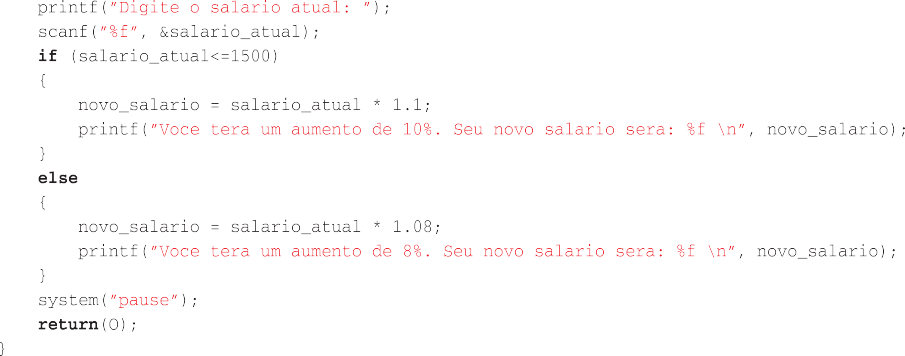


Execução do Programa para uma soma menor ou igual a 2.

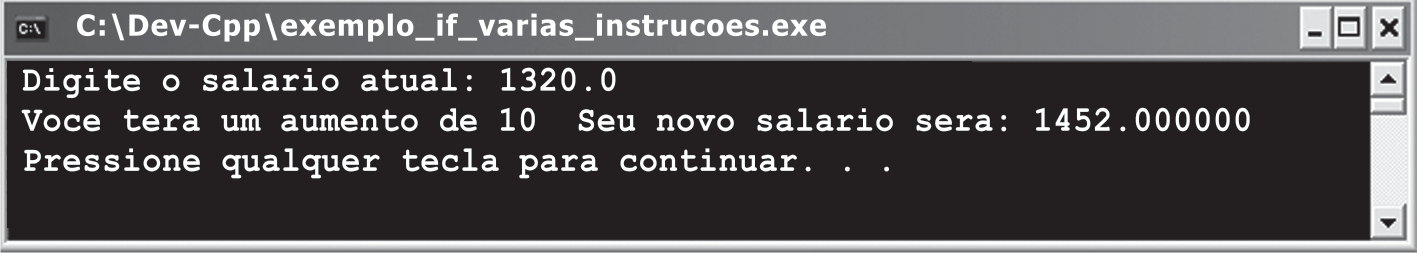
Vale ressaltar que, no exemplo anterior, apenas uma instrução é executada tanto no bloco do if quanto no bloco do else. Por isso, não foi necessário o uso das chaves { }. Caso tivéssemos mais de uma instrução em algum desses blocos, o uso das chaves seria obrigatório!

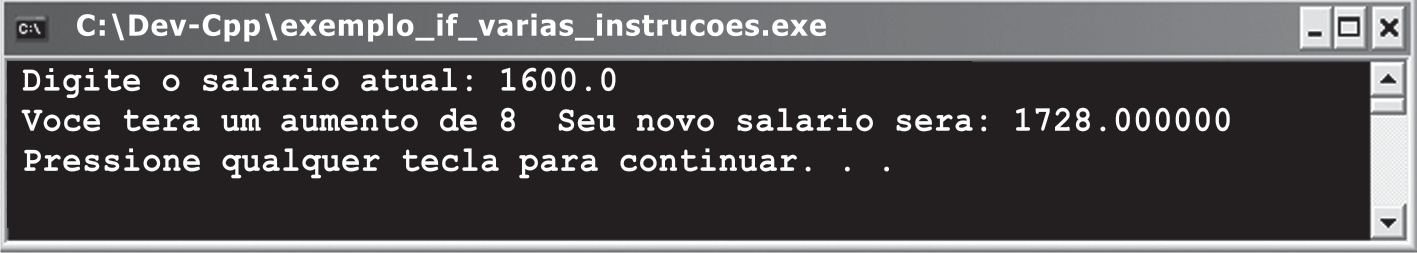
Abaixo, é apresentado um exemplo no qual o uso das chaves foi necessário. Nesse exemplo, uma empresa dará um aumento para os funcionários de acordo com o salário atual de cada um. Caso o funcionário receba até R$ 1.500,00, ele terá um aumento de 10%. Caso o salário seja maior que R$ 1.500,00, o aumento será de 8%. Assim, o programa solicita a digitação do salário e, de acordo com o valor atual, calcula o novo valor e exibe uma mensagem.





Exibe o resultado desse programa em um caso onde o salário é até R$1.500,00 e a Figura 22 exibe o resultado da execução quando o salário atual é maior que R$1.500,00.

Resultado da execução para um salário de até R$1.500,00

Resultado da execução para um salário maior que R$1.500,00

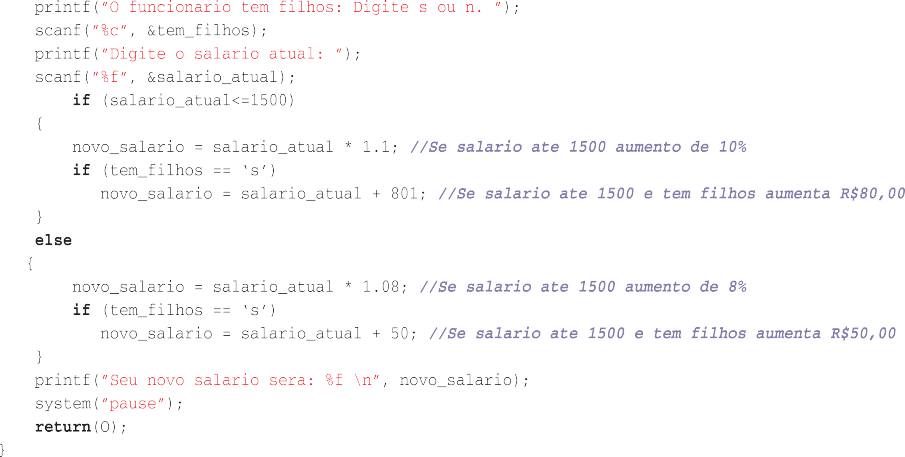
Exercicios:

Refaça os exemplos anteriores utilizando if – else.

Construa um programa que leia um número inteiro e imprima a informação se este número é ou não divi- sível por 5. Dica: Utilize o operador % (resto de di- visão inteira).

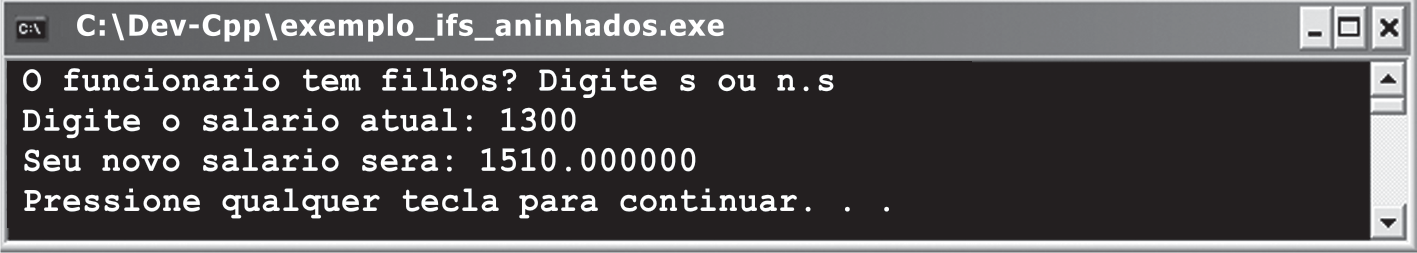
**COMANDOS IF-ELSE ANINHADOS.**

Podemos aninhar construções if-else, em outras palavras, podemos colocar comandos if-else ou comandos if dentro de outros comandos if-else.



Nesse exemplo, além do aumento percentual sobre os salários, os empregados também receberão uma ajuda caso tenham filhos. Assim, os empregados com salários até R$ 1.500,00 receberão o aumento de 10% e, se tiverem filhos, receberão mais R$ 80,00. Já os funcionários com salários maiores que R$ 1.500,00 receberão o aumento de 8% e, se tiverem filhos, receberão mais R$50,00. Assim, além de informar o salário, deverá ser informado também se o funcionário tem filho ou não, digitando ‘s’ para sim e ‘n’ para não. Note que foi acrescentada uma condição if dentro dos blocos if-else existentes para somar a gratificação no caso de ter filho. Note também que, nesse exemplo, utilizamos um printf único, fora das estruturas de condição que exibe o salário final.

Exibe o resultado da execução deste programa para um funcionário cujo salário é R$ 1.300,00 e que tem filhos.



Resultado da execução para um salário de R$ 1.300,00 e que tem filhos.

**Exercicios:**

Faça um programa que leia três valores distintos a serem digitados pelo usuário, determine e exiba o menor deles.

Sabendo que triângulo é uma figura geométrica de três lados em que cada um dos lados é menor que a soma dos outros dois, faça um algoritmo que rece- ba três valores e verifique se eles podem ser os com- primentos dos lados de um triângulo.

Faça um programa que leia o salário bruto e calcule o salário líquido. Para esse programa, o salário lí- quido será o salário bruto menos os descontos de INSS e IR, seguindo as regras:

* caso o salário seja menor que R$1.500,00, não devemos descontar IR e descontaremos 8% de INSS;
* para salários a partir R$1.500,00, descontaremos 5% de IR e 11% de INSS.