Universidade Federal do Piauí
Centro de Ensino Aberto e a Distância
Curso de Sistemas de Informação

Algoritmos e Programação I

Linguagem C

Prof. Arlino Magalhães
arlino@ufpi.edu.br

Agenda

1. Introdução a Linguagem C
2. Tipos Básicos
3. Variáveis
4. Constantes
5. Operadores
6. Comando de Saída
7. Comando de Entrada
8. Exemplos
9. Compilando o Código
10. Variáveis Booleanas
11. Comentários
12. Abreviação de Expressões
13. Cast (Modelador)
14. Exercícios

Introdução a Linguagem C

Foi criada por Dennis Ritchie (1941, 2011†) em 1972 no centro de pesquisas da *Bell Laboratories*.

Sua primeira utilização importante foi a reescrita do Sistema Operacional UNIX, que até então era escrito em *assembly*.

Se tornou tão popular que por volta de 1980, já existiam várias versões de compiladores C oferecidas por várias empresas.

É uma linguagem de propósito geral, mas é mais utilizada para escrever compiladores, analisadores léxicos, bancos de dados, editores de texto, etc.

Tipos Básicos

Tipos Básicos de C:

- **char:** o valor armazenado é um caractere. Caracteres geralmente são armazenados em códigos (usualmente o código ASCII).
- **int:** número inteiro.
- **float:** número em ponto flutuante de precisão simples. São conhecidos normalmente como números reais.
- **double:** número em ponto flutuante de precisão dupla.
- **void:** serve para indicar que um resultado não tem um tipo definido.

Variáveis

Regras básicas para nomear variáveis:

1. o nome de uma variável deve ser iniciado com uma letra e seguido por outras letras ou números;
2. nunca devem ser utilizados caracteres que não sejam alfanuméricos, com exceção do *underline* (_);
3. acentos não devem ser utilizados;
4. letras maiúsculas e minúsculas são consideradas caracteres diferentes.

Declaração de variáveis:

- `int i, idade, numero;`
- `float salario, altura;`

Atribuição:

- `idade = 31;`
- `sexo = 'A';`

Valores de caracteres devem usar aspas simples.

Obs.: não só as variáveis mas toda a linguagem C é *"**Case Sensitive**"*, isto é, maiúsculas e minúsculas fazem diferença. Por exemplo: **Idade ≠ idade**, ou seja, são duas variáveis diferentes.

Constantes

As constantes seguem as mesmas regras utilizadas nas variáveis, exceto pelo fato de seus valores poderem ser modificados durante a execução do programa.

Utilizamos o comando define na linguagem C para criar constantes.

Exemplos:

- `#define PI 3.14159265`
- `#define TAXA 0.99`

Operadores - Aritméticos

Operador	Símbolo
adição	+
subtração	-
multiplicação	*
divisão	/
divisão inteira	/
resto da divisão inteira	%

Operadores - Relacionais

Operador	Símbolo
igual	==
maior	>
menor	<
maior ou igual	>=
menor ou igual	<=
diferente	!=

Operadores - Lógicos

Operador	Símbolo
conjunção	&&
disjunção	||
negação	!

Comando de Saída

Sintaxe: `printf ("<informações_de_controle>", <lista_de_variáveis> );`

Informação de controle é a definição do tipo de dado do valor a ser exibido (geralmente de uma variável). Isto é feito usando-se os códigos de controle, que usam a notação **%**.

Código	Significado
%d	Inteiro
%f	Float
%c	Caractere
%s	String
%%	Caractere porcentagem (%)

Comando de Saída

Exemplos:

- `printf ("Digite o valor do número:");`
  → Digite o valor do número:
- `printf ("%f", 40.34);`
  → 40.34
- `printf ("Um caractere %c e um inteiro %d", 'D', 120);`
  → Um caractere D e um inteiro 120
- `printf ("%s eh um exemplo", "Este");`
  → Este eh um exemplo
- `printf ("%s%d%%", "Juros de", "10");`
  → Juros de 10%

A linguagem C utiliza o padrão numérico americano, ou seja, é utilizado o ponto (.) no lugar da vírgula (,).

Caracteres devem ser delimitados por aspas simples.

Strings (cadeias de caracteres) devem ser delimitadas por aspas duplas.

Comando de Entrada

Sintaxe: `scanf ("<informações_de_controle>", &<lista_de_variáveis> );`

Exemplos:

- `scanf ("%f", &salario);`
- `scanf ("%i", &idade);`
- `scanf ("%c", &letra);`
- `scanf ("%s", &nome);`
- `scanf ("%s%i%f", &nome", &idade, &salario);`

As informações de controle são utilizadas igual como no comando printf.

O caractere **&** indica que o valor digitado pelo usuário será armazenado no endereço de memória da variável.

Exemplo

A função **main** é a primeira a ser executada quando o programa for executado.

O **int** indica que a função main retornará um valor do tipo inteiro.

```
#include <stdio.h>

int main () {
    printf ("Alô mundo!\n");
    return 0;
}
```

A **#include** inclui a biblioteca **stdio.h**. A stdio possui funções de I/O.

Os caracteres **{** e **}** delimitam o início e fim da função main, respectivamente.

O **return** retorna um valor da função **main**.

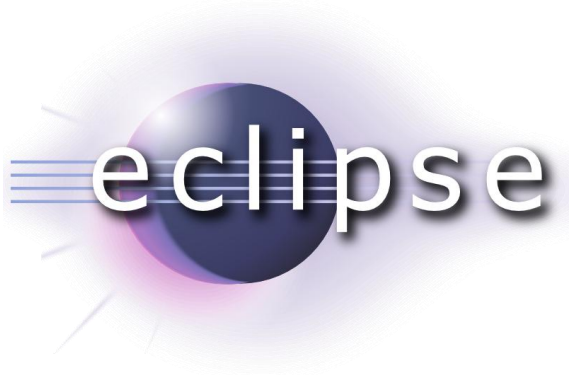
Exemplo - Velocidade Média

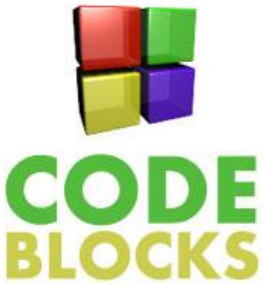

```c
#include <stdio.h>

int main(){
    float velocidade, tempo, distancia;
    printf ("Digite a distância:");
    scanf ("%f", &distancia);
    printf ("Digite o tempo:");
    scanf ("%f", &tempo);

    velocidade = distancia / tempo;
    printf ("Velocidade média = %f \n", velocidade);

    return 0;
}
```

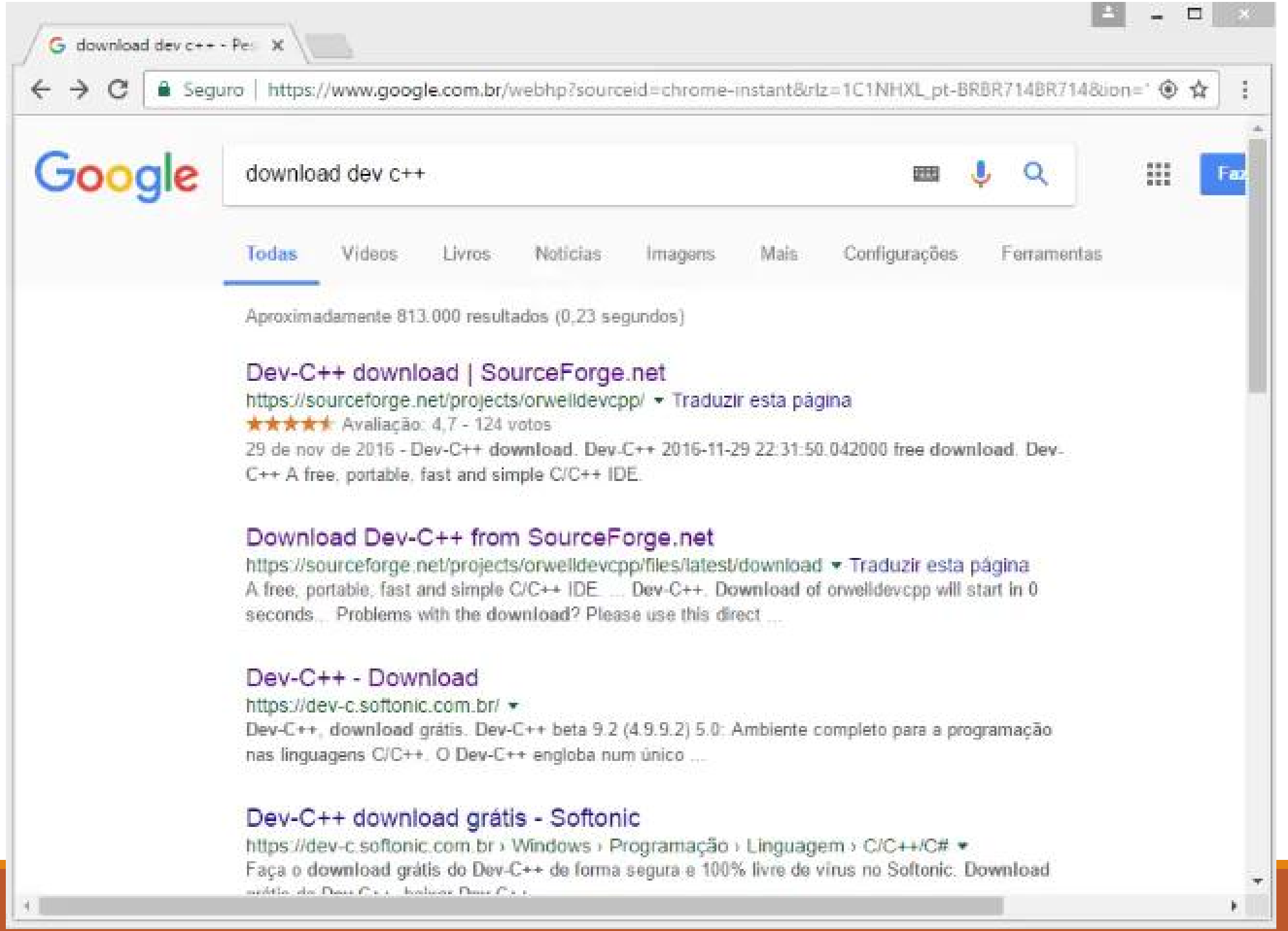
Exemplo - Operações sobre Circunferência

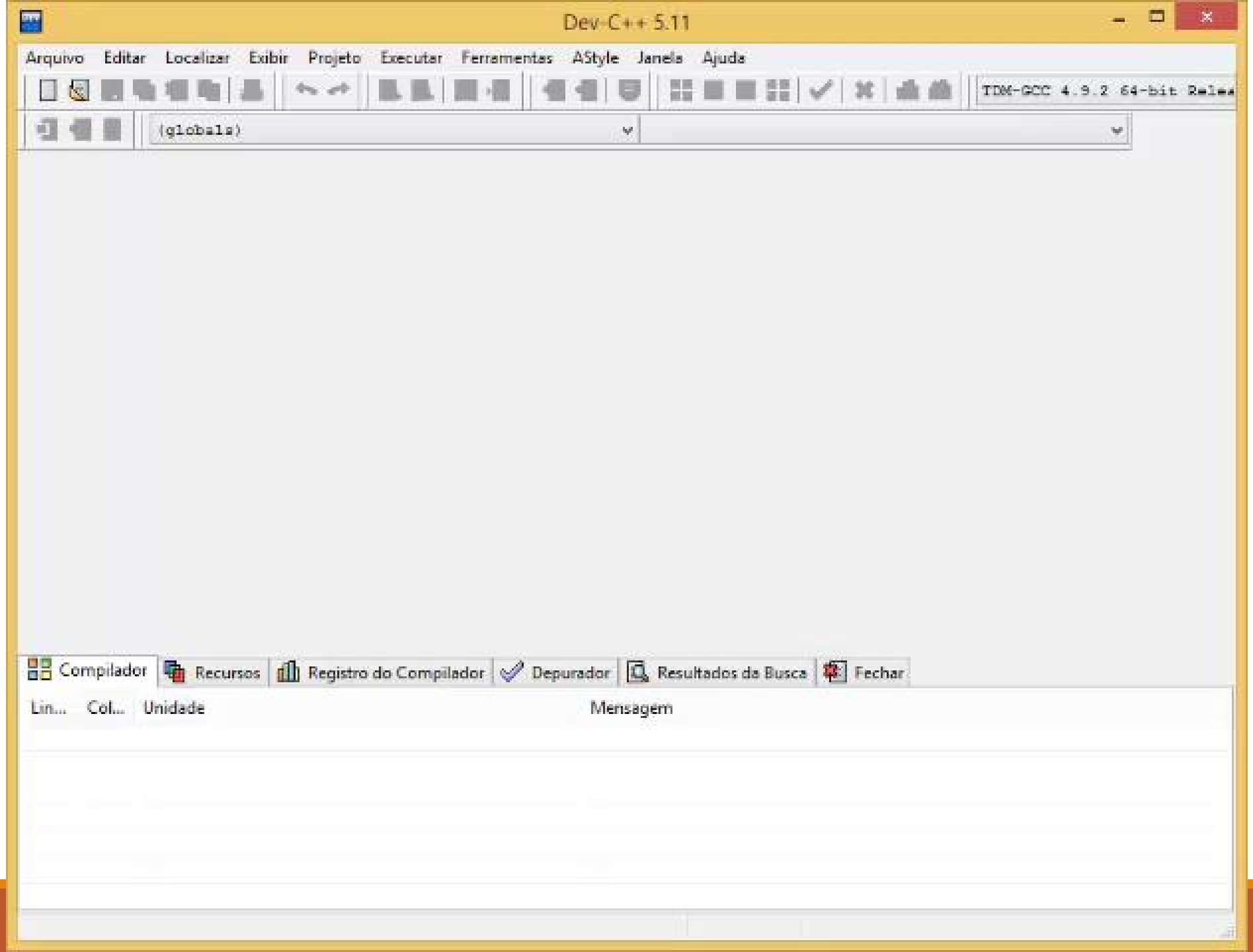

```c
#include <stdio.h>

int main(){
    #define PI 3.14
    float perimetro, area, raio;
    printf ("Digite o raio:");
    scanf ("%f", &raio);
    perimetro = 2*PI*raio;
    area = PI*raio*raio;
    printf ("Perímetro = %f\n", perimetro);
    printf ("Área = %f\n", area);

    return 0;
}
```

IDEs e Compiladores para C






download dev c++ - Pe
Seguro | https://www.google.com.br/webhp?sourceid=chrome-instant&rlz=1C1NHXL_pt-BRBR714BR714&ion=
Google
download dev c++
Todas
Vídeos
Livros
Notícias
Imagens
Mais
Configurações
Ferramentas
Aproximadamente 813.000 resultados (0,23 segundos)
Dev-C++ download | SourceForge.net
https://sourceforge.net/projects/orwelldevcpp/ Traduzir esta página
Avaliação: 4,7 - 124 votos
29 de nov de 2016 - Dev-C++ download. Dev-C++ 2016-11-29 22:31:50.042000 free download. Dev-C++ A free, portable, fast and simple C/C++ IDE.
Download Dev-C++ from SourceForge.net
https://sourceforge.net/projects/orwelldevcpp/files/latest/download Traduzir esta página
A free, portable, fast and simple C/C++ IDE. ... Dev-C++. Download of orwelldevcpp will start in 0 seconds. ... Problems with the download? Please use this direct ...
Dev-C++ - Download
https://dev-c.softonic.com.br/
Dev-C++, download grátis. Dev-C++ beta 9.2 (4.9.9.2) 5.0: Ambiente completo para a programação nas linguagens C/C++. O Dev-C++ engloba num único ...
Dev-C++ download grátis - Softonic
https://dev-c.softonic.com.br › Windows › Programação › Linguagem › C/C++/C#
Faça o download grátis do Dev-C++ de forma segura e 100% livre de vírus no Softonic. Download


Dev-C++ 5.11
Arquivo Editar Localizar Exibir Projeto Executar Ferramentas AStyle Janela Ajuda
TDM-GCC 4.9.2 64-bit Relea
(globals)
Compilador
Recursos
Registro do Compilador
Depurador
Resultados da Busca
Fechar
Lin... Col... Unidade
Mensagem

Variáveis Booleanas

A linguagem C não possui explicitamente variáveis do tipo booleano. Entretanto, a linguagem considera um número com valor 0 (zero) igual a falso e qualquer número com valor diferente de 0 (zero) igual a verdadeiro.

```
#include <stdio.h>

int main() {
    int expressao = 1;
     if (expressao)
            printf ("Essa expressão é verdadeira! :)\n");
     else
            printf ("Essa expressão é falsa! :(\n");

    return 0;
}
```

Comentários

Os comentário servem para explicar detalhes que julgarmos necessários em nosso algoritmo. Desta forma, aumentamos a legibilidade do nosso algoritmo. Os comentário não são considerados pelo compilador no processo de compilação.

Tipos de comentários:

- Os caracteres **/*** e ***/** permitem comentários de várias linhas e marcam o início e fim de um comentário, respectivamente.
- Os caracteres **//** (duas barras) permitem comentários em uma única linha.

```
/* Arlino Magalhães
   arlino@ufpi.edu.br
*/
float  saldo,         //variável saldo atual
       novo_saldo; //variável novo saldo
```

Abreviação de Expressões

A linguagem C admite as seguintes equivalências, que podem ser usadas para simplificar expressões ou para facilitar o entendimento de um programa:

Expressão Original	Expressão Equivalente
x = x + k;	x += k;
x = x - k;	x -= k;
x = x * k;	x *= k;
x = x / k;	x /= k;
x = x + 1	x ++
	++ x
x = x - 1	x --
	-- x

** k é uma constante*

Cast (Modelador)

Um *cast* é aplicado a uma expressão forçando-a a retorna um tipo especificado.

Sintaxe: (<*tipo de variável*>) expressão


```c
#include <stdio.h>

int main(){
    float f1;
    f1 = 10/7;
    printf ("%f \n", f1);
    f1 = (float) 10/7;
    printf ("%f \n", f1);

    return 0;
}
```

A divisão entre dois números inteiros sempre retorna um número inteiro.

Para que a divisão entre dois números inteiros retorne um número real é necessário a utilização do *cast*.

Exercício 01

Fazer um algoritmo que receba com entrada o saldo de uma aplicação e exiba como resultado o novo saldo após um reajuste de 10%.

Entrada:
- Saldo da aplicação: **saldo**.

Saída:
- Novo saldo: **novo_saldo**.

novo_saldo = saldo + saldo * 0.1

```c
#include <stdio.h>

int main(){
    float  saldo, novo_saldo;

    printf ("Digite a saldo da aplicação:");
    scanf ("%f", & saldo);
    novo_saldo = saldo + saldo * 0.1;
    printf ("Novo saldo = %f\n", novo_saldo);

    return 0;
}
```

Exercício 02

Faça um algoritmo em que seja dado um valor para x como entrada para a função $f(x) = 3x^2 + 2x + 7$. O programa deve exibir o valor de f(x).

Entrada:
- Valor de x: **x**.

Saída:
- Valor de f(x): **fx**.

fx = 3*x*x + 2*x + 7

```c
#include <stdio.h>

int main(){
    float  x, fx;

    printf ("Digite o valor de x:");
    scanf ("%f", & x);
    fx = 3*x*x + 2*x + 7;
    printf ("F(x) = %f\n", fx);

    return 0;
}
```

Exercício 03

Fazer um algoritmo que receba como entrada a quantidade de horas trabalhadas por um professor e o valor de sua hora-aula. O algoritmo deve retornar o valor do salário do professor.

Entrada:
- Quantidade de horas trabalhadas: **qtd_horas**
- Valor da hora-aula: **valor_hora**

Saída:
- Salário: **salario**.

salario = qtd_horas * valor_hora

```c
#include <stdio.h>

int main(){
    float  qtd_horas, valor_hora, salario;
    printf ("Digite a quantidade de horas:");
    scanf ("%f", &qtd_horas);
    printf ("Digite o valor da hora-aula:");
    scanf ("%f", &valor_hora);
    salario = qtd_horas * valor_hora;
    printf ("Salário bruto = %f\n", salario);

    return 0;
}
```