

LABORATÓRIO DE PROGRAMAÇÃO

AMBIENTES DESENVOLVIMENTO C

REFERÊNCIAS

- **LINGUAGEM C**

- ANSI WEBSTORE - ISO/IEC 9899-212 (Ultima versão oficial)
<https://webstore.ansi.org/Standards/INCITS/INCITSISOIEC98992012>
- ISO / IEC 9899 – 2011
<http://www.open-std.org/jtc1/sc22/wg14/www/standards>
<http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1570.pdf>
- SCHILDT, H., C COMPLETO E TOTAL 3ª Ed
<http://www.inf.ufpr.br/lesoliveira/download/c-completo-total.pdf>

INSTALAÇÃO AMBIENTES

- **DevC++**

- Tutorial: <http://linguagemc.com.br/tutorial-de-instalacao-do-dev-c/>
- Vídeo: <https://www.youtube.com/watch?v=00cTn4-xxrY> (acessado em 15/08/23 09:30)

- **Code Blocks**

- Tutorial: <http://linguagemc.com.br/tutorial-para-instalacao-do-code-blocks/>
- Vídeo: <https://www.youtube.com/watch?v=CiwPDUOvIMU>

COMPILADORES ONLINE

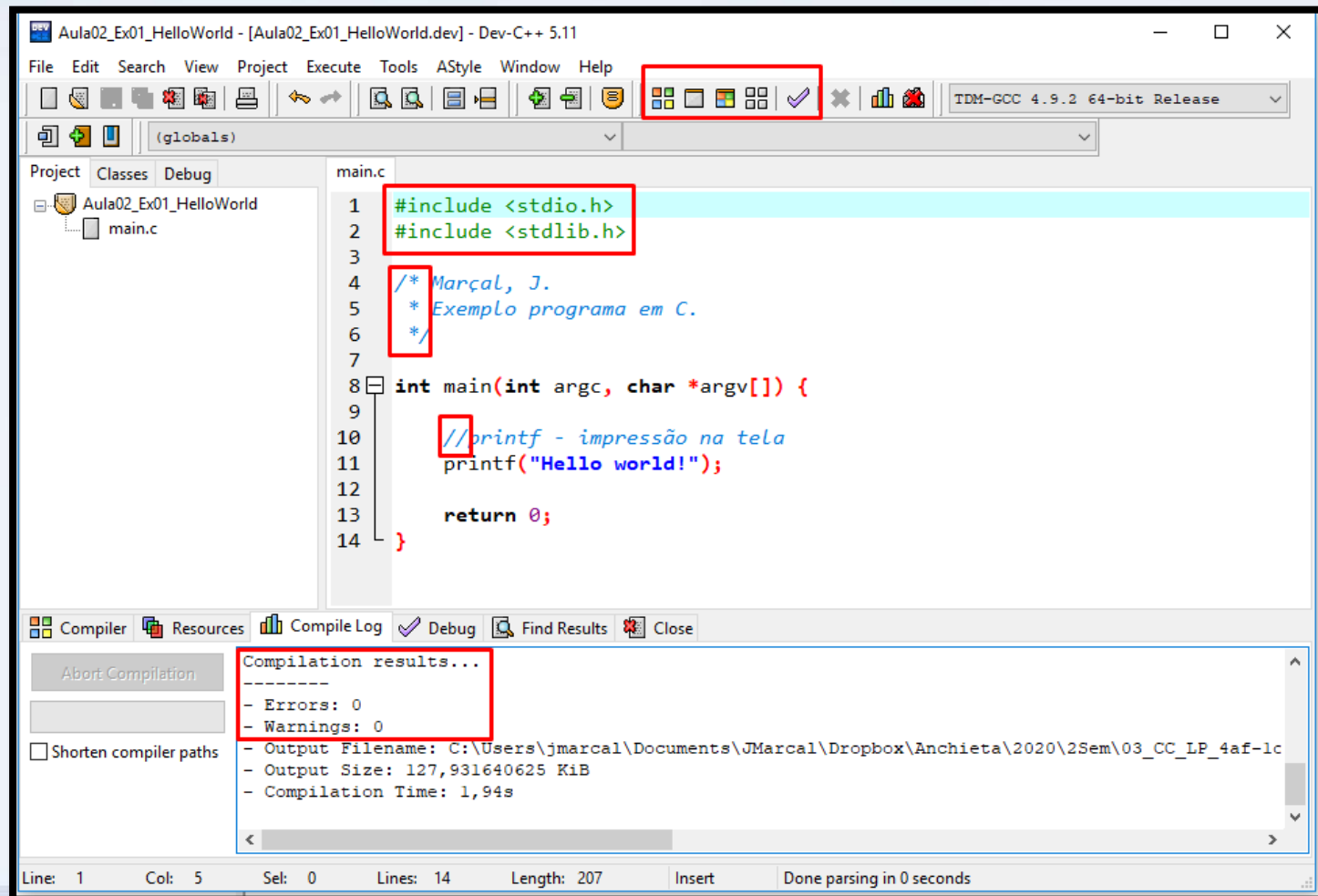
- **COMPILADORES ONLINE**

- ONLINEGDB: https://www.onlinegdb.com/online_c_compiler
- REPL: <https://repl.it/languages/c>
- CODECHEF: <https://www.codechef.com/ide>
- PROGRAMIZ: <https://www.programiz.com/c-programming/online-compiler/>
- PAIZA: <https://paiza.io/en/projects/new>

Nota: Existem vários compiladores online que podem ser utilizados durante as aulas, caso identifique algum e goste compartilhe com a turma. ;)

DEVCC++ IDE

- IDE (INTEGRATED DEVELOPMENT ENVIRONMENT)



LINGUAGEM C – ESTRUTURA BÁSICA

- Diretivas de pré-processamento

include – permite inclusão de outros arquivos no programa

stdio.h, stdlib.h – arquivos de cabeçalho, bibliotecas com instruções de entrada/saída

main – função de inicialização de programas em C

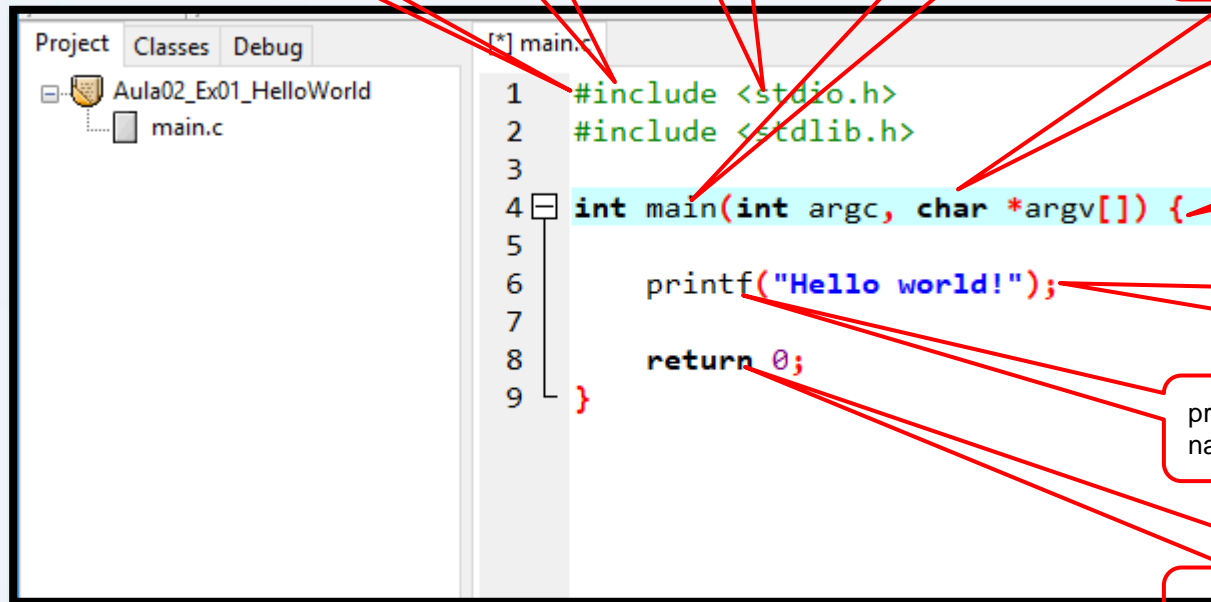
argc, *argv[] – parâmetros da função de inicialização

início de bloco de instrução

Indica término de instrução.

printf – função de impressão na tela.

return – instrução de retorno do método de inicialização.



The screenshot shows a code editor window with the following C code:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(int argc, char *argv[]) {
5
6     printf("Hello world!");
7
8     return 0;
9 }
```

Annotations with red lines pointing to specific parts of the code:

- Line 1: #include <stdio.h> (points to "# - Diretivas de pré-processamento")
- Line 2: #include <stdlib.h> (points to "include – permite inclusão de outros arquivos no programa")
- Line 4: int main(int argc, char *argv[]) { (points to "main – função de inicialização de programas em C" and "argc, *argv[] – parâmetros da função de inicialização")
- Line 6: printf("Hello world!"); (points to "printf – função de impressão na tela.")
- Line 8: return 0; (points to "return – instrução de retorno do método de inicialização.")
- Line 4: { (points to "início de bloco de instrução")
- Line 6: ; (points to "Indica término de instrução.")

LINGUAGEM C – ARQUIVOS

.c – arquivo de código fonte da aplicação.

.win – arquivo de controle de compilação do DevC++

.dev – arquivo de projeto do DevC++

.layout – arquivo de layout do DevC++

.exe – arquivo binário gerado após processo de compilação em linguagem de máquina.

Nome	Data de modificaç...	Tipo	Tamanho
Aula02_Ex01_HelloWorld.dev	11/08/2020 20:21	Dev-C++ Project ...	1 KB
main.c	11/08/2020 20:21	C Source File	1 KB
Aula02_Ex01_HelloWorld.layout	11/08/2020 20:21	Arquivo LAYOUT	1 KB
Makefile.win	11/08/2020 20:21	Arquivo WIN	2 KB
main.o	11/08/2020 20:21	Arquivo O	1 KB
Aula02_Ex01_HelloWorld.exe	11/08/2020 20:21	Aplicativo	128 KB

FUNDAMENTOS

FUNDAMENTOS

32 palavras reservadas

- auto
- break
- case
- char
- const
- continue
- default
- do
- double
- else
- enum
- extern
- float
- for
- goto
- if
- int
- long
- register
- return
- short
- signed
- sizeof
- static
- struct
- switch
- typedef
- union
- unsigned
- void
- volatile
- while

Principais bibliotecas em C

- `stdlib.h`
- `stdio.h`
- `math.h`
- `string.h`
- `limits.h`
- `ctype.h`
- `time.h`
- `stdbool.h`

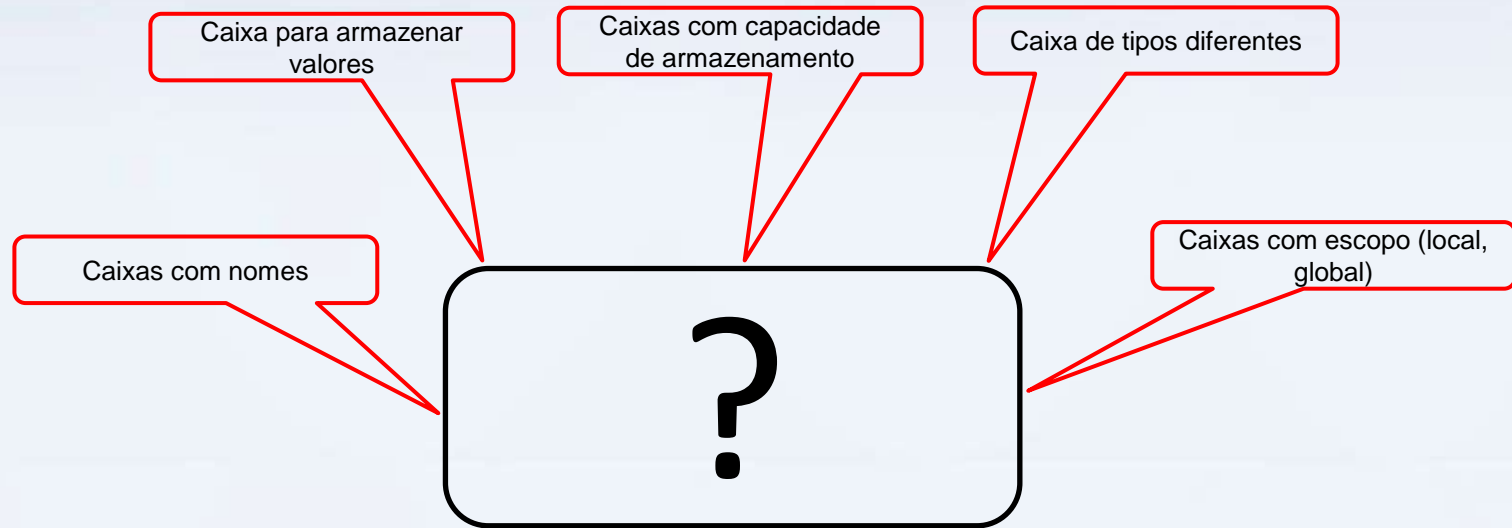
Case Sensitive

O C é "Case Sensitive"

Vamos começar o nosso curso ressaltando um ponto de suma importância: o C é "Case Sensitive", isto é, *maiúsculas e minúsculas fazem diferença*. Se se declarar uma variável com o nome soma ela será diferente de **Soma**, **SOMA**, **SoMa** ou **sOmA**. Da mesma maneira, os comandos do C **if** e **for**, por exemplo, só podem ser escritos em minúsculas pois senão o compilador não irá interpretá-los como sendo comandos, mas sim como variáveis.

FUNDAMENTOS

• VARIÁVEIS



O essa expressão faz?

$$m = \left[p1 + p2 \right] / 2$$

FUNDAMENTOS

- VARIÁVEIS

Tipos

Tamanho em bits ou bytes

Notação para entrada/saída

Limites (capacidade de armazenamento)

Tipo	Num de bits	Formato i/o	Início	Fim
char	8	%c	-128	127
unsigned char	8	%c	0	255
int	32	%d	-2.147.483.648	2.147.483.647
unsigned int	32	%u	0	4.294.967.295
long int	32	%li	-2.147.483.648	2.147.483.647
unsigned long int	32	%lu	0	4.294.967.295
short int	16	%hi	-32.768	32.767
unsigned short int	16	%hu	0	65.535
float	32	%f	$(+/-)10^{-38}$	$(+/-)10^{38}$
double	64	%lf	$(+/-)10^{-308}$	$(+/-)10^{308}$
long double	96			

FUNDAMENTOS

• VARIÁVEIS

Versão 1, MENOS amigável.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  /* Marçal, J.
5   * Exemplo programa em C, prática:
6   * variáveis, printf, scanf.
7   */
8
9  int main(int argc, char *argv[]) {
10     float m = 0.0;
11     float p1, p2 = 0.0;
12
13     scanf("%f", &p1);
14     scanf("%f", &p2);
15
16     m = (p1 + p2) / 2;
17
18     printf("%f", m);
19
20     return 0;
21 }
```

Versão 2, MAIS AMIGÁVEL no entanto não realiza nada de diferente do ponto de vista de algoritmo.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  /* Marçal, J.
5   * Exemplo programa em C, prática:
6   * variáveis, printf, scanf.
7   */
8
9  int main(int argc, char *argv[]) {
10     // Declaração das variáveis
11     float m = 0.0;
12     float p1, p2 = 0.0;
13
14     // Entrada
15     printf("...: CALCULADORA DE MEDIA :...\n");
16     printf("Digite o valor da primeira nota:\n");
17     scanf("%f", &p1);
18     printf("Digite o valor da segunda nota:\n");
19     scanf("%f", &p2);
20
21     // Processamento
22     m = (p1 + p2) / 2;
23
24     // Saída
25     printf("Sua media e: %f", m);
26
27     return 0;
28 }
```

FUNDAMENTOS

• SCANF (Leitura), PRINTF (Impressão)

scanf – Códigos de formatação utilizados pela função scanf.

Código	Função
%c	Lê um único caractere
%i ou %d	Lê um inteiro decimal
%e	Lê um número em notação científica
%f	Lê um número em ponto flutuante
%o	Lê um número octal
%s	Lê uma <u>string</u>
%x	Lê um número <u>haxadecimal</u>
%u	Lê um decimal sem sinal
%li ou %ld	Lê um inteiro longo
%lf	Lê um double
%Lf	Lê um long double

\a	soa o alarme do microcomputador (beep)
\b	o cursor retrocede uma coluna.
\f	Alimentação de formulário (FF).
\n	o cursor avança para uma nova linha.
\r	o cursor retrocede para o início da linha.
\t	o cursor avança para próxima marca de tabulação.
\"	exibe aspas duplas.
'	exibe aspas simples.
\\	exibe uma única barra invertida.

printf – Tabela de marcadores de printf.

Tipo	Descrição
d, i	Número inteiro decimal.
u	Número inteiro decimal sem sinal (unsigned int).
f, F	Número real (float ou double) em notação de ponto decimal fixo. Os indicadores de não-é-número e infinito são impressos como: nan e infinity para f; NAN e INFINITY para F.
e, E	Número real (float ou double) em notação científica: [-]d.ddd e[+ -].ddd. O expoente sempre contém dois dígitos e é precedido por e ou E.
g, G	Número real (float ou double) exibido com notação de ponto decimal fixo ou notação científica conforme sua magnitude.
x, X	Exibe um inteiro sem sinal (unsigned int) como valor hexadecimal em minúsculas (x) ou maiúsculas (X).
o	Exibe um inteiro sem sinal (unsigned int) como valor octal.
s	Imprime uma string.
c	Imprime um caractere (char).
p	Imprime um ponteiro para void, conforme implementação.
%	Exibe o símbolo % (mas não aceita flags, width, precision e length).

printf - Tabela de caracteres especiais (sequência de escape).

FUNDAMENTOS

- **SCANF, PRINT** (Consumo combustível)

Exemplo 1 – Simples

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(int argc, char *argv[]) {
5     // Declaracao variáveis
6     int distancia, numeroPedagios = 0;
7     float autonomiaVeiculo, precoCombustivel,
8         valorUnitarioPedagio = 0.0;
9     float consumoCombustivel, custoCombustivel, custoPedagios,
10        custoTotalViagem = 0.0;
11
12     // Entrada
13     printf("Digite a distância (kilometros):\n");
14     scanf("%i", &distancia);
15     printf("Digite a autonomia do veículo (km/l):\n");
16     scanf("%f", &autonomiaVeiculo);
17     printf("Digite o preço do combustível (l):\n");
18     scanf("%f", &precoCombustivel);
19     printf("Digite o número de pedágios:\n");
20     scanf("%i", &numeroPedagios);
21     printf("Digite o valor medido dos pedágios:\n");
22     scanf("%f", &valorUnitarioPedagio);
23
24     //Processamento
25     consumoCombustivel = distancia / autonomiaVeiculo;
26     custoCombustivel = consumoCombustivel * precoCombustivel;
27     custoPedagios = numeroPedagios * valorUnitarioPedagio;
28     custoTotalViagem = (custoCombustivel + custoPedagios);
29
30     //Saída
31     printf("Combustível: %f \n", custoCombustivel);
32     printf("Pedágios: %f \n", custoPedagios);
33     printf("Total: %f \n", custoTotalViagem);
34
35     return 0;
36 }
```

Exemplo 2 – Português, alinhamento horizontal (saída), formatação numérica (saída)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <locale.h>
4
5 int main(int argc, char *argv[]) {
6     setlocale(LC_ALL, "Portuguese");
7
8     // Declaracao variáveis
9     int distancia, numeroPedagios = 0;
10    float autonomiaVeiculo, precoCombustivel,
11        valorUnitarioPedagio = 0.0;
12    double consumoCombustivel, custoCombustivel, custoPedagios,
13        custoTotalViagem = 0.0;
14
15    // Entrada
16    printf("Digite a distância (kilometros):\n");
17    scanf("%i", &distancia);
18    printf("Digite a autonomia do veículo (km/l):\n");
19    scanf("%f", &autonomiaVeiculo);
20    printf("Digite o preço do combustível (l):\n");
21    scanf("%f", &precoCombustivel);
22    printf("Digite o número de pedágios:\n");
23    scanf("%i", &numeroPedagios);
24    printf("Digite o valor medido dos pedágios:\n");
25    scanf("%f", &valorUnitarioPedagio);
26
27    //Processamento
28    consumoCombustivel = distancia / autonomiaVeiculo;
29    custoCombustivel = consumoCombustivel * precoCombustivel;
30    custoPedagios = numeroPedagios * valorUnitarioPedagio;
31    custoTotalViagem = (custoCombustivel + custoPedagios);
32
33    //Saída
34    printf("\n");
35    printf("...:RELATÓRIO:... \n");
36    printf("Combustível:\t %3.2f \n", custoCombustivel);
37    printf("Pedágios: \t %3.2f \n", custoPedagios);
38    printf("Total: \t %3.2f \n", custoTotalViagem);
39
40    return 0;
41 }
```

DÚVIDAS/PERGUNTAS

