

## Linguagem de Programação C

---

- Estrutura de um programa
- Anatomia de um programa em C
  - Material: LP\_Aula01



## Histórico de C

Criada em 1972 nos Bells Telephone laboratories por Dennis Ritchie.

Finalidade:

- Permitir a escrita de um sistema operacional (Unix), usando uma linguagem de alto nível se comparado ao Assembly.

A linguagem resulta da evolução de uma outra linguagem, chamada de B, desenvolvida no mesmo laboratório por Ken Thompson.



# Algumas vantagens da linguagem

## Rapidez

- Consegue obter performance semelhante ao Assembly, usando instruções em alto nível.

## Simples

- Sintaxe simples, número diminuto de palavras reservadas, de tipos de dados e de operadores.

## Portável

- Padrão ANSI – código escrito em uma máquina pode ser compilado em outra máquina (com poucas ou sem alterações)

## Popular

- É a mais conhecida e utilizada no mundo.

## Modular

- Permite a programação modular, facilita a separação de projetos em módulos distintos e independentes, uso de funções.

## Alto Nível

- Linguagem de terceira geração, permite acesso a maior parte das funcionalidades de Assembly.

## Outros

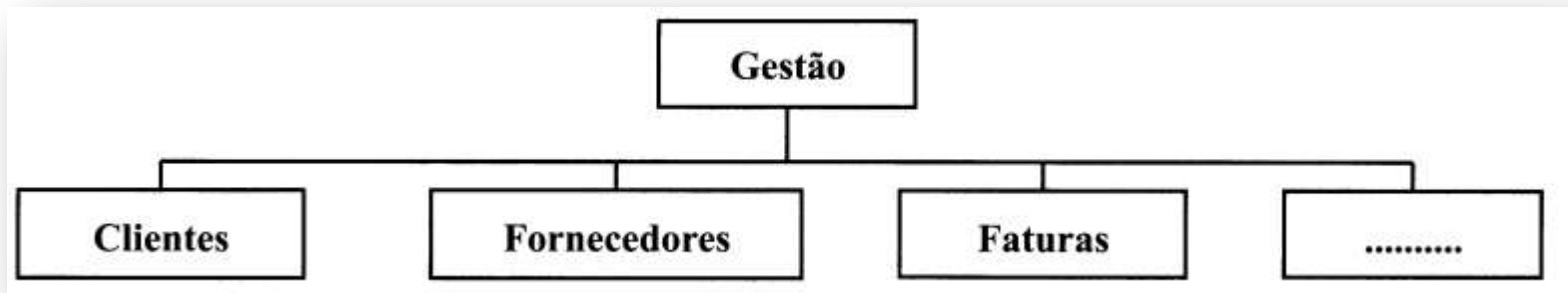
- Bibliotecas adicionais, evolução (POO) C++. Java se baseia em C/C++.



## Filosofia da programação em C

### Modularidade:

- Separar e implementar pequenos pedaços de códigos que realizem corretamente uma única função, e a realize bem. Exemplo:

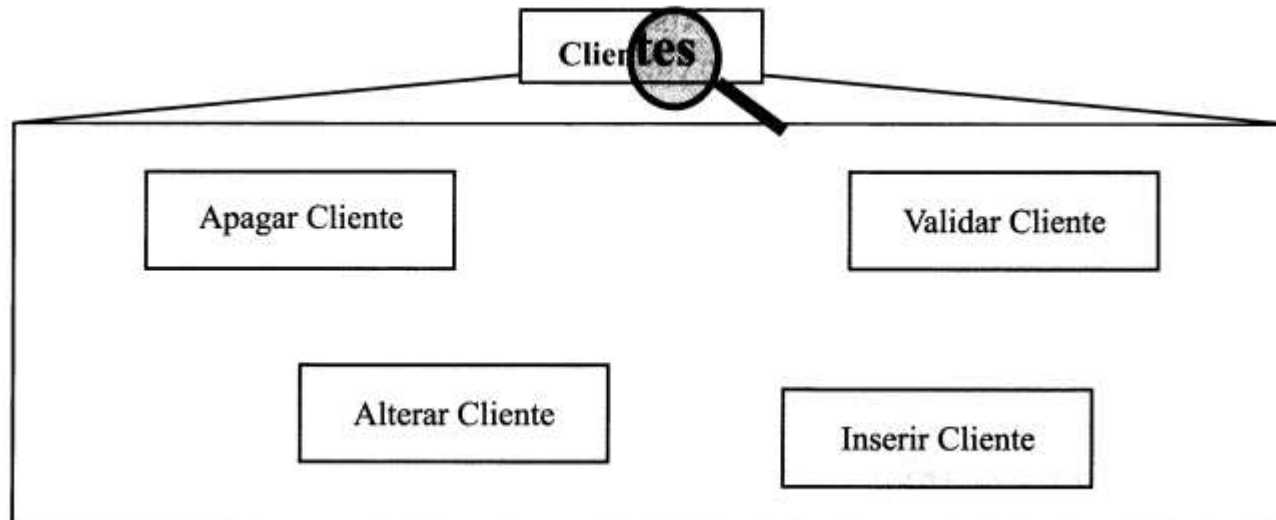


Cada módulo é implementado de maneira independente.

Cada módulo, por sua vez, é dividido nos diversos componentes que o compõe.



# Possíveis componentes do módulo Clientes.



## C versus C++

A linguagem C é um subconjunto da linguagem C++, isto é, C++ contém todas as características da linguagem C e mais um subconjunto de características próprias.

Nota:

- Para se dar um salto para C++, é imprescindível que o aluno tenha o domínio de C.





# Ciclos do desenvolvimento de uma aplicação



## Ciclo de 04 fases distintas:

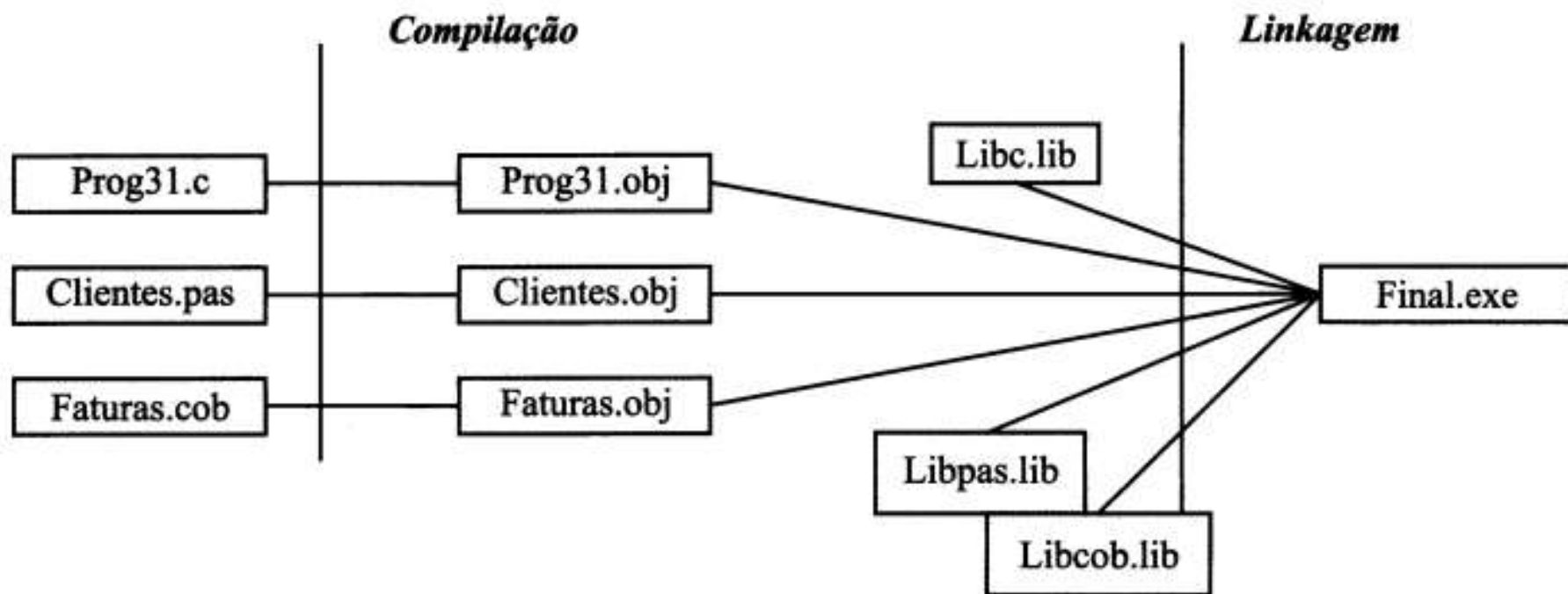
- 1) **Edição do código-fonte:** arquivos com extensão .c
- 2) **Compilação do programa:** verificação da sintaxe das instruções. Caso não haja erro é criado um arquivo código objeto.
- 3) **Linkagem dos objetos:** o arquivo executável é criado a partir do arquivo objeto (obtido da compilação) e através das “linkagens” das bibliotecas. Fase do linker.
- 4) **Execução do programa.**

Próxima figura ilustra o processo.





# Criação do executável: Final.exe



# Anatomia de um programa C



# Anatomia de um programa C

Um programa em linguagem C é formado por uma ou mais funções.

Cada função possui um nome exclusivo e corresponde à um bloco de código, delimitado por um par de chaves: { }

Contém um conjunto de declarações, expressões, comandos de controle e chamadas à outras funções.



## main()

A função denominada ***main*** é obrigatória em todos os programas, pois é o seu ponto de entrada, isto é, o programa começa a ser executado no início da função **main** e termina ao final desta função.

Normalmente a declaração desta função possui a seguinte forma: ***main(void)***

```
1: main()  
2: {  
3: }
```



## main()

Os parênteses sem mais nada após a função indicam que ela não recebe qualquer informação exterior.

Nota: C é Case Sensitive. Faz diferenciação entre maiúsculas e minúsculas. Todas as instruções de C são escritas em letras minúsculas. Usa-se letras maiúsculas quando se deseja utilizar variáveis, mensagens ou funções.



# Exemplo de um programa em c (olá mundo)





# Olá Mundo – Hello World

No exemplo temos:

Linha 1:

- Não é C. É uma diretiva que indica ao compilador que deverá adicionar ao processo um arquivo chamado “stdio.h” Biblioteca de entrada e saída.

Linha 2:

- Função principal – entrada principal do programa.

Linhas 3 e 5:

- Respectivamente início e fim do bloco de comandos.

Linha 4:

- Comando de saída – exibe a mensagem “Hello World” na console.

```
1: #include <stdio.h>
2: main()
3: {
4:     printf("Hello World");
5: }
```



# Anatomia do Exemplo Anterior

---

```
#include <stdio.h>
```

Inclui também as funções de Entrada e Saída

---

```
main()
```

O Programa começa aqui

---

```
{
```

Início do Bloco de Instruções

---

```
printf("Hello World");
```

Escrever a *string* “Hello World” usando a função printf

---

```
}
```

Fim do Bloco de Instruções (e fim de programa)

---

Fonte: DAMAS(2007, 21p).



## New Line

Em pascal usa-se os comandos Write e Writeln para a saída em tela, com a diferença que o segundo, além da saída, avança para a próxima linha.


Em C usa-se o mesmo comando printf. Porém para avançar para uma nova linha usa-se “\n” (New Line). Veja o exemplo:

```
1: #include <stdio.h>
2: main()
3: {
4:     printf("Hello World\n");
5: }
```



# Problema com delimitador de string

```
1: #include <stdio.h>
2: main()
3: {
4:     printf("Hoje está um "LINDO" dia!!!\n");
5: }
```



Para que não ocorra erro de compilação usa-se um caractere “\” barra invertida antes da aspas duplas.

```
1: #include <stdio.h>
2: main()
3: {
4:     printf("Hoje está um \"LINDO\" dia!!!\n");
5: }
```



## Caractere especial “\”

Usado para retirar o significado especial que um caractere apresenta.

No caso anterior, retirou o significado da delimitação de string dos caracteres (“”).

No caso de \n – representa um caractere que de outro modo seria quase impossível representar.

Veja a lista completa dos caracteres que podem ser representados pelo \.



# Caractere e sua representação quando precedido do \

Fonte: DAMAS(2007, 24p).

\7	<i>Bell</i> (sinal sonoro do computador)
\a	<i>Bell</i> (sinal sonoro do computador)
\b	<i>BackSpace</i>
\n	<i>New Line</i> (mudança de linha)
\r	<i>Carriage Return</i>
\t	Tabulação Horizontal
\v	Tabulação Vertical
\\	Caractere \ (forma de representar o próprio caractere especial \ )
\'	Caractere ' (aspas simples)
\"	Caractere " (aspas)
\?	Caractere ? (ponto de interrogação)
\ooo	Caractere cujo código <i>ASCII</i> em Octal é ooo
\x <sub>nn</sub>	Caractere cujo código <i>ASCII</i> em Hexadecimal é <sub>nn</sub>
%%	caractere %





## Comentários

Comentários não são interpretados pelo compilador, ou seja, são ignorados pelo compilador, e o programa executável não terá qualquer sinal deles.

Comentário em C, é qualquer conjunto de caracteres compreendido entre os sinais de /\* e \*/

### Exemplo 1

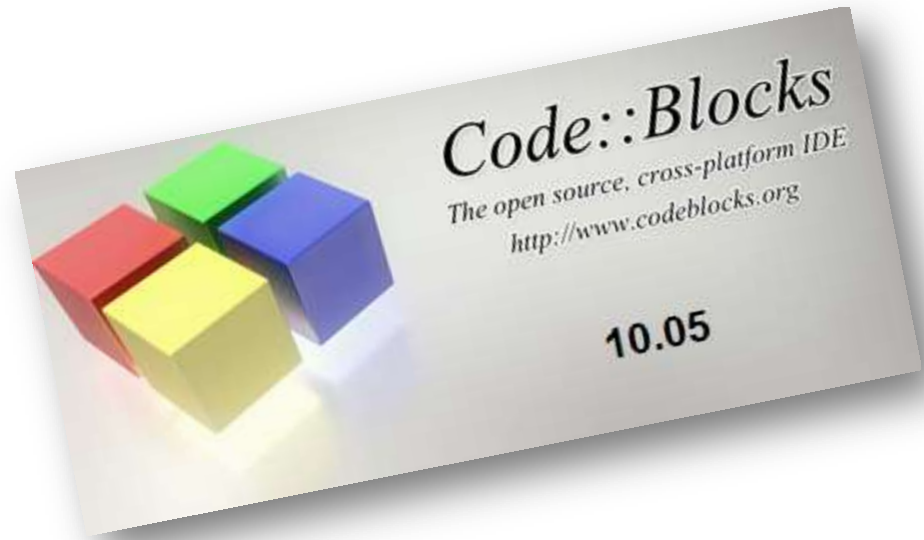
```
/******  
* PROG0199.C :   Comentários e Companhia      *  
* AUTOR:         Luís Damas                    *  
* DATA:         01/11/1995                    *  
*****/
```

### Exemplo 2

```
/* PROG0199.C :   Comentários e Companhia  
* AUTOR:         Luís Damas  
* DATA:         01/11/1995  
*/
```



# Code::Blocks



## Code::Blocks

**Code::Blocks** é uma IDE (*Integrated Development Environment* - Ambiente de Desenvolvimento Integrado), com destaque de sintaxe, criado para atender as necessidades dos usuários mais exigentes. Possui um framework de **plugins**, deste modo, o usuário pode melhorar a funcionalidade do mesmo.



## Onde baixar

Este ambiente pode ser obtido na  
URL <http://www.codeblocks.org/downloads>



## Criar um Projeto

Depois que o CODE::BLOCKS tiver sido carregado, abra o menu **File** e selecione a opção **New/Project**.

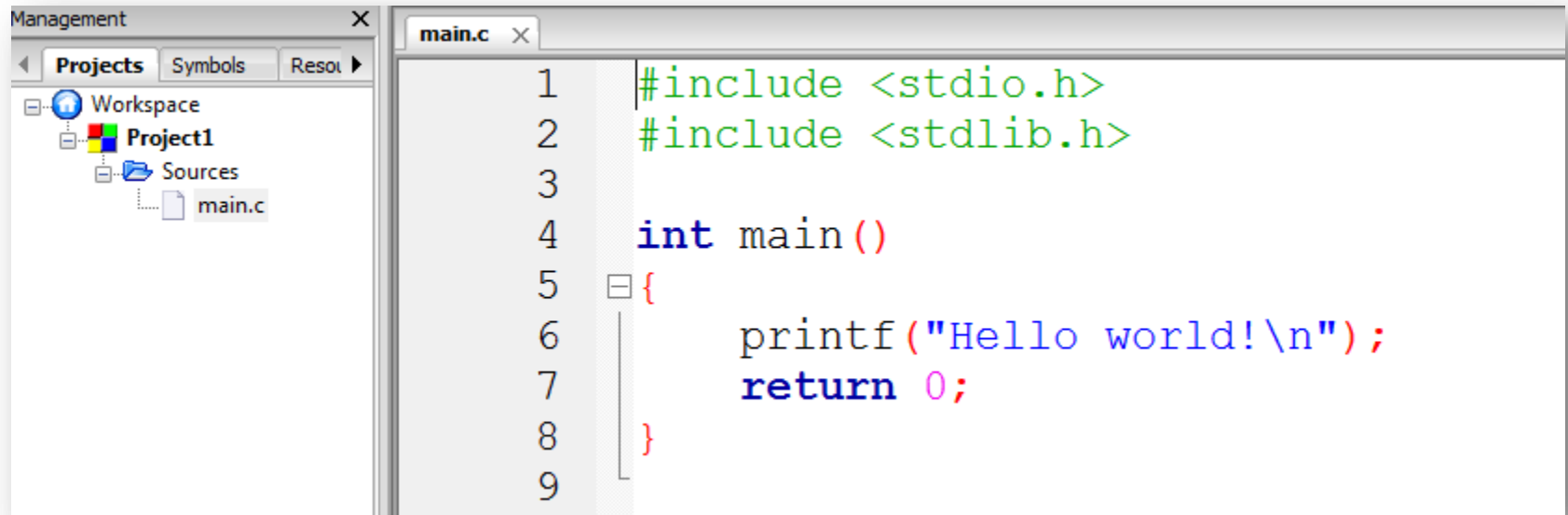
Na janela que surge, clique no ícone **Console Application**, e em seguida no botão **GO**.

Selecione a linguagem C, clique depois no botão **Next**.

Dê um nome ao projeto, selecione **Next** e **Finish**.



# Módulo Principal – main()



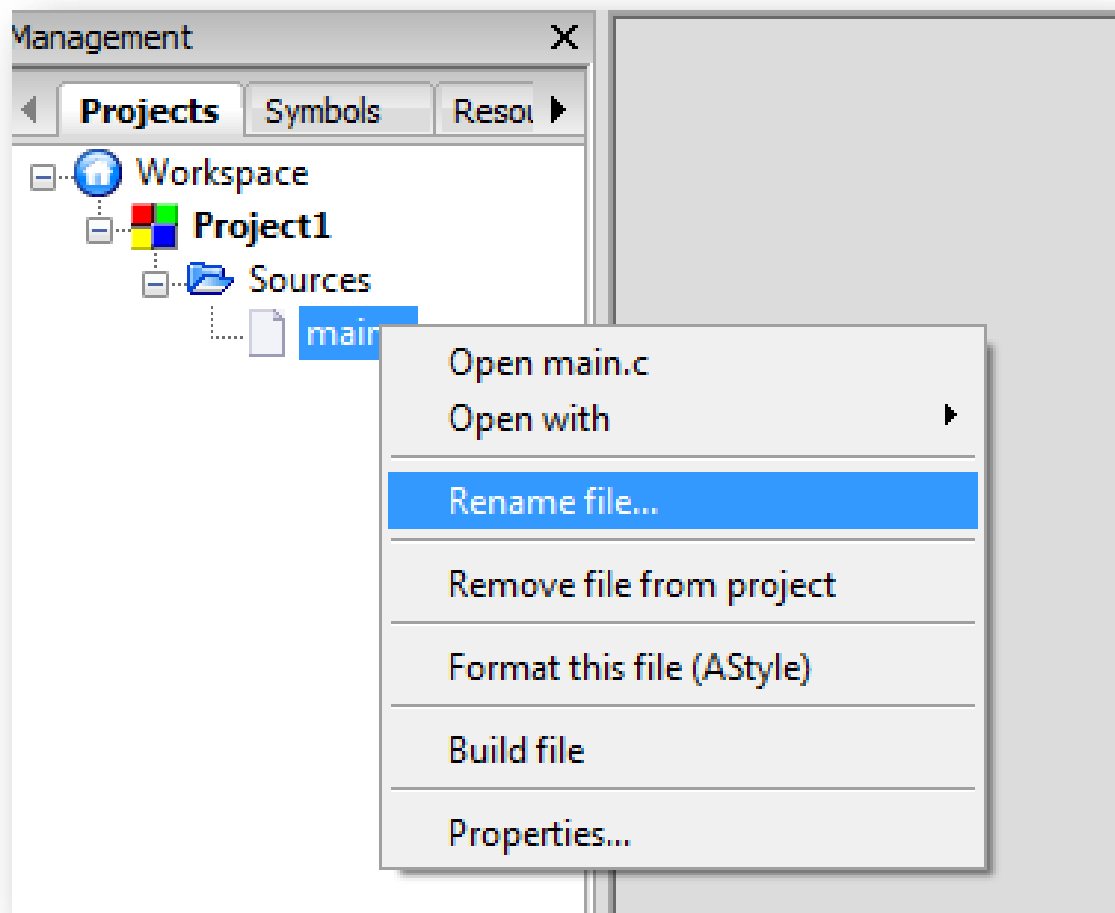
The image shows a screenshot of a C code editor. On the left, a 'Management' sidebar displays a project structure: 'Workspace' contains 'Project1', which contains 'Sources' and 'main.c'. The main editor window, titled 'main.c', shows the following code:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     printf("Hello world!\n");
7     return 0;
8 }
9
```





# Renomeando Arquivo



# Primeiro Programa em C

```
#include <stdio.h>
#include <stdlib.h>
```

```
int main()
{
    printf("Hello world!\n");
    return 0;
}
```

**Atente aos  
comentários e  
observações do  
professor.**

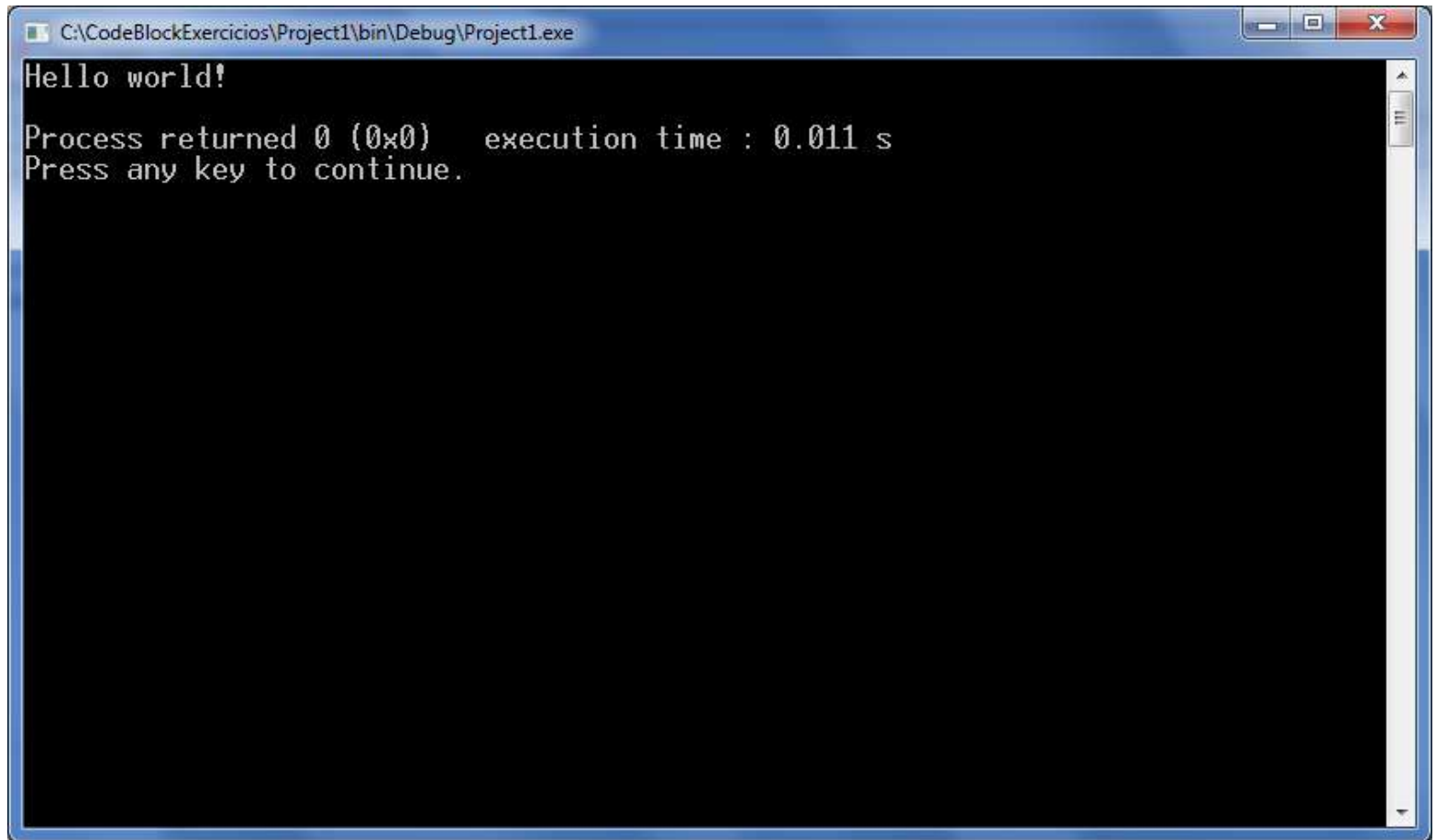


# Compilar e Executar um Programa em C

Para compilar e executar o programa, clique no botão **Build and run**.



# Resultado



A screenshot of a Windows command prompt window. The title bar shows the file path: C:\CodeBlockExercicios\Project1\bin\Debug\Project1.exe. The window has a black background with white text. The text displayed is: "Hello world!", "Process returned 0 (0x0) execution time : 0.011 s", and "Press any key to continue.".

```
C:\CodeBlockExercicios\Project1\bin\Debug\Project1.exe
Hello world!
Process returned 0 (0x0) execution time : 0.011 s
Press any key to continue.
```



## Exercícios – Aula 01

1) Escreva um programa em C que apresente a seguinte tela.

```
1 -      Clientes
2 -      Fornecedores
3 -      Faturas

0 -      Sair
```



## Exercícios – Aula 1

- 2) Escreva um programa em C que apresente duas linhas com a string “Aqui vai o apito”, ouvindo-se ao final de cada string um sinal sonoro.
- 3) Escreva um programa em C que indique qual o significado dos seguintes caracteres especiais: `\n`, `\\`, `\t`, `%%`





CENTRO PAULA SOUZA

---

**Fatec**

Mogi Mirim  
Arthur de Azevedo

Prof. Me. Marcos Roberto de Moraes, o Maromo

**FIM**



# Referências Bibliográficas

DAMAS, L. M. D. Linguagem C. LTC, 2007.

HERBERT, S. **C completo e total**. 3a. ed. Pearson, 1997.

