INSTITUTO FEDERAL
Sul-rio-grandense | Campus Gravataí

Introdução a Linguagem de Programação C

Padrão C ANSI

Prof. Hunder Everto Correa Junior

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

Sumário

- **Introdução**
- **Pequeno Histórico**
- **Caracterísitcas Gerais**
- **Compilação x Interpretação**

Um pequeno histórico da Linguagem C

- Ken Thompson inspira-se na linguagem BCPL para criar B.
- No início da década de 70, o UNIX também começava a ser desenvolvido:
  - Kernel escrito em assembly.
- Brian Kernigham e Dennis Ritchie inspiram-se em B para criar uma linguagem que permitisse desenvolver o kernel UNIX mais facilmente:
  - Mais fácil de trabalhar.
  - Igualmente poderosa.

Um pequeno histórico

- Nascia C, definida no livro *The C Programming Language.*
- Tornou-se padrão para desenvolvimento de aplicativos.
- Tornou-se padrão ANSI em 1983
- Hoje em dia, utilizada em aplicativos de alto desempenho ou que tenham contato com hardware (kernel, drivers).

Características Gerais

- Linguagem muito poderosa.
- Não tão rápida quanto assembly.
- Mas muito mais fácil de programar.
- Se as linguagens forem divididas em alto e baixo nível, C será de médio nível.

Características Gerais

Quarto Nível 4GL	Rational Rose
Alto Nível	Ada Modula-2 Visual Basic Delphi Java C++
Médio Nível	Pascal COBOL FORTRAN C C++ FORTH BASIC
Baixo Nível	Assembly/Macro Assembly

Compilação vs. Interpretação

- C é uma linguagem compilada.
  - Instruções do programador “traduzidas” para a linguagem de máquina apenas uma vez.
- Outras linguagens são interpretadas:
  - Basic, Java.
  - Instruções do programador são “traduzidas” a cada execução.

Compilação vs. Interpretação

- Vantagens da compilação:
  - Desempenho
  - Independência do interpretador:
- Para executar o aplicativo, só é necessário o executável deste.
- Desvantagens
  - Portabilidade
- Código compilado para um processador não é executável em outro.

Compilação vs. Interpretação

- Há espaço para as duas abordagens
- Algumas verdades:
  - Código interpretado sempre será mas lento que o compilado.
  - Com alguns cuidados durante a compilação, é possível apenas recompilar o programa para a nova plataforma, sem alterar o código.

Compilação vs. Interpretação

- O processo de compilação é iniciado após a codificação:
  - Sintaxe é conferida.
  - Varredura é feita para encontrar diretivas de compilação.
  - Código é traduzido para instruções de máquina, gerando o programa-objeto.
  - Programa objeto será unido com outros programas-objeto e bibliotecas para gerar o aplicativo. Esta etapa é chamada de linkedição ou linkagem (linking).

Compilação vs. Interpretação

- Para cada arquivo C, é gerado um programa-objeto.
  - Como os programas-objetos serão unidos depois, um mesmo aplicativo pode, e geralmente é, formado por vários arquivos-fonte.
- Existem arquivos que são “coleções” de funções. Estes arquivos são as bibliotecas.
  - Bibliotecas são como programas-objeto.

Bibliotecas

- São unidas com os programas-objeto para criar o aplicativo.
- As funções em uma biblioteca são descritas em um arquivo de cabeçalho (header). Estes arquivos possuem a extensão h.
- Incluir um arquivo header em um arquivo-fonte permite o uso das funções da biblioteca descrita por ele.

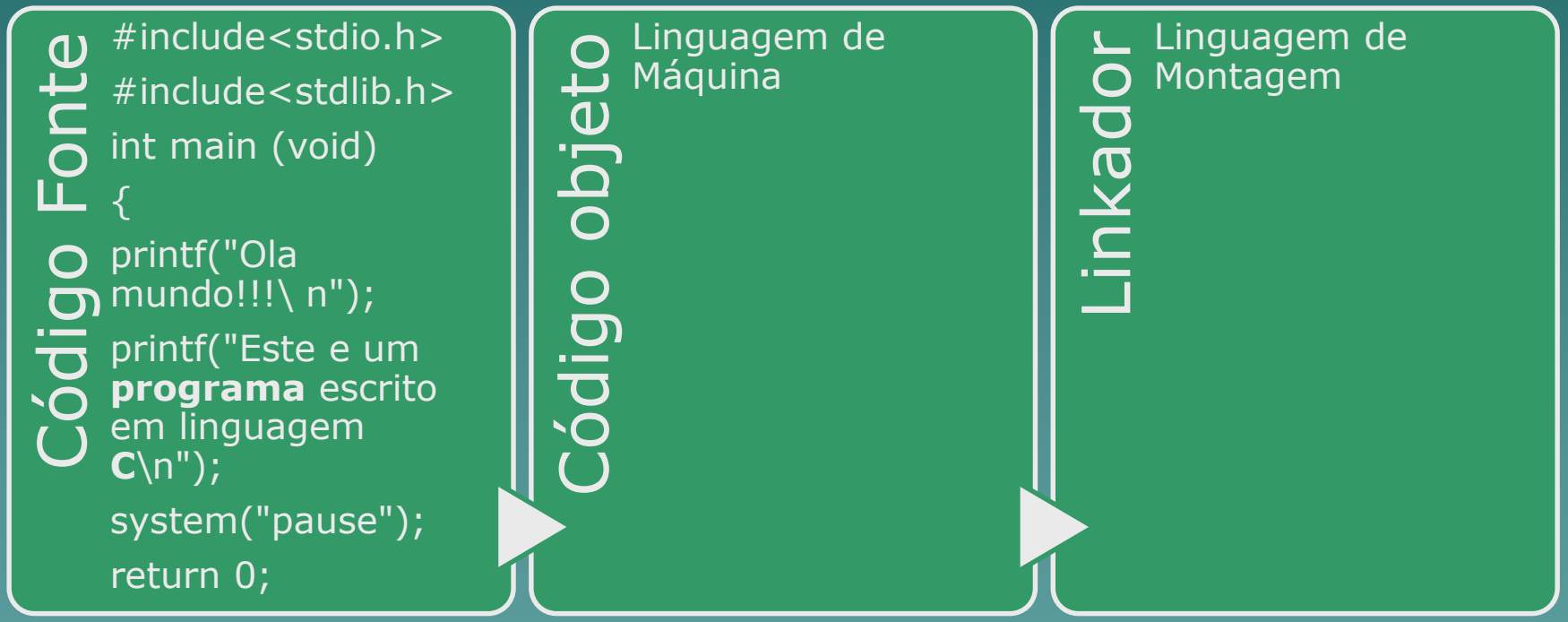
Linkagem

- A linkagem de uma biblioteca pode ser estática ou dinâmica:
  - Na linkagem estática, o código da biblioteca é copiado para o aplicativo. O aplicativo não precisa mais da biblioteca.
  - Na linkagem dinâmica, o código não é copiado, há apenas uma referência para a biblioteca.

Linkagem

- Portabilidade:
  - Codificando corretamente, é possível compilar aplicativo para outros sistemas.
  - Há a necessidade de existirem as bibliotecas para o sistema-alvo.

Resumindo


Código Fonte
#include<stdio.h>
#include<stdlib.h>
int main (void)
{
printf("Ola mundo!!!\ n");
printf("Este e um **programa** escrito em linguagem **C**\n");
system("pause");
return 0;
Código objeto
Linguagem de Máquina
Linkador
Linguagem de Montagem

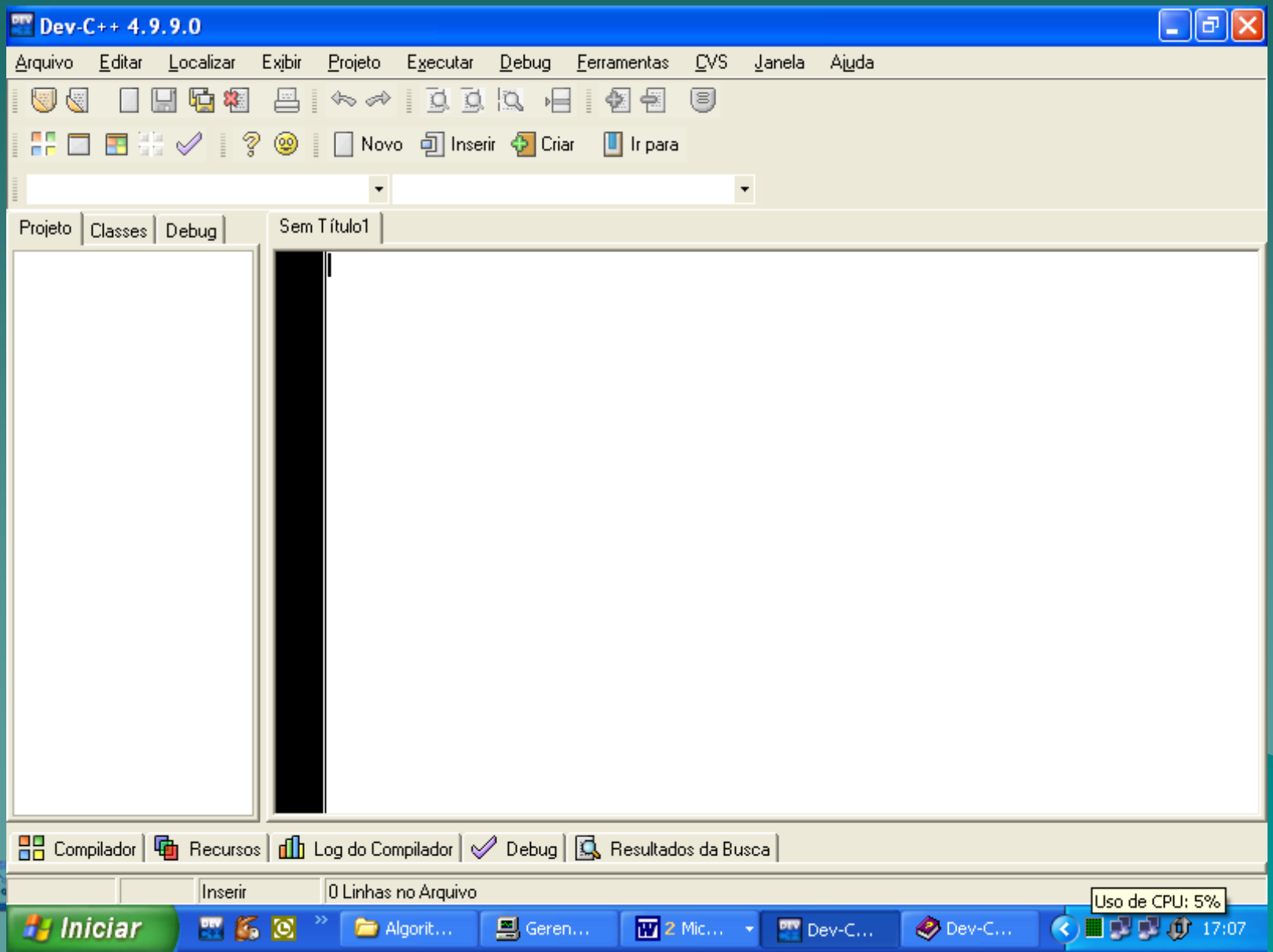
O ambiente Dev-C++

- O Dev-C++ é um ambiente de desenvolvimento de programas em C e C++ (editor, compilador, bibliotecas...)
- Pode ser baixado de http://www.bloodshed.net/devcpp.html

Usando o Dev-C++

- Inicie o Dev-C++ pelo ícone ou pelo menu
- Crie um novo arquivo, com o comando *Arquivo, Novo, Arquivo Fonte*
- Edite o programa da página seguinte:

Ambiente Dev-c++



Usando o Dev-C++

- Salve o programa com o nome **exemplo.cpp** em um diretorio com o seu nome
- Compile e execute o programa pressionando a tecla **F9**
- Se houver algum erro de sintaxe, aparece uma ou mais mensagens no rodapé da janela. Neste caso, corrija o programa e repita.

Dicas

- Termine todos os comandos com ;
- Quando ocorrer um erro de compilação, dê um duplo clique sobre a mensagem de erro para destacar o comando errado no programa
- Verifique também a linha anterior, que pode ser a responsável pelo erro, especialmente se faltar o ;
- Use comentários, iniciados por // ou entre /* */

```
/* isto é um comentário */
// isto também é um comentário
```

Estrutura do C :

```
# include <biblioteca.h >
Int main()
{
    comando 1;
    comando 2;
}
```

Bibliotecas

Conforme vimos no exemplo, a linguagem C precisa incluir as bibliotecas no código do programa.

A importância da biblioteca em C é imensa, pois ela nos poupa de muita programação. Uma vez que a função já está pronta dentro da biblioteca, basta importar tal biblioteca e utilizar a função que queremos.

Importando uma biblioteca em C

- Em C, a importação de bibliotecas são mais simples, bastando acrescentar para cada biblioteca um include em uma linha diferente e o nome da biblioteca seguido de ponto H (.h) - .h é a extensão do arquivo da biblioteca que vem da palavra inglesa HEADER (cabeçalho) - se você esquecer de colocá-lo o programa não será compilado.
- No exemplo abaixo, vamos incluir duas bibliotecas padrões de C.

Diretivas para o processador - Bibliotecas

- Diretiva `#include` permite incluir uma biblioteca
- Bibliotecas contêm funções pré-definidas, utilizadas nos programas
- Exemplos

`#include <stdio.h>`	Funções de entrada e saída
`#include <stdlib.h>`	Funções padrão
`#include <math.h>`	Funções matemáticas
`#include <string.h>`	Funções de texto

Exemplo 1:

```
#include <stdio.h>
#include <stdlib.h>
/* Um Primeiro Programa */
int main ()
{
        printf ("Ola! Eu estou vivo!\n");
        system("pause");
        return(0);
}
```

Exemplo 2:

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
  int x;
  scanf("%d",&x);
  printf("%d",x);
  system("pause");
  return(0);
}
```