

PROGRAMAÇÃO I

Prof. Luiz Albano

@ luiz.albano@ifsp.edu.br



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus Bragança Paulista

Agenda

Tema: Introdução Linguagem C

- Conceitos Iniciais
- Introdução a Linguagem C
- Ambiente de desenvolvimento: DevC
- Estrutura de um programa em C
- Comando de saída



Conceitos Iniciais

O que é um programa?



Conceitos Iniciais

O que é um programa?

Conjunto de instruções que contém as operações necessárias para, a partir da **entrada de dados, obter ou calcular** um resultado que poderá ser disponibilizado por algum dispositivo de **saída ou gerar uma informação como saída**.





Conceitos Iniciais

Software

- Conjunto de programas que controlam o funcionamento do hardware.
- Programas são construídos a partir de algoritmos.
- Algoritmo é a sequência de instruções/comandos para se atingir um objetivo.
- Depois de finalizado a sequência de instruções de um programa, o algoritmo é convertido para uma linguagem de máquina (compilação).
- O produto final desta construção é o programa.



Conceitos Iniciais

Linguagem de Programação

- Conjunto de regras e/ou representações utilizadas para criar programas.
- Através da linguagem de programação estabelecemos uma comunicação com o computador.
- Permite determinar e controlar ações.
- A proximidade das regras com a linguagem humana determina o nível de abstração da linguagem.
- Atualmente existem três níveis de abstração para linguagens de programação: linguagem de máquina, linguagem de baixo nível e linguagem de alto nível.

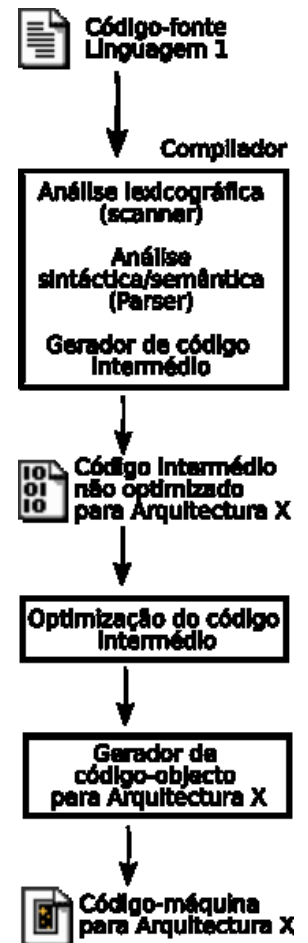
Conceitos Iniciais

Interpretação e Compilação

Todo código escrito em uma linguagem de programação é traduzido para código de máquina.

Esta tradução pode ser feita através do processo de **compilação** ou por **interpretação**.

- **Compilação:** o código é traduzido por um programa denominado compilador e em seguida armazenada de forma que possa ser executado infinita vezes sem a necessidade de nova compilação.
- **Interpretação:** o código é traduzido toda vez que é executado.



Conceitos Iniciais

Interpretação e Compilação

O papel do compilador é, basicamente, “**traduzir**” um código em linguagem de alto nível para um código em linguagem de máquina.



Planejamento de um Programa

ENTRADA

PROCESSAMENTO

SAÍDA

Quais dados eu tenho e/ou tenho que fornecer para o processamento?

Que testes devo fazer com os dados existentes ou fornecidos para produzir a saída desejada?

O que e como devem ser as informações a serem exibidas após o processamento?

Variáveis e constantes

Entrada de dados (scanf), condições de teste (if), repetições de comandos para atender ao processamento (for, while, do while)

Informações e/ou dados produzidos. Exibição em tela com printf, gravação de arquivo, acionamento de atuadores, etc.



© Can Stock Photo



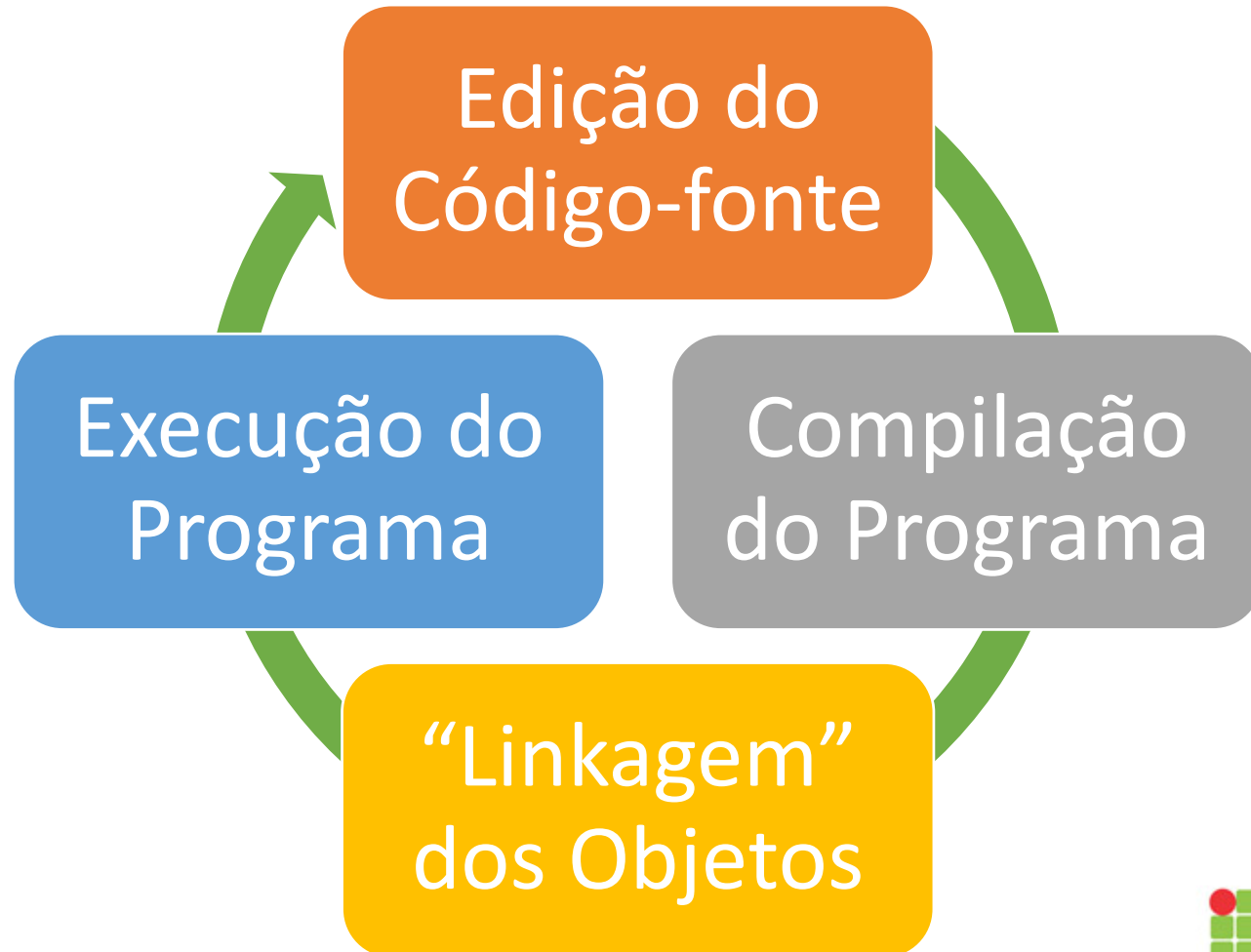
Conceitos Iniciais

Linguagem C

- **A origem do nome é simples:** é a linguagem que sucede a linguagem B (lógica!).
- Desenvolvida em 1972 por Dennis Ritchie nos laboratórios da Bell Telephones.
- Tinha como finalidade permitir a escrita de um sistema operacional (o Unix), utilizando uma linguagem de alto nível, evitando o recurso ao Assembly.
- Padrão ANSI C: estabelecido em 1983, comitê para padronização e correção de erros de compatibilização da linguagem.

Conceitos Iniciais

Ciclo de Desenvolvimento de um Programa em C





Conceitos Iniciais

Linguagem C: Linguagem Estruturada

A característica principal de uma linguagem estruturada é o compartilhamento do código e dos dados.

Habilidade de uma linguagem em seccionar e esconder do resto do programa todas as informações necessárias para se realizar uma tarefa específica.

- Funções (subrotinas)
- Variáveis locais
- Estruturas predefinidas



Ferramenta de Desenvolvimento

DevC++



Ferramenta de Desenvolvimento

DevC++

Em nossa disciplina utilizaremos a IDE DevC++.

IDE

IDE, do inglês *Integrated Development Environment* ou **Ambiente Integrado de Desenvolvimento**, é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de *software* com o objetivo de agilizar este processo.

A ferramenta pode ser baixada em:

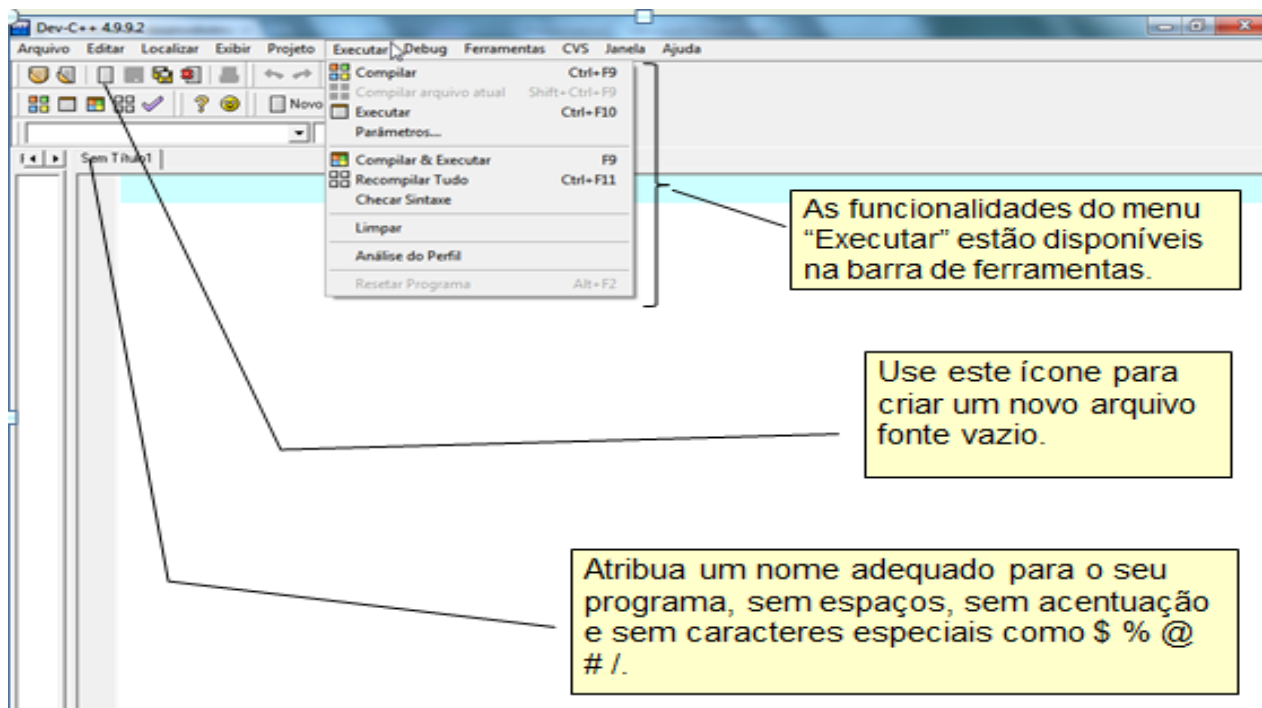
<https://sourceforge.net/projects/orwelldvcpp/>

Ferramenta de Desenvolvimento

DevC++

Para escrever seus programas, após a instalação da IDE, abra a mesma e clique em:

Novo –> Arquivo Fonte





Ferramenta de Desenvolvimento

DevC++

Estrutura básica de um programa em C

```
main()  
{  
  
}
```



Ferramenta de Desenvolvimento

DevC++

Programa de Exemplo:

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

main()
{
    printf("Turma A\n");
    system("pause");
}
```



Ferramenta de Desenvolvimento

DevC++

Fonte x Executável

Entendendo a estrutura do programa exemplo:

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

As duas linhas indicam a inclusão de bibliotecas que possuem as funções que o C utiliza.

Sempre inicie seus programas com essas duas linhas. Ou pelo menos a inclusão da biblioteca **stdio.h**.



Ferramenta de Desenvolvimento

DevC++

Fonte x Executável

Entendendo a estrutura do programa exemplo:

main()

A função `main()` é sempre a primeira a ser executada no programa C. Em todo programa desenvolvido em C, existirá uma função `main()`.



Ferramenta de Desenvolvimento

DevC++

Fonte x Executável

Entendendo a estrutura do programa exemplo:

```
{  
    printf("Turma A\n");  
    system("pause");  
}
```

{ -> É o início de um bloco de comandos no programa. Para toda chave { que inicia um bloco de comandos, teremos uma chave } que será responsável por informar o fechamento desse bloco.

A função **printf()** permite a exibição de mensagens no monitor.

O código especial \n é responsável por fazer saltar uma linha. Mais à frente aprofundaremos os estudos da função printf().



Ferramenta de Desenvolvimento

DevC++

Fonte x Executável

Entendendo a estrutura do programa exemplo:

```
{  
    printf("Turma A\n");  
    system("pause");  
}
```

system("pause");

Possibilita uma pausa no programa a fim de visualizarmos o resultado na tela. Caso contrário, ele seria exibido tão rapidamente que não conseguiríamos vê-lo.



Ferramenta de Desenvolvimento

DevC++

Fonte x Executável

Entendendo a estrutura do programa exemplo:

```
{  
    printf("Turma A\n");  
    system("pause");  
}  
  
}
```

Indica o fim do programa. O fim de main(). Essa chave está fechando o bloco aberto.



Ferramenta de Desenvolvimento

DevC++

Fonte x Executável

Um detalhe importante sobre a linguagem C é que, ao contrário de algumas outras linguagens, em C **há distinção entre caracteres maiúsculos e minúsculos**. Assim, em C, é diferente chamar uma variável de **num** ou **Num**. Portanto, para evitar erros, por padrão, costumamos utilizar apenas caracteres minúsculos nos nomes de variáveis.

Observe também que todos os comandos da linguagem C são escritos apenas **com caracteres minúsculos**. Agora que compreendemos cada linha do nosso primeiro programa em C, vamos abrir o ambiente Dev-C++ seguindo os passos apresentados no início do capítulo e, então, digitar esse programa no ambiente.



Ferramenta de Desenvolvimento

DevC++

Fonte x Executável

Após digitar seu primeiro programa salvar como arquivos fontes de C (C source files).

Depois de salvar o arquivo, devemos compilar e executar o programa a fim de visualizarmos seu resultado. Para compilar e executar o programa, podemos utilizar a tecla **F11** ou acessar o menu **Executar > Compilar & Executar**.

Caso você solicite a compilação antes de salvar o arquivo, automaticamente aparecerá a janela de Salvar arquivo para depois o ambiente compilar seu programa. Neste caso, siga as instruções dadas anteriormente para salvar arquivo.

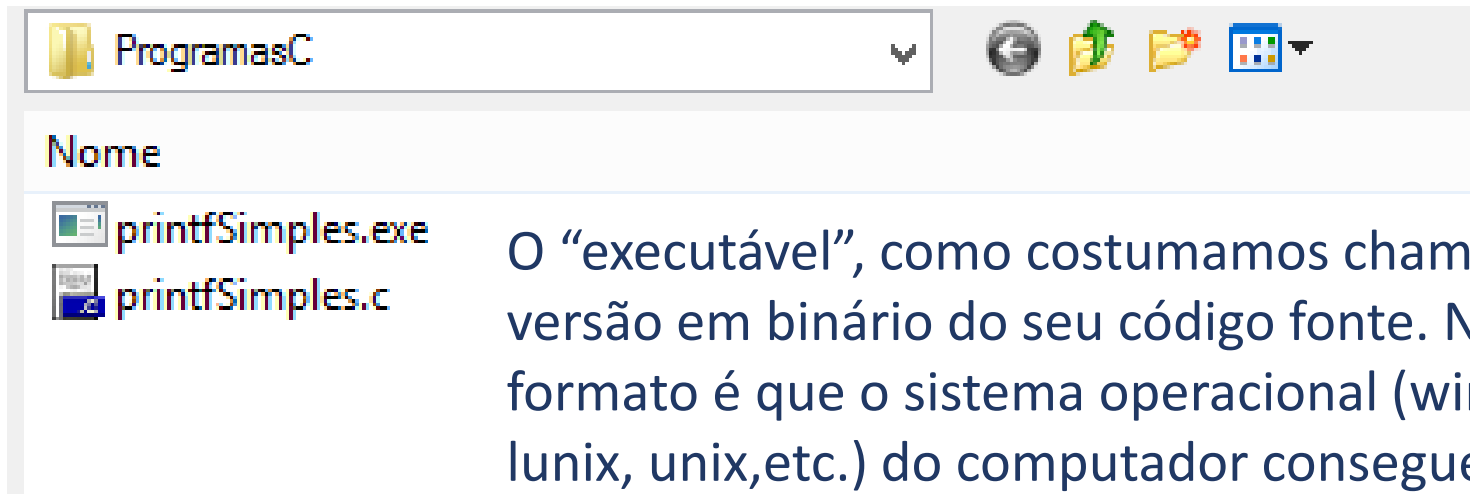


Ferramenta de Desenvolvimento

DevC++

Compilando e Executando

Após o processo de compilação é criado o arquivo referente ao programa executável que recebe a extensão “.exe”.



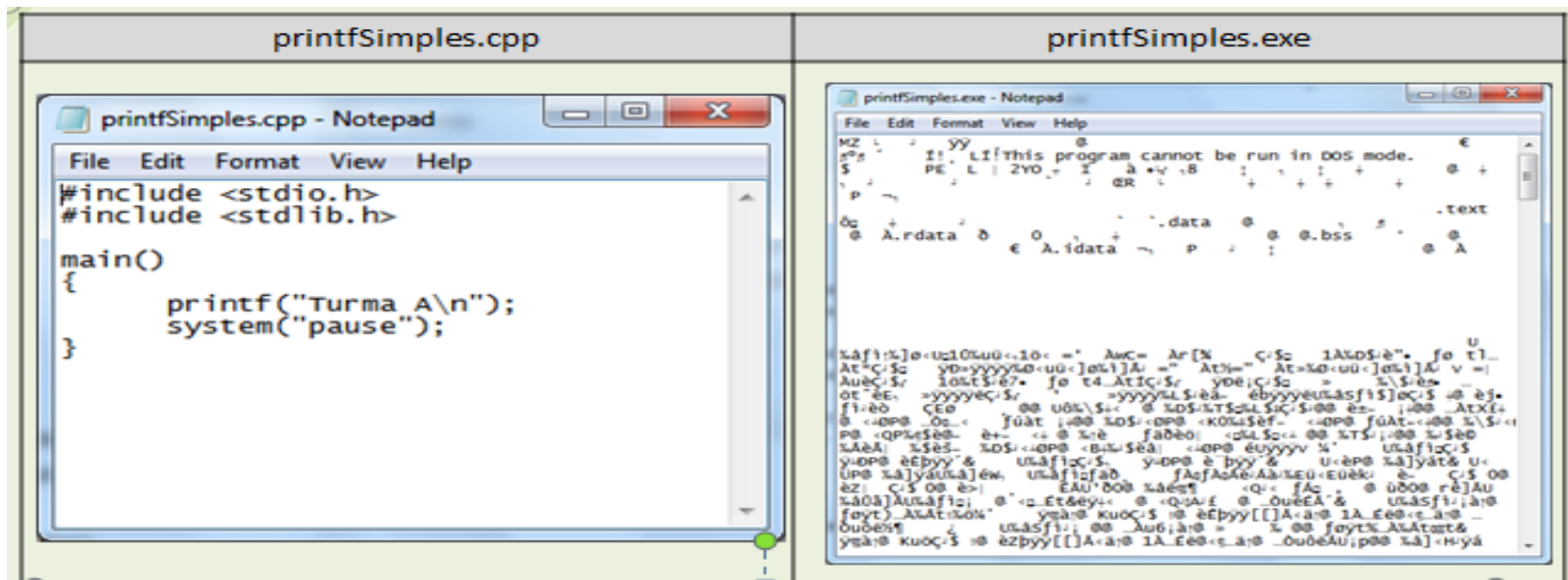
O “executável”, como costumamos chamar, é a versão em binário do seu código fonte. Neste formato é que o sistema operacional (window, linux, unix,etc.) do computador consegue executar as instruções codificadas.

Ferramenta de Desenvolvimento

DevC++

Compilando e Executando

A visualização dos dois arquivos (fonte e executável) fica bem diferente, embora se trate do mesmo programa.

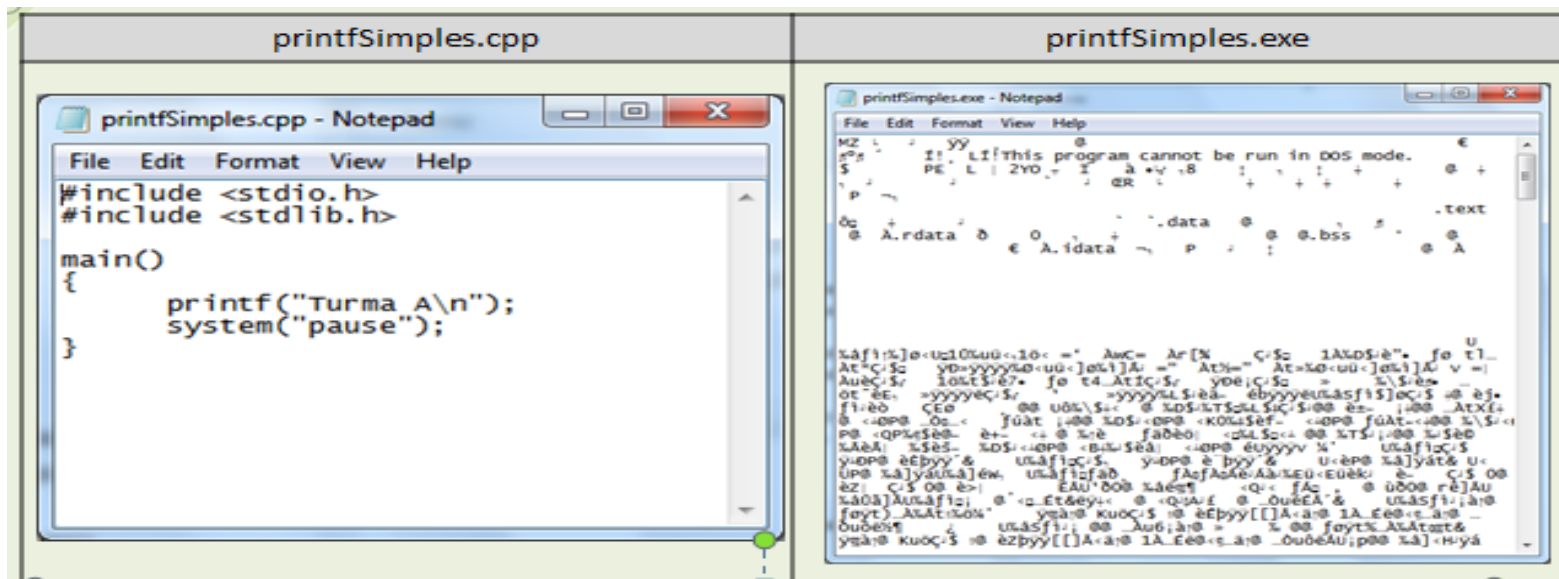


The image displays two side-by-side Notepad windows. The left window, titled 'printfSimples.cpp - Notepad', shows the source code of a C++ program. The code includes `<stdio.h>` and `<stdlib.h>`, and contains a `main()` function that prints 'Turma A\n' and calls `system("pause");`. The right window, titled 'printfSimples.exe - Notepad', shows the assembly code generated from the source file. It includes headers for `File`, `Edit`, `Format`, `View`, and `Help`. The assembly code is organized into sections: `.text` (containing the main function's assembly), `.data` (containing the string 'Turma A\n'), and `.bss` (containing the `system` function's assembly). The assembly code is written in a mix of uppercase and lowercase letters, with some characters in italics, and includes various assembly instructions and directives.

DevC++

Compilando e Executando

Você deve tomar o cuidado de manter a versão do código fonte compatível com o executável para que seja possível corrigir erros ou implementar novas condições no programa sem perder acertos e implementações anteriores.

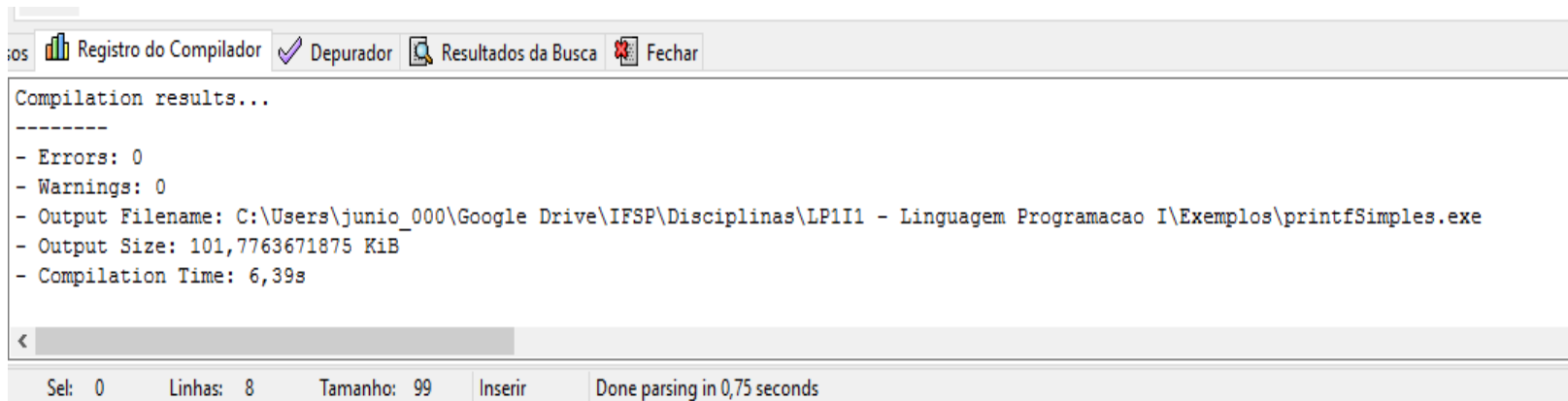


Ferramenta de Desenvolvimento

DevC++

Compilando

Quando você compilar aparecerá na sua tela se ocorreu com sucesso ou se houve algum erro. Exemplo de compilação com sucesso:



The screenshot shows the 'Registro do Compilador' (Compiler Log) window in DevC++. The window has a title bar with buttons for 'Registro do Compilador', 'Depurador', 'Resultados da Busca', and 'Fechar'. The main text area displays the following compilation results:

```
Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\junio_000\Google Drive\IFSP\Disciplinas\LP1I1 - Linguagem Programacao I\Exemplos\printfSimples.exe
- Output Size: 101,7763671875 KiB
- Compilation Time: 6,39s
```

At the bottom of the window, there is a status bar with the following information: 'Sel: 0', 'Linhas: 8', 'Tamanho: 99', 'Inserir', and 'Done parsing in 0,75 seconds'.

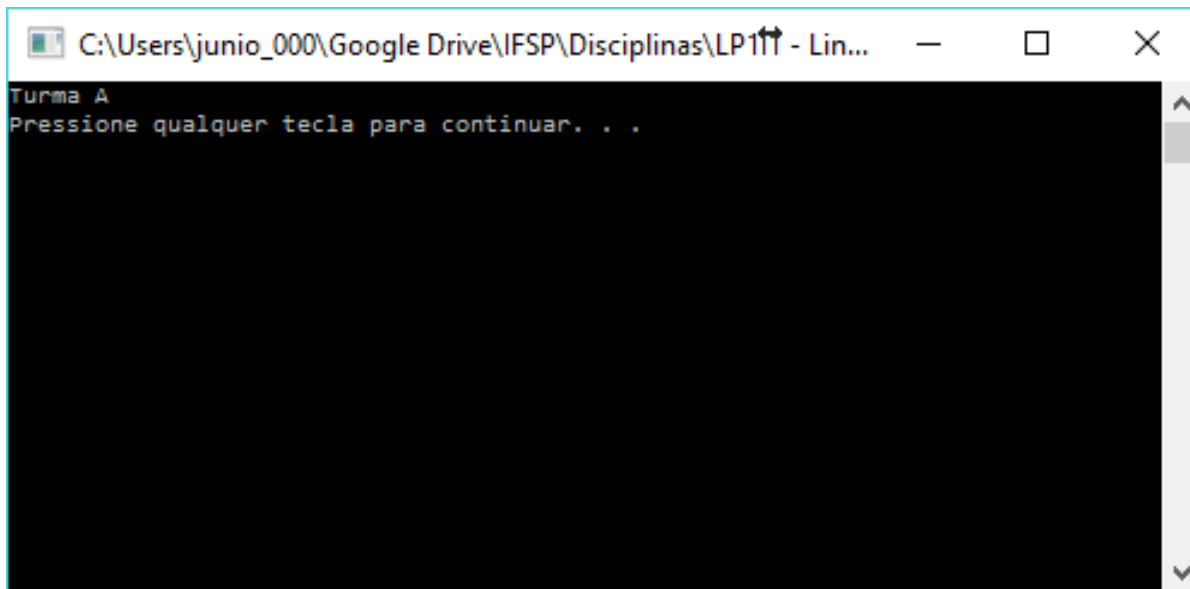


Ferramenta de Desenvolvimento

DevC++

Executando

Quando você executar, tendo como exemplo o programa que fizemos, aparecerá:



```
C:\Users\junio_000\Google Drive\IFSP\Disciplinas\LP1\T1 - Lin...  
Turma A  
Pressione qualquer tecla para continuar. . .
```



Linguagem C

Saída de Dados



Linguagem C

Saída de Dados

Comando: `printf()`

A função `printf()` é uma função que produz a saída de dados para o dispositivo padrão. Esta função permite que uma mensagem formatada seja exibida na tela do computador.

As mensagens devem ser escritas entre aspas duplas.

Para utilizarmos, necessitamos da biblioteca de funções da linguagem C: **`stdio.h`**. Fazemos a inclusão desta biblioteca no início do código fonte através da seguinte declaração:

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




Linguagem C

Saída de Dados

Formatação de Dados

Como dito anteriormente a função **printf()** exibe uma mensagem formatada no dispositivo de saída padrão.

Sempre que quisermos inserir um tipo de dado externo a mensagem, como números, strings ou caracteres, devemos substituir este valor por um código que indica o formato do dado que será formatado na mensagem.

A tabela do próximo slide apresenta os códigos de formatos de dados para os tipos suportados pela linguagem C. Estes códigos devem ser utilizados dentro da mensagem enviada à função `printf()` e na sequência da mensagem, separados por vírgula (,), indicar os respectivos valores de substituição.



Linguagem C

Saída de Dados

Formatação de Dados

Código	Descrição
%c	Utilizado quando a função for apresentar um caractere (tipo char)
%d	Utilizado quando a função for apresentar um número inteiro (tipo int)
%f	Utilizado quando a função for apresentar um número com casas decimais (tipo float)
%s	Utilizado quando a função for apresentar uma cadeia de caracteres (várias letras e palavras).

Exemplos:

```
printf("Valor R$: %f", 954.25);  
printf("Data %d/%d/%d", 5, 3, 2018);  
printf("Letra %c", 'B');  
printf("Disciplina: %s", "Programacao I");
```

Linguagem C

Saída de Dados

Caracteres Especiais

O símbolo \ é utilizado para remover o significado que o caractere representa, adicionando uma função ou representação especial conforme a combinação apresentada na tabela ao lado.

Exemplo no caso das aspas (“), retira o significado do delimitador de strings (conjunto de caracteres).

Caractere Especial	Função / Representação
\n	Nova linha
\t	Tabulação
\f	Salto de página
\a	Sinal sonoro
\r	Retorna o cursor no início da linha
\\	Barra invertida
\0	Caractere nulo
\'	Aspa simples
\"	Aspas duplas



Linguagem C

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.

Linguagem C

Comentários

```
#include <stdio.h>
#include <stdlib.h>
int main( )
{
    int matricula= 2233; /* podemos escrever comentários desta forma */
    float media_final=80.5; // ou apenas com duas barras no início do comentário
    char discipl[10]="Prog I";// a var discipl[10], pode armazenar até 10 caracteres
    printf ("O aluno matricula = %d \n", matricula);
    printf ("Disciplina = %s \n", discipl);
    printf ("Ficou com media = %f \n\n", media_final);
    system("pause");
    return(0);
}
```



Linguagem C

Comentários

Comentário de Linha

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.

Exemplos:

```
//isto é um comentário
```

```
printf("Texto"); //a partir daqui temos um comentário
```



Linguagem C

Comentários

Comentário de Bloco

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.

Exemplos:

```
/*
```

```
Isto é um comentário de bloco
```

```
Podemos continuar escrevendo até que seja declarado
```

```
*/
```

Dúvidas?