

Rodrigo Henrich

rodrigohenrich@faccat.br



Introdução a Linguagem C

Conceitos básicos

- Considerada uma das mais bem sucedidas linguagens de programação de alto nível;
- Linguagens de alto nível são as que se aproximam da linguagem humana;
- Foi criada em 1972 nos laboratórios da Bell por Dennis Ritchie
- Mais tarde revisada e padronizada para o padrão ANSI (American National Standards Institute ou Instituto Nacional Americano de Padrões)
- É uma linguagem estruturalmente simples e de fácil portabilidade, existe um compilador para a maioria das arquiteturas.

Conceitos básicos

- Em geral o compilador C gera códigos mais enxutos e rápidos que outras linguagens
- Trata-se de uma linguagem procedural, ou seja permite decompor problemas complexos em problemas menores e desta forma mais simples
- Possibilita acesso de baixo nível a memória, ou seja permite acesso diretamente ao microprocessador

Conceitos básicos

- Além de permitir o uso de instruções Assembly para programas de desempenho crítico
- Podemos considerar o C uma linguagem multiplataforma, onde um código escrito em uma plataforma pode ser facilmente recompilado para outra, sem grande modificações.

Ambiente de programação

Usando um editor de texto simples

- De forma muito simples o código em C pode ser escrito em qualquer editor de texto
 - Notepad (Windows)
 - Notepad++ (Windows)
 - Sublime Text (Multiplataforma)
 - Vim (Terminal Linux)
 - Gedit (Editor de texto Gnome)
 - CLion (Multiplataforma)
 - Visual Studio code (Multiplataforma)
 - Code Blocks (Windows)

Usando um editor de texto simples

- De uma forma simples há inúmeros editores de código que podem ser utilizados
- De modo geral alguns destes editores oferecem suporte a linguagem C e inúmeras outras é o caso do
 - Sublime
 - Notepad++
 - Clion
 - Visual Studio code
 - Code Blocks

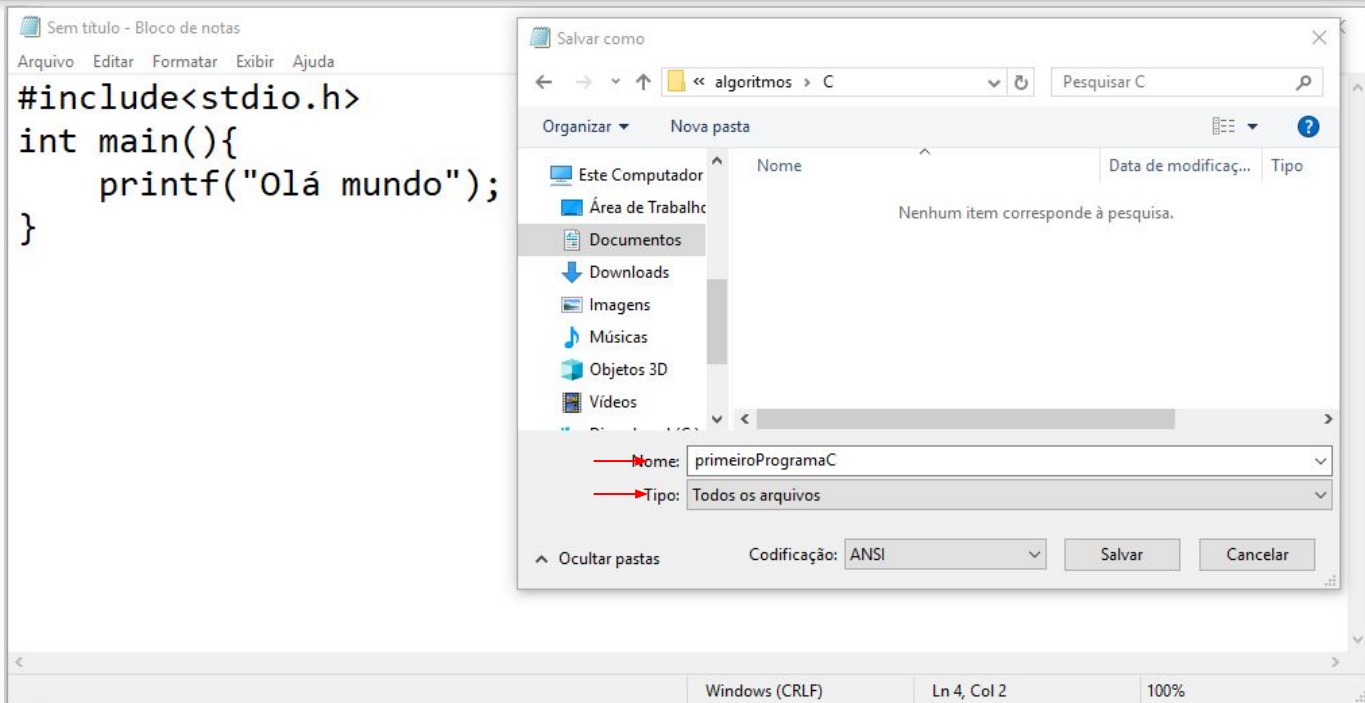
Usando um editor de texto simples

- No entanto na maioria deles é preciso compilar o código externamente
- Para isso temos que usar um compilador
- O mais usado deles é o gcc (GNU C Compiler)

Notepad

- Ele permite editar código de qualquer linguagem
- Muito prático, já vem com windows
- Não dá suporte a formatação de código
- É preciso tomar cuidado na hora de salvar para selecionar a extensão do arquivo

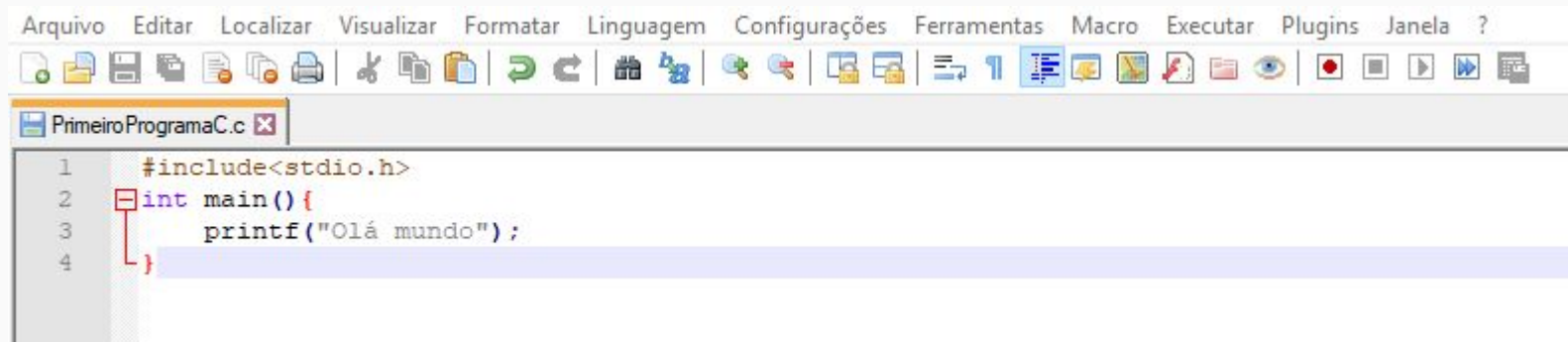
Notepad



Notepad++

- Assim como o notepad que acompanha o Windows, o notepad++ é muito simples de usar
- No menu linguagem é possível selecionar a linguagem que deseja salvar o programa
- Ele irá selecionar um esquema de cores adequado para a linguagem escolhida
- Ou basta salvar o arquivo com a extensão correta, em nosso caso .c que ele irá colorir o código automaticamente

Notepad++



The image shows the Notepad++ application window. The title bar reads "PrimeiroProgramaC.c". The menu bar includes "Arquivo", "Editar", "Localizar", "Visualizar", "Formatar", "Linguagem", "Configurações", "Ferramentas", "Macro", "Executar", "Plugins", "Janela", and "?". The toolbar contains various icons for file operations, editing, and execution. The editor area displays the following C code:

```
1  #include<stdio.h>
2  int main(){
3      printf("Olá mundo");
4  }
```

The code is syntax-highlighted, with the opening curly brace of the `main` function on line 2 highlighted in red. The entire code block is currently selected, indicated by a light blue background.

SublimeText

- Diferente dos anteriores o Sublime Text não é gratuito
- Ele permite o uso sem registro, porém emite alertas frequentes do seu status
- Nele para codificar em C basta salvar o arquivo fonte com a extensão .c
- Ele tem um excelente suporte a formatação de código
- Existe uma versão portable, que pode ser levada facilmente em um pendrive
- O código salvo precisa ser compilado com um compilador externo

SublimeText



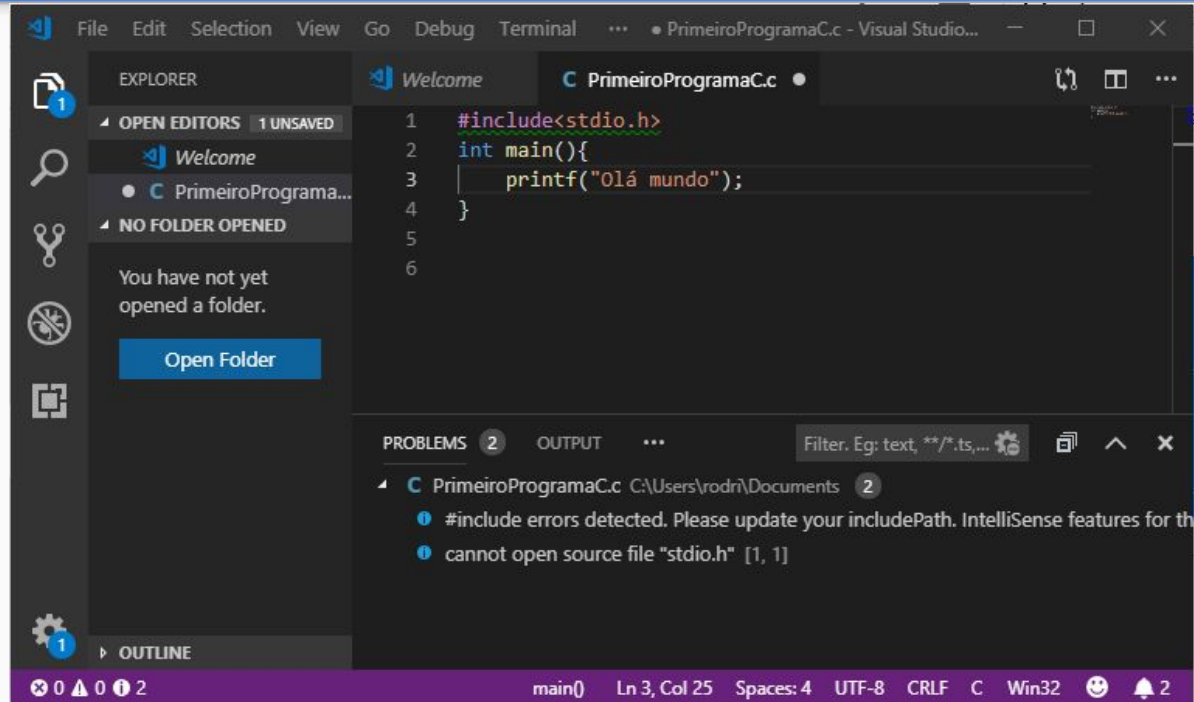
The screenshot shows the Sublime Text 2 editor window. The title bar indicates the file path is C:\Users\rodri\Documents\OlaMundo.c and the application is Sublime Text 2 (UNREGISTERED). The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. A single tab labeled OlaMundo.c is open. The editor displays the following C code:

```
1  #include<stdio.h>
2  int main(){
3      printf("Olá mundo");
4  }
5
6
```

Visual Studio Code

- Gratuito e de código fonte aberto
- Para criar um arquivo em C basta clicar em novo arquivo
- Depois salvá-lo como .c
- Ele pode ser configurado para compilar o código, mas para isso antes é preciso ter um compilador instalado no sistema
- Permite gerenciar projetos, contendo vários arquivos de código fonte

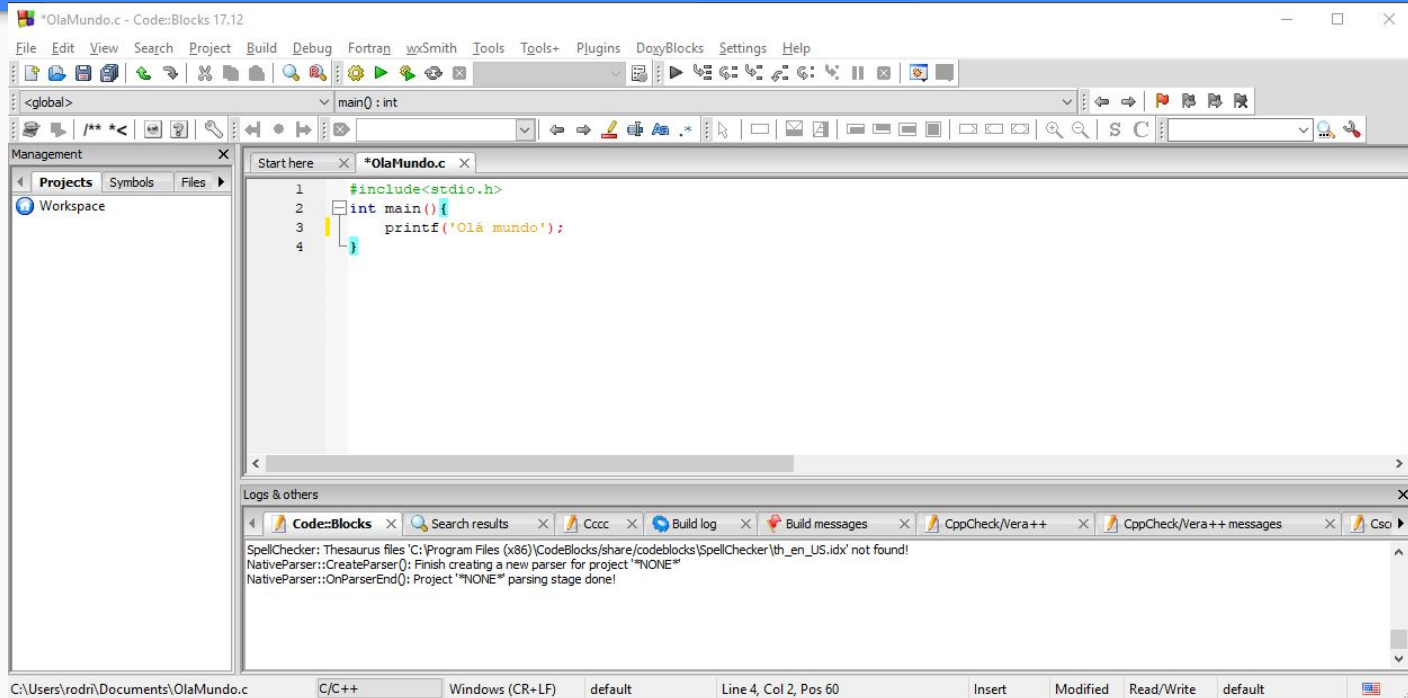
Visual Studio Code



Code Blocks

- Gratuito
- Permite editar o código
- Permite compilar o código
 - Depende de compilador externo
- Suporte a formatação de código

Code Blocks



Compilando o código

- Depois de produzido nosso código fonte é a hora de compilar o código fonte
- Para isso vamos usar o gcc
- Podemos chamar ele via linha de comando

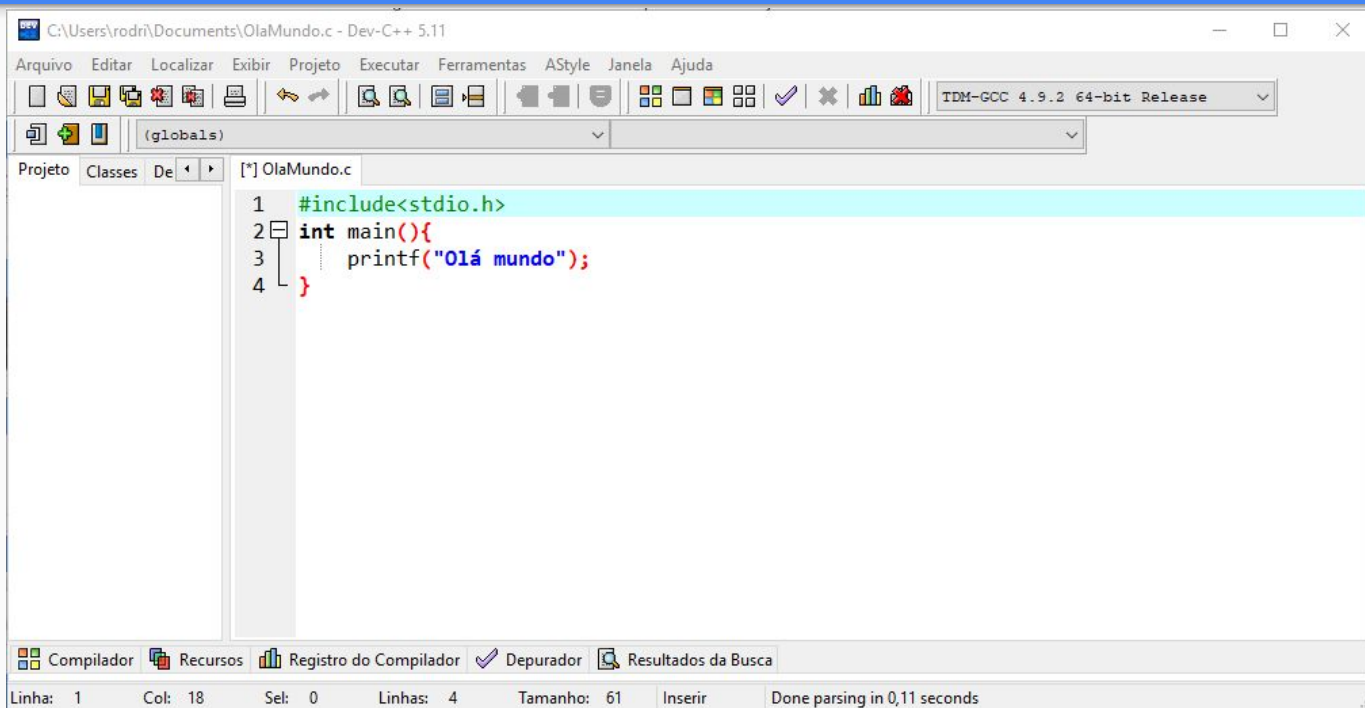
gcc via linha de comando

- gcc <**código.c**> será compilado no arquivo **a.out**
- gcc <**código.c**> -o <**arquivo.exe**>
- Caso exista algum erro de sintaxe no código será mostrado na tela.

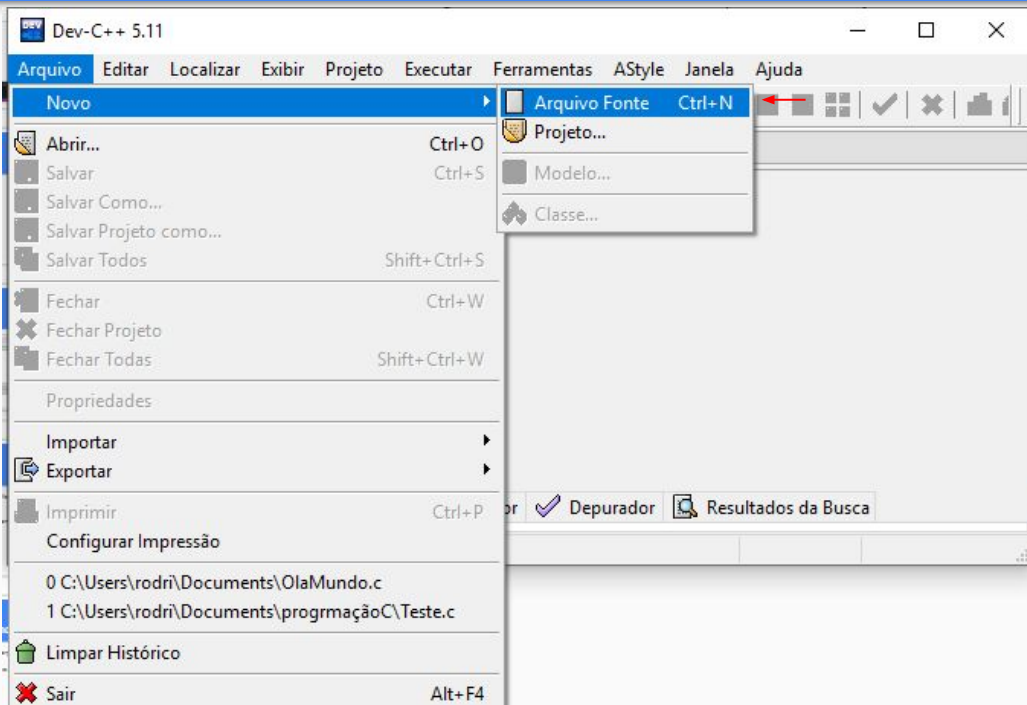
Ambiente integrado de desenvolvimento

- Uma IDE permite vários recursos de desenvolvimento
 - Além de desenvolver código
 - Permite compilar e executar o código criado
 - Dispõe de ferramentas de debug
- Neste caso vamos usar o DevC++, uma excelente ferramenta para desenvolvimento de projetos em C

DevC++

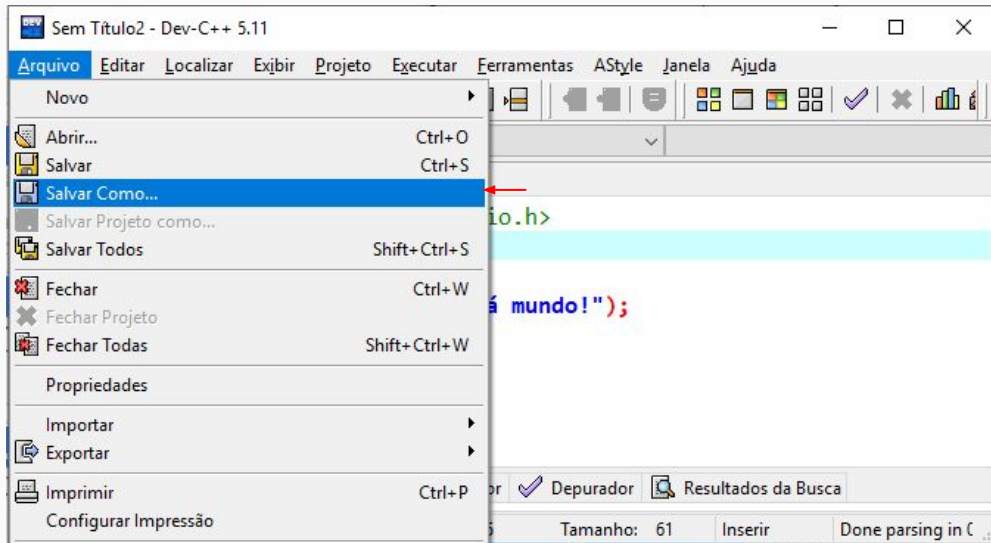


Usando o DevC++



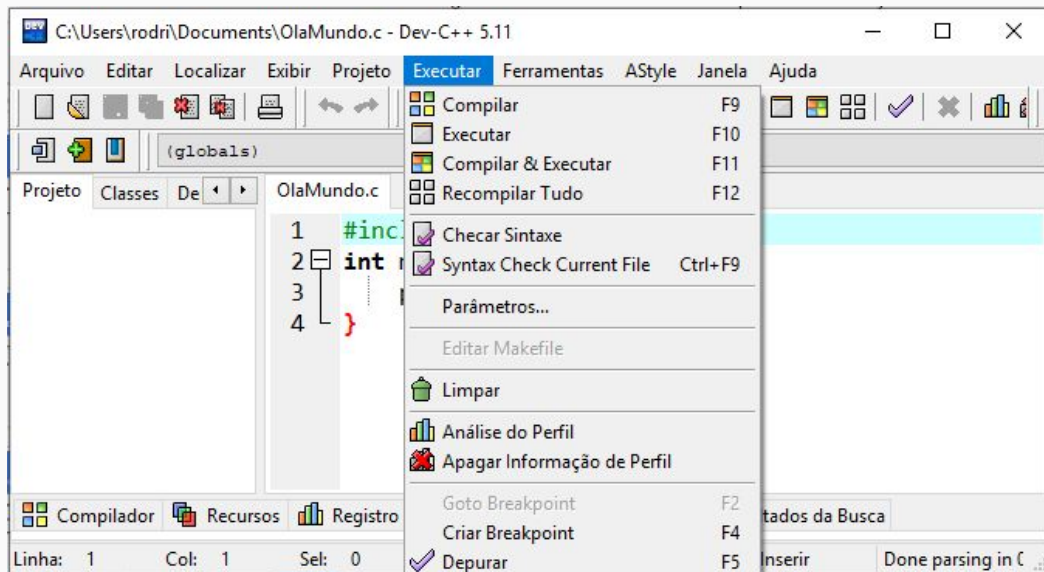
A seta ao lado aponta como criar um novo arquivo fonte


Usando o DevC++




A seta mostra como salvar o novo arquivo fonte

Usando o DevC++




 Compilar F9

Compila do código que estiver aberto

 Executar F10

Executa o código recém compilado

 Compilar & Executar F11

Compila e executa o código

Primeiro programa em C

- Abra o DevC++ em seu computador digite e execute o código abaixo

```
1  #include<stdio.h>
2  int main(){
3      printf("Ola mundo");
4  }
```

Primeiro programa em C

- O resultado esperado é o seguinte

```
Ola mundo
-----
Process exited after 0.02594 seconds with return value 9
Pressione qualquer tecla para continuar. . .
```

Estrutura do programa

```
1  #include<stdio.h>
2  #include<conio.h>
3
4  int main(){
5      printf("Ola mundo");
6      system("pause");
7      return 0;
8  }
```

Inclusão das bibliotecas de funções e declarações globais

Início do programa

Código do programa

Fim do programa

Estrutura do programa

- Logo nas primeiras linhas são incluídas as definições globais do nosso código fonte, bibliotecas, constantes, variáveis globais
- A inclusão das bibliotecas segue a seguinte sintaxe
 - `#include<nome_biblioteca.h>`
- Logo em seguida a declaração da função **int main()** de momento ela será a única função por enquanto
- em seguida **{** é o marcador de bloco de código semelhante ao início dos nossos algoritmos.

Estrutura do programa

- O comando `printf("Olá mundo");` permite mostrar informações na tela, assim como no algoritmo as frases e constantes devem estar entre ""
- A função `printf` está definida na biblioteca de funções `stdio.h`, então sem ela incluída, essa função não funciona.
- Todos os comandos são finalizados com `;` para marcar o final do comando
- A função `system()`, permite enviar um comando ao sistema operacional, por exemplo
 - `system("pause");` faz o sistema parar e esperar que uma tecla seja pressionada;
 - `system("cls");` limpa a tela do terminal antes de escrever mais informações;

Estrutura do programa

- () definem início e fim da declaração de parâmetros de uma função, no exemplo a função system, espera apenas um parâmetro, que é o nome da função a ser executada
 - system("pause");
- return 0; encerra a nossa função main retornando 0 ao sistema, dessa forma o sistema entende que nosso problema terminou sem problemas
- } faz o fechamento da função main e encerra nosso programa.

Estrutura do programa

- Detalhe que para funções as chaves `}` são obrigatórias mesmo que nossa função tenha apenas um comando.

```
1  #include<stdio.h>
2
3  void main(){
4      printf("Ola mundo");
5  }
```

Indentação de código

- Outro assunto relevante para ser tratado aqui é a indentação de código;
- Mesmo não influenciando no funcionamento do programa, ele permite uma melhor leitura
 - Algumas linguagens como o Python dependem da indentação para o correto funcionamento
- Basicamente a palavra indentação significa um recuo em relação a margem
- Aqui vamos usar ele sempre que algum comando estiver dentro de outro ou for consequência de outro, assim como nos algoritmos

Indentação de código



```
1  #include<stdio.h>
2
3  void main(){
4      printf("Ola mundo");
5      if(a == 0)
6          printf("a vale 0");
7      else{
8          printf("a não vele 0");
9          printf("qual será o valor de a?");
10     }
11 }
```



```
1  #include<stdio.h>
2
3  void main(){
4      printf("Ola mundo");
5      if(a == 0)
6          printf("a vale 0");
7      else{
8          printf("a não vele 0");
9          printf("qual será o valor de a?");
10     }
11 }
```

Comentário no código

- Em alguns casos pode ser necessário explicar ou anotar determinadas informações no meio do código fonte.
- Não podemos escrever essas informações simplesmente porque isso causaria um erro no programa
- Para isso existem os comentários
- Esses são trechos de código que serão ignorados pelo compilador na hora de compilar o programa.

Comentário no código

- Em C e em muitas linguagens de programação um bloco de comentário segue a seguinte sintaxe
- *//Duas barras fazem com que a linha seja comentada*
- */*Permite comentar um bloco inteiro de informação, desta forma não apenas uma linha mas uma parte inteira será comentada, até que o compilador encontre os caracteres */*

Comentário no código

```
1  #include <stdio.h>
2  int main(){
3      int x = 10;
4      //Isso é uma linha comentada em C
5      printf("Isso será impresso na tela!");
6      /*Agora estou comentando um pequeno bloco de texto
7       este, não será interpretado pelo compilador, mas
8       ajuda para que possa explicar determinadas
9       informações no meio do código fonte*/
10     char a = "A";
11 }
```