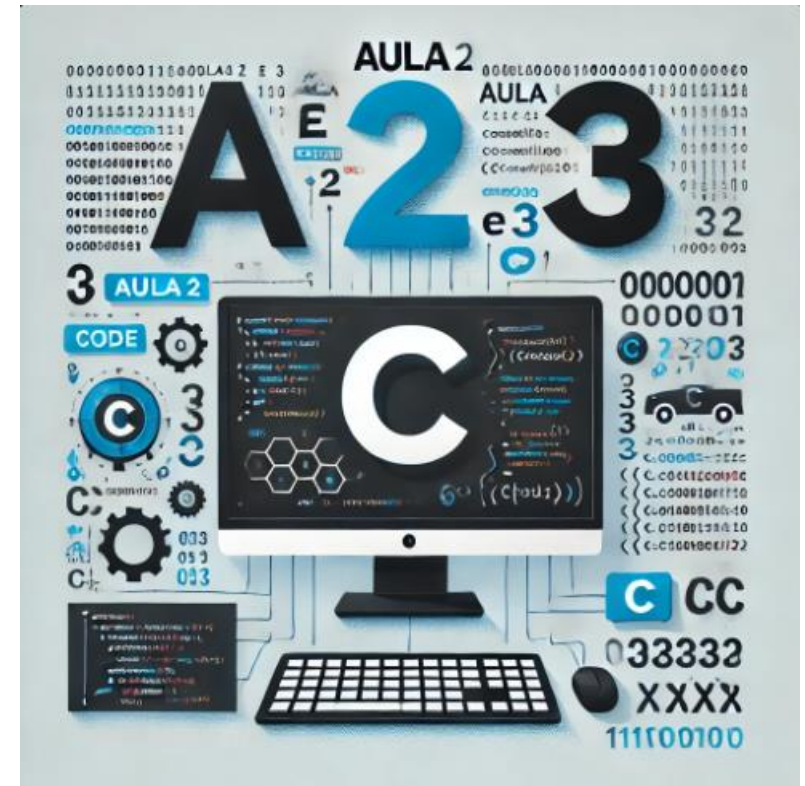


AULA 2 e 3

Conteúdos Específicos

- Conceito de Função
- Nomenclatura de uma Função
- Estrutura de uma Função
- A instrução Return
- Função Void
- Argumentos de uma Função
- Exemplos práticos



Sumário da aula 2 e 3

- Conceito de função e diferenças entre subprogramas e funções.
- Tipos de funções:
 - Com retorno de valor.
 - Do tipo void (sem retorno de valor).
- Nomenclatura e características das funções.
- Argumentos de uma função.
- Resolução de exercícios práticos conduzidos.



SUBPROGRAMA vs FUNÇÃO

Diferenças entre SUBPROGRAMA E FUNÇÃO

- Um **subprograma** é um **bloco de código independente**, ou seja fora da **função main()** e que **executa uma tarefa específica** como por exemplo calcular uma soma.
- Uma função é um tipo específico de subprograma que retorna um valor - **Exemplo 1**.
- Em C os **subprogramas** são **funções** que retornam um valor (return) ou não retornam valor através da função (void) - **Exemplo 2**.

Exemplo 1

```
4 int soma(int a, int b) {  
5     return a + b;  
6 }
```

Exemplo 2

```
4 void saudacao()  
5 {  
6     printf("Olá! Bem-vindo!\n");  
7 }
```

TIPOS DE SUBPROGRAMAS

Funções com Retorno de valor:

- São funções que executam uma tarefa e retornam um valor ao programa que as chamou.
- O tipo de retorno pode ser um número inteiro, ponto flutuante, caracter, entre outros.
- São usadas quando é necessário calcular ou obter um resultado que será usado posteriormente.
- Para serem executadas têm de ser chamadas.

```
Soma.cpp
1  #include <stdio.h>
2
3  int soma(int a, int b)
4  {
5      return a + b;
6  }
7
8  int main()
9  {
10     int resultado = soma(5, 3);
11     printf("Resultado da soma: %d\n", resultado);
12     return 0;
13 }
14
```

TIPOS DE SUBPROGRAMAS

Funções Void sem Retorno de valor:

- São funções que executam uma tarefa, mas **não retornam nenhum valor.**
- São utilizadas para:
 - Exibição de mensagens;
 - Alteração de variáveis;
 - Validações.

```
[*] Soma.cpp Void.cpp
1  #include <stdio.h>
2
3  void exibirMensagem()
4  {
5      printf("Ola, mundo!\n");
6  }
7
8  int main()
9  {
10     exibirMensagem();
11     return 0;
12 }
```

RELEMBRANDO

Embora ainda sem saber como escrever uma função, já as temos utilizado ao longo dos nossos programas.

Exemplos:

- `main()` - Função principal do programa.
- `printf(...)` – Função de **escrita associada à biblioteca `stdio.h`** – (Output).
- `scanf(...)` – Função de leitura (**também associada à biblioteca `stdio.h`**) e armazenamento – (Input).

```
[*] Sem Titulo1
1  #include <stdio.h>
2
3  int soma(int a, int b) {
4      return a + b;
5  }
6
7  int main() {
8      int a, b;
9
10     printf("Introduza o primeiro número: ");
11     scanf("%d", &a);
12
13     printf("Introduza o segundo número: ");
14     scanf("%d", &b);
15
16     int resultado = soma(a, b);
17     printf("Resultado da soma: %d\n", resultado);
18
19     return 0;
20 }
```

NOMENCLATURA DE FUNÇÕES

Nome de uma função:

- Deve seguir as mesmas regras das variáveis;
- Ser único no programa;
- Fácil leitura.



NOMENCLATURA DE FUNÇÕES

Relembrando as regras:

- O nome de uma variável/função pode ser constituído por letras do alfabeto (minúsculas, maiúsculas), dígitos e ainda pelo carácter underscore.
- O primeiro carácter não pode ser um dígito. Terá de ser uma letra ou o carácter underscore.
- É desaconselhável a utilização do underscore para inicio da variável/função.
- Uma variável/função não pode nunca ter um nome que esteja reservado à própria linguagem C como por exemplo: float, if, for etc....

CARACTERÍSTICAS DE UMA FUNÇÃO

- Cada função precisa de ter um nome único para ser invocada no programa.
- Uma função pode ser chamada a partir de outras funções.
- A função deve realizar **uma única tarefa bem definida**.
- Deve comportar-se como uma "caixa negra": **o importante é o resultado**, não o processo interno.
- O código da função deve ser **o mais genérico possível** para ser reutilizável.
- Pode receber **argumentos** para adaptar-se a diferentes situações.
- Pode **retornar um valor** como **resultado** do seu processamento.



PARÂMETROS E ARGUMENTOS

Características:

- **Argumento:** Variável que serve de cálculo à função.
- **Passagem de Argumentos:** Os argumentos são passados para a função entre parênteses e separados por vírgulas.

Exemplo : soma(10, 20).



ARGUMENTOS DE UMA FUNÇÃO

O que acontece neste exemplo?

Há retorno de valor?

Ou há impressão de um resultado?

```
2
3 void saudacao(char nome[], int idade)
4 {
5     printf("Olá, %s tens %d anos.\n", nome, idade);
6 }
7 int main()
8 {
9     saudacao("João", 25);
10    saudacao("Maria", 30);
11
12    return 0;
13 }
```



ARGUMENTOS DE UMA FUNÇÃO

O que acontece neste exemplo?

Há retorno de valor?

Ou há impressão de um resultado?

```
1  # include <stdio.h>
2
3  int soma(int a, int b)
4  {
5      return a + b;
6  }
7
8  int main()
9  {
10     printf("%d\n", soma(3,5));
11     return 0;
12 }
```



VARIÁVEIS LOCAIS E GLOBAIS

Variável local:

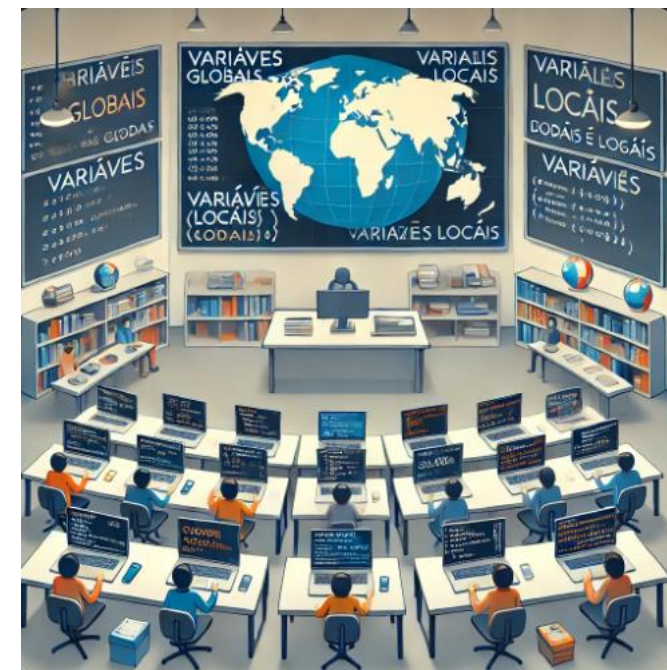
- É definida **dentro da função**.
- A variável é válida e assume o valor atribuído dentro da função.

```
2 void funcao() {  
3     int local = 10; // Variável local  
4     printf("Variável local: %d\n", local);  
5 }
```

Variável Global:

- É definida **fora da função**.
- A variável é válida e assume valor atribuído em todo o programa.

```
1  
2 int global = 20; // Variável global  
3  
4 void funcao() {  
5     printf("Variável global na função: %d\n", global);  
6 }
```



NOMENCLATURA DE FUNÇÕES

Exercício prático conduzido:

1. Função que permite calcular a soma de dois números inteiros.
2. Função sem retorno que imprime uma mensagem de boas-vindas.
3. Função que permite verificar se um número é par ou ímpar.

1

```
[*] Sem Título1  
1  
2 int soma(int a, int b)  
3 {  
4     return a + b;  
5 }
```

2

```
[*] Sem Título1  
1  
2 void mensagemBoasVindas()  
3 {  
4     printf("Bem-vindo ao programa de exemplos de funções!\n");  
5 }
```

3

```
[*] Sem Título1  
1 void verificaParImpar(int numero)  
2 {  
3     if (numero % 2 == 0)  
4         printf("%d é par.\n", numero);  
5     else  
6         printf("%d é ímpar.\n", numero);  
7  
8 }
```

PARÂMETROS E ARGUMENTOS

Exercício explicativo:

- Escreva um programa em C que utilize uma função que receba dois argumentos para calcular a área do retângulo e retornar o resultado.
- Identificação das variáveis.
- Escreva as funções necessárias da biblioteca de funções do C para que o utilizador possa na consola calcular os valores pretendidos.

```
[*] Sem Titulo1
1  #include <stdio.h>
2
3  float calculaAreaRetangulo(float largura, float altura)
4  {
5      return largura * altura;
6  }
7
8  int main()
9  {
10     float largura, altura;
11
12
13     printf("Digite a largura do retângulo: ");
14     scanf("%f", &largura);
15
16     printf("Digite a altura do retângulo: ");
17     scanf("%f", &altura);
18
19
20     float area = calculaAreaRetangulo(largura, altura);
21
22
23     printf("A área do retângulo é: %.2f\n", area);
24
25     return 0;
26 }
```

FUNCIONAMENTO DE UMA FUNÇÃO

- **Execução sob Invocação:** O código de uma função só é executado quando esta é invocada (chamada) no programa.
- **Suspensão Temporária do Programa Principal:** Quando a função é chamada, a execução do programa principal é "suspensa" temporariamente. A função executa as suas instruções.
- **Retorno ao Ponto de Invocação:** Após a função terminar, o controlo de execução volta ao local no programa onde a função foi invocada.
- **Recebimento de Valores:** A função pode receber valores que influenciam o seu comportamento.
- **Devolução de Resultado:** A função pode retornar ou não um resultado para o programa que a chamou.

VANTAGENS DE UTILIZAR FUNÇÕES

- **Reaproveitamento de Código:** Permite reutilizar código já escrito, seja por si ou por outros programadores.
- **Evita Repetição de Código:** Previne a repetição de blocos de código em várias partes do programa.
- **Facilidade de Alteração:** Ao modificar uma função, a alteração é feita apenas dentro desta, tornando a manutenção mais rápida e eficiente.
- **Organização do Programa:** Mantém os blocos do programa menores e mais organizados, facilitando a compreensão.
- **Leitura Facilitada:** Melhora a legibilidade do código fonte.
- **Separação de Tarefas:** Permite separar o programa em partes independentes, tornando-o mais modular.

CONCEITO DE FUNÇÃO

- Na programação em C, **todos os programas** devem ter a **função main()**, independentemente de outras funções que incluam.
- A **execução de um programa começa sempre na main()**, mesmo que esta não esteja no início do código.
- **Todas as funções só serão executadas se forem chamadas pela main()**.



CONCEITO DE FUNÇÃO

- **Uma função é um bloco de código** que pode ser chamado para executar uma tarefa.
- Permite **agrupar instruções** o que facilita a leitura do código.
- Permite **decompor códigos extensos** em pedaços mais pequenos e reutilizáveis.
- Existem funções mais simples e outras mais complexas. As funções podem conter uma instrução ou várias instruções.

O que significa isto?

Sintaxe em C:

```
nomeDaFuncao()
```

```
{
```

```
    < bloco de  
    instruções ; >
```

```
}
```

FUNÇÕES COM INSTRUÇÕES

- Uma função com **uma instrução**:

```
int soma(int a, int b)
{
    return a + b;
}
```

- Uma função com **mais instruções**:

```
int calculo(int a, int b)
{
    int soma;
    soma = a + b;
    soma = soma + 1;
    return soma;
}
```

ESTRUTURA De UMA FUNÇÃO

- Uma função com várias instruções:

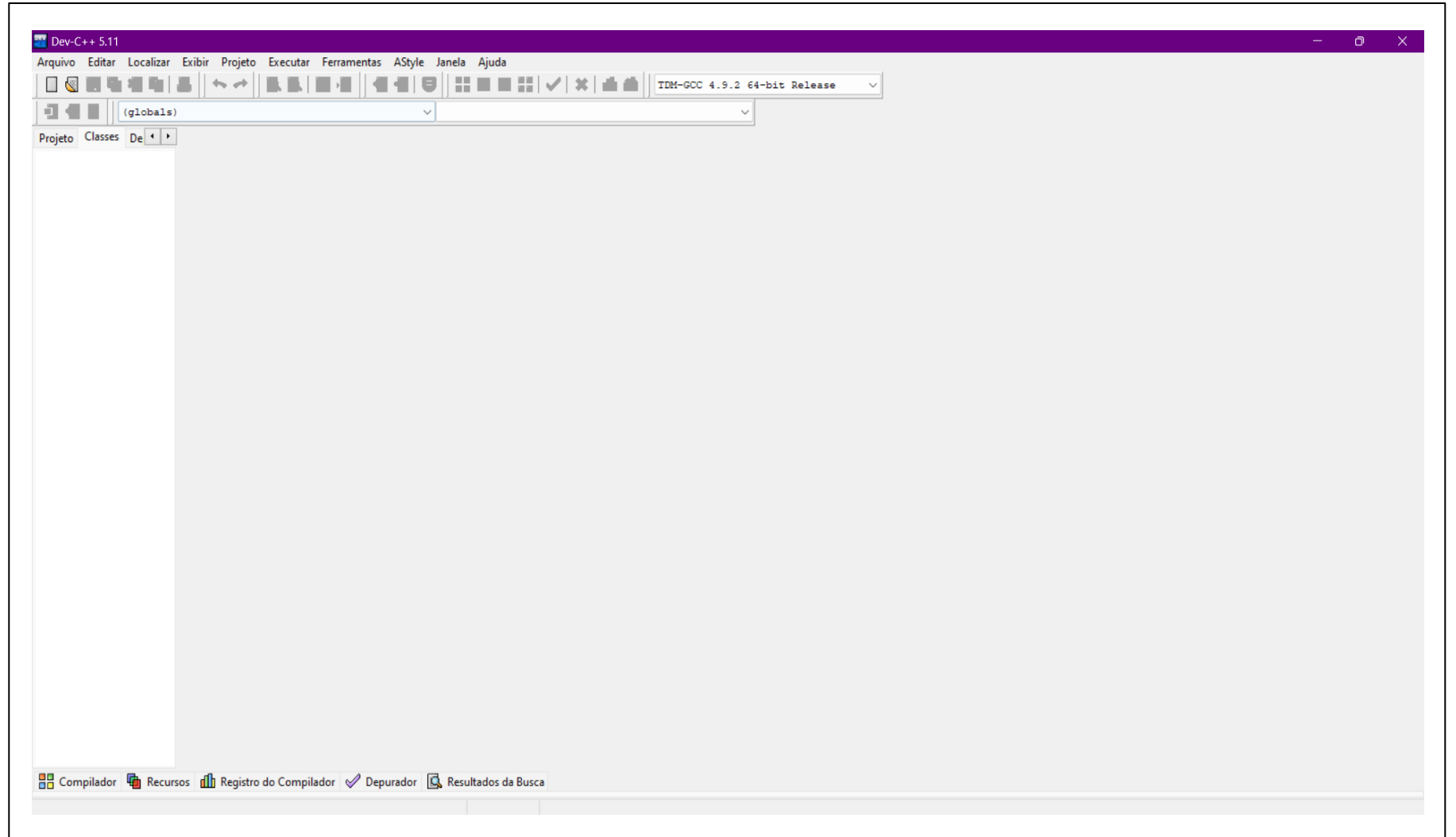
```
int calculo(int a, int b)
{
    int soma;
    soma = a + b;
    soma = soma + 1;
    return soma;
}
```

- Nome da função: calculo
- Cabeçalho da função: int calculo(int a, int b).
- O cabeçalho inclui o tipo de retorno (int), o nome da função (calculo), e os argumentos de entrada (int a, int b).
- As variáveis – São valores que são passados para a função.

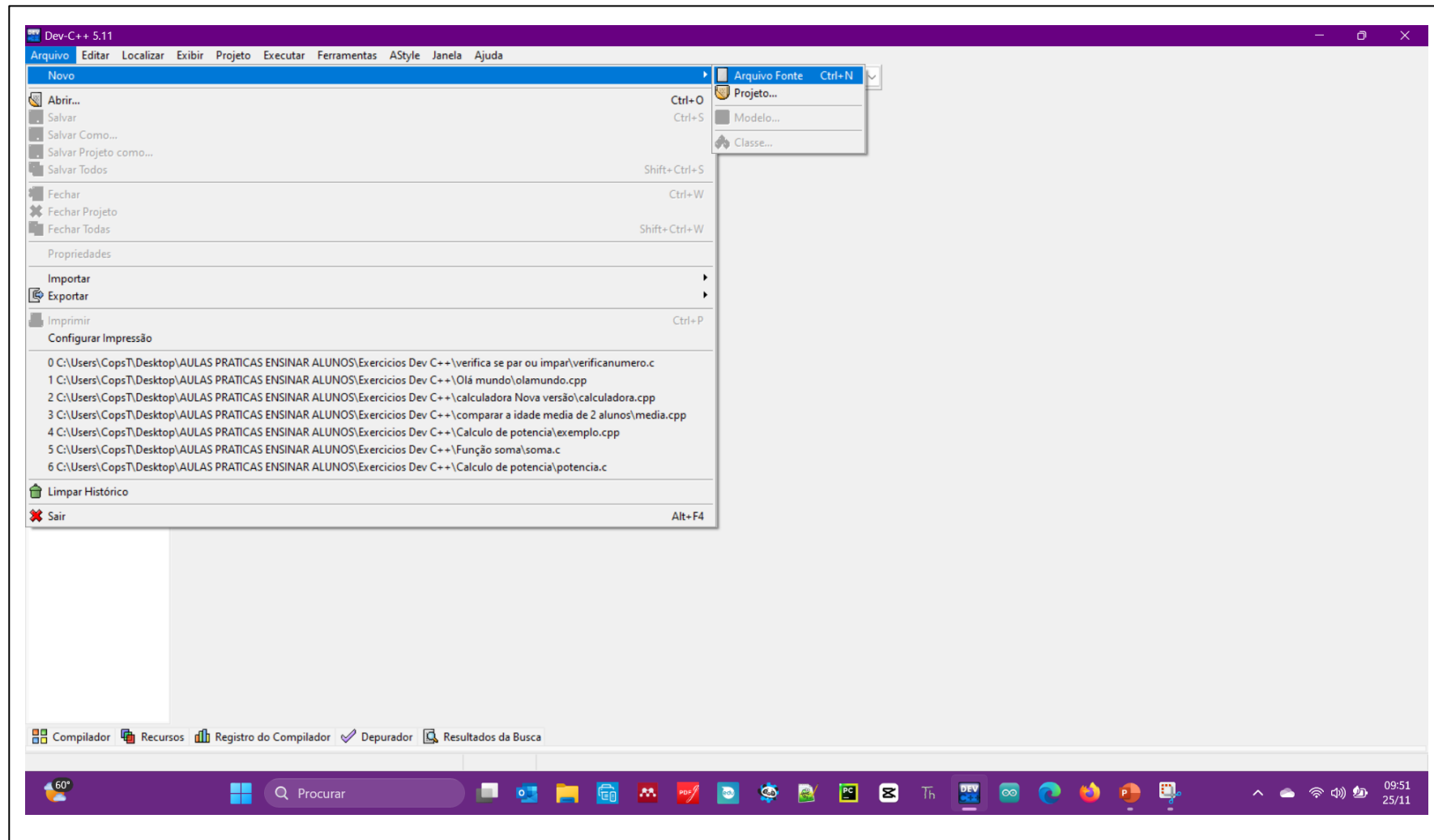
INICIO DO EXERCICIO 1 CONDUZIDO STEP BY STEP

- Vamos criar um programa em C que utilize uma função para calcular a soma de dois números inteiros à escolha do utilizador.

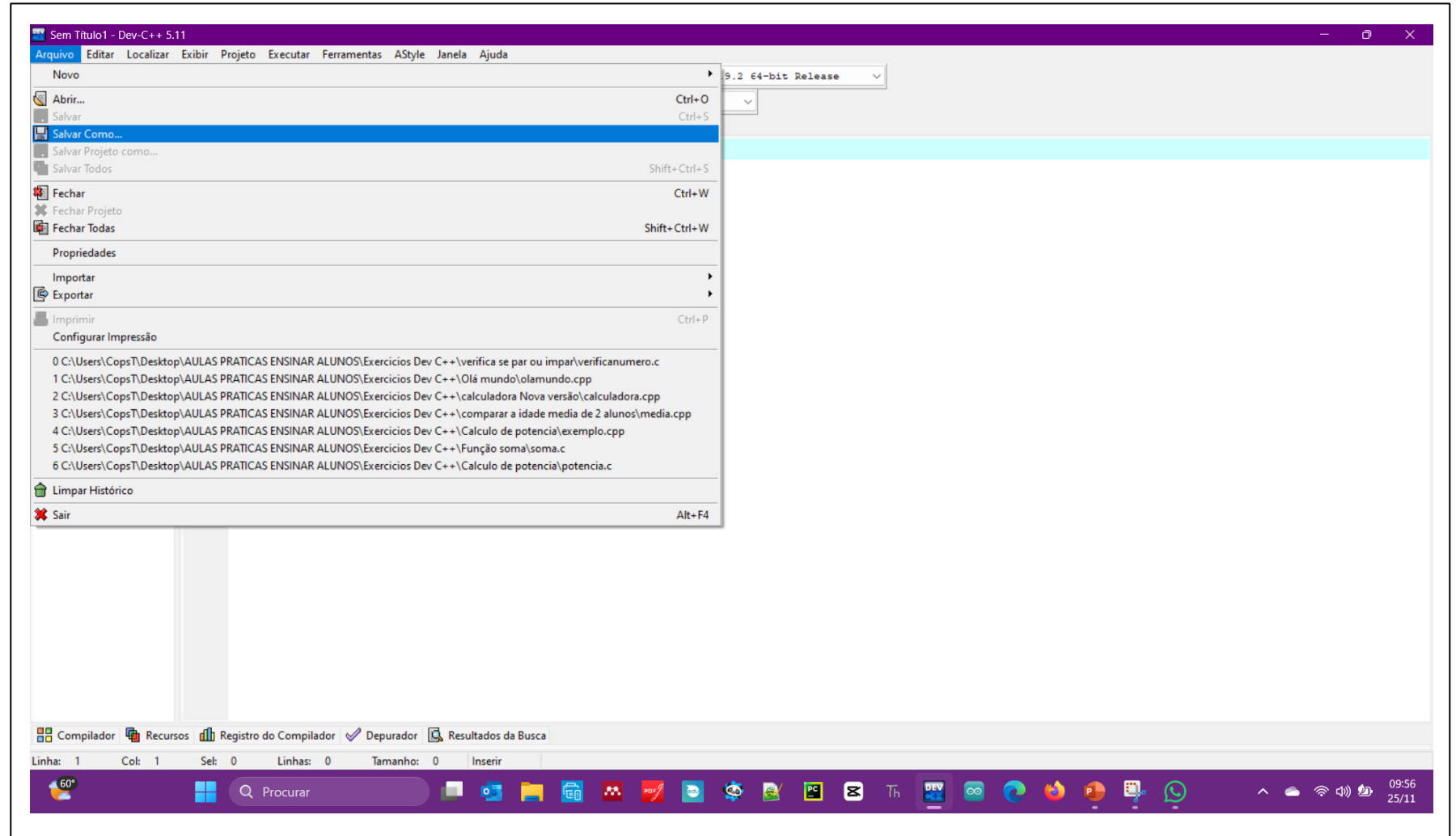
Passo 1 – Abrir o programa Dev C++



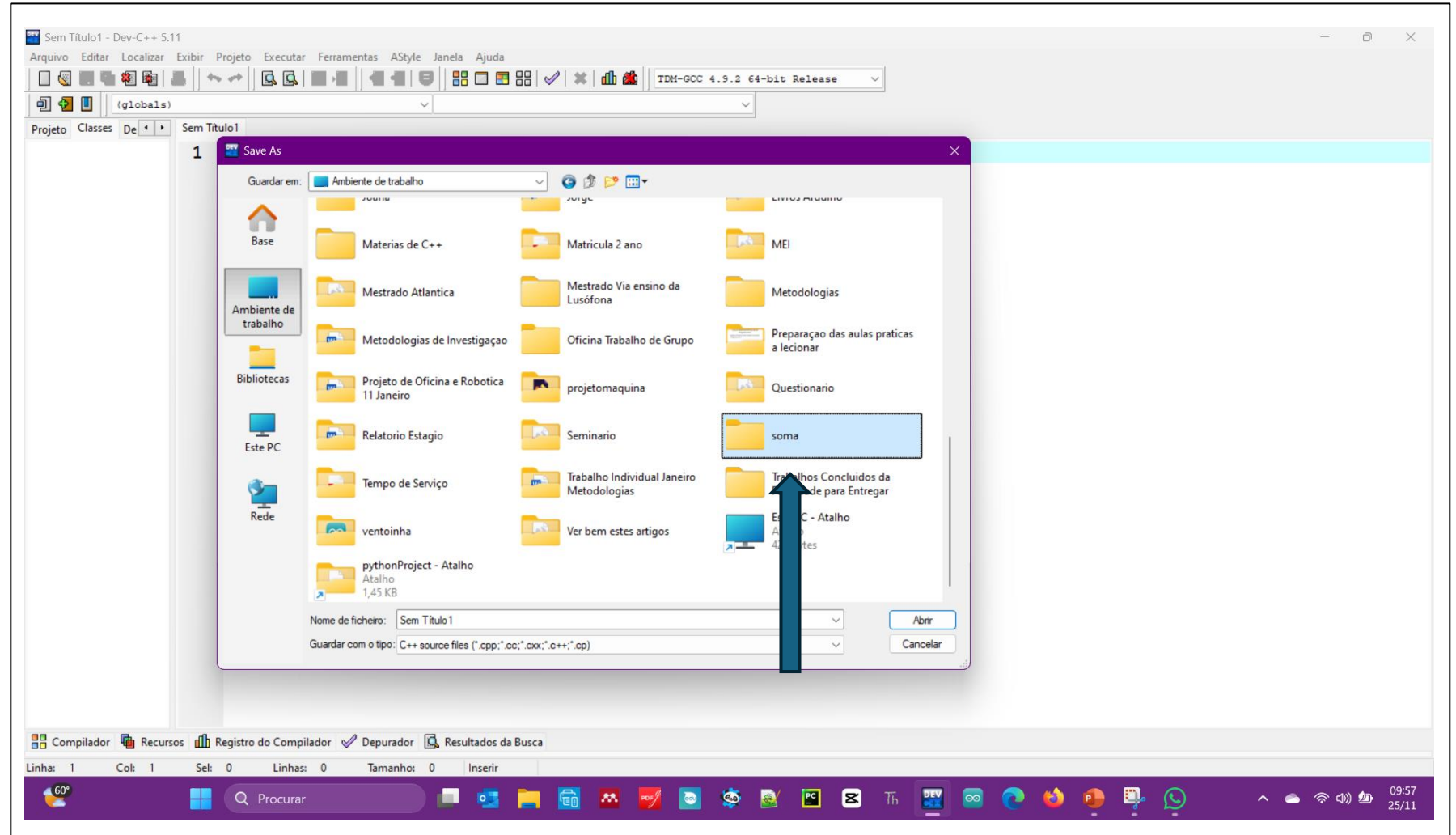
Passo 2 – Vamos a arquivo – Novo – Arquivo Fonte



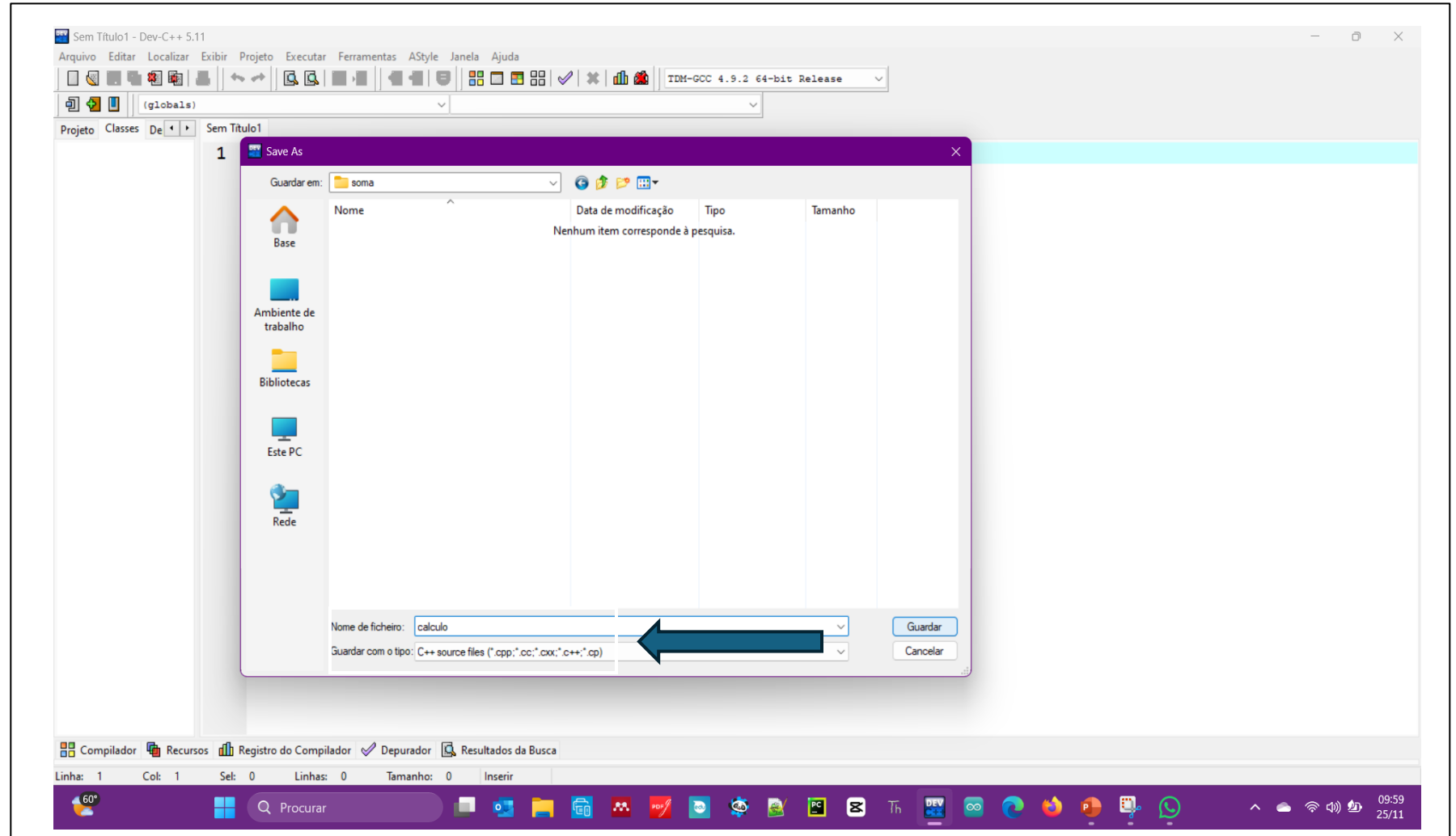
Passo 3 – Vamos agora salvar o ficheiro e vamos a salvar como...



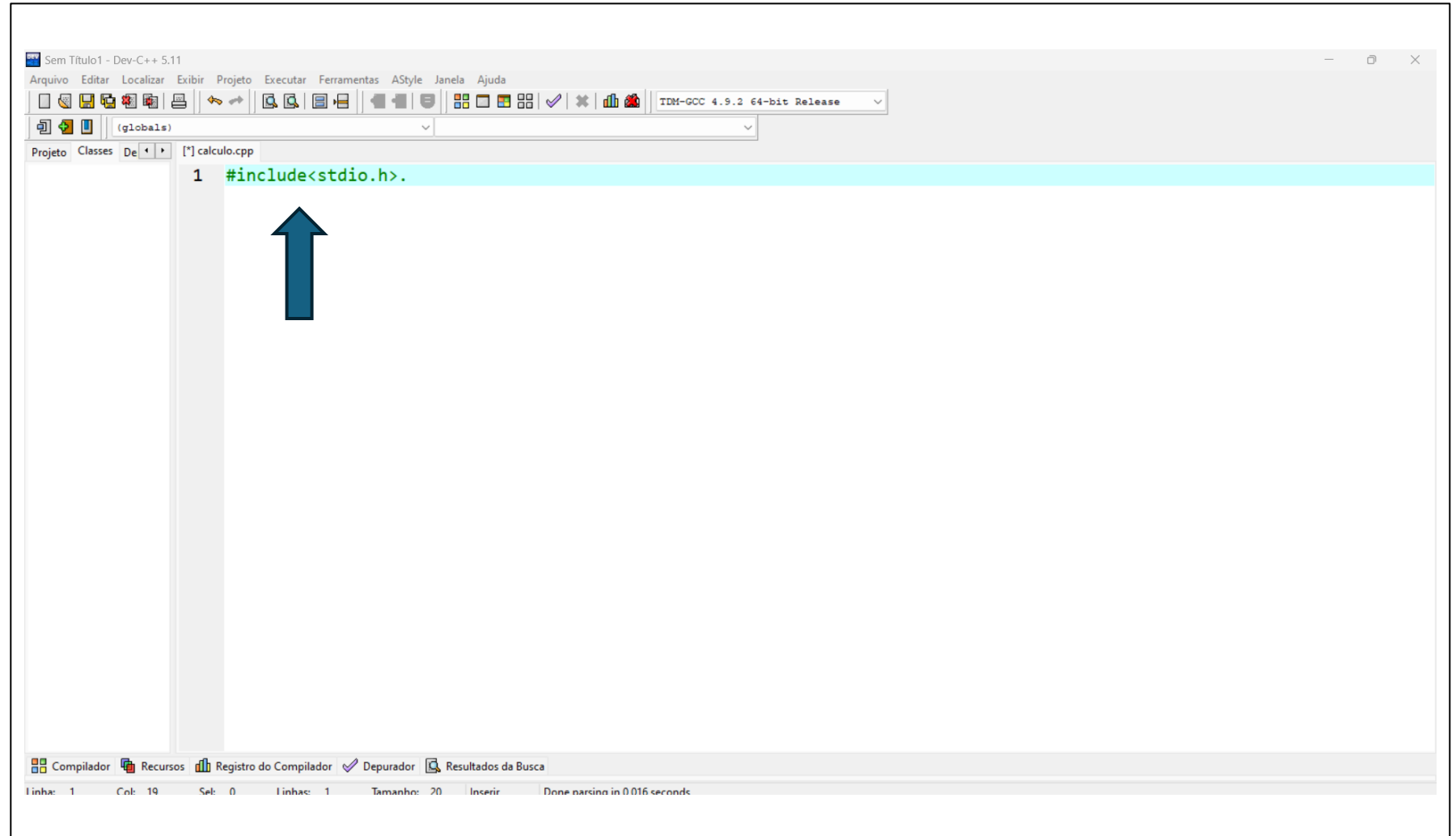
Passo 4 – Vamos agora salvar o ficheiro numa pasta chamada “soma”



Passo 5 – Vamos agora salvar o ficheiro dentro da pasta soma e dão lhe o nome “calculo”



Passo 6 – No topo do programa do Dev C++ linha 1 devemos incluir a biblioteca - `#include<stdio.h>`

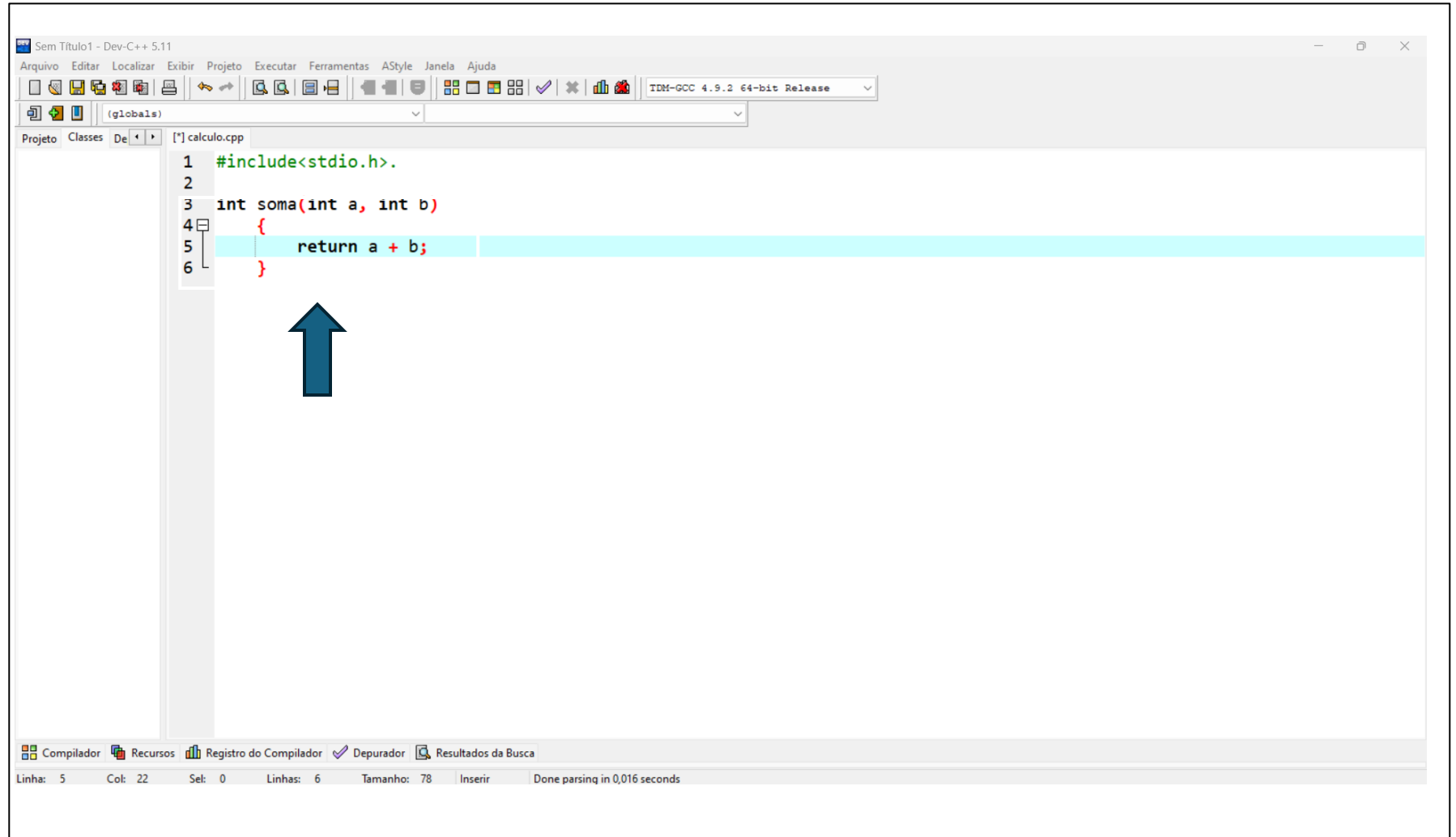


The screenshot shows the Dev-C++ IDE interface. The main editor window displays the file `calculo.cpp` with the following code on line 1:

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

A blue arrow points upwards to the code line. The status bar at the bottom indicates the file is 1 line long, 19 columns wide, and contains 20 characters. The compilation status shows "Done parsing in 0.016 seconds".

Passo 7 – Vamos declarar a função `int soma(int a, int b)` antes da função `main()`

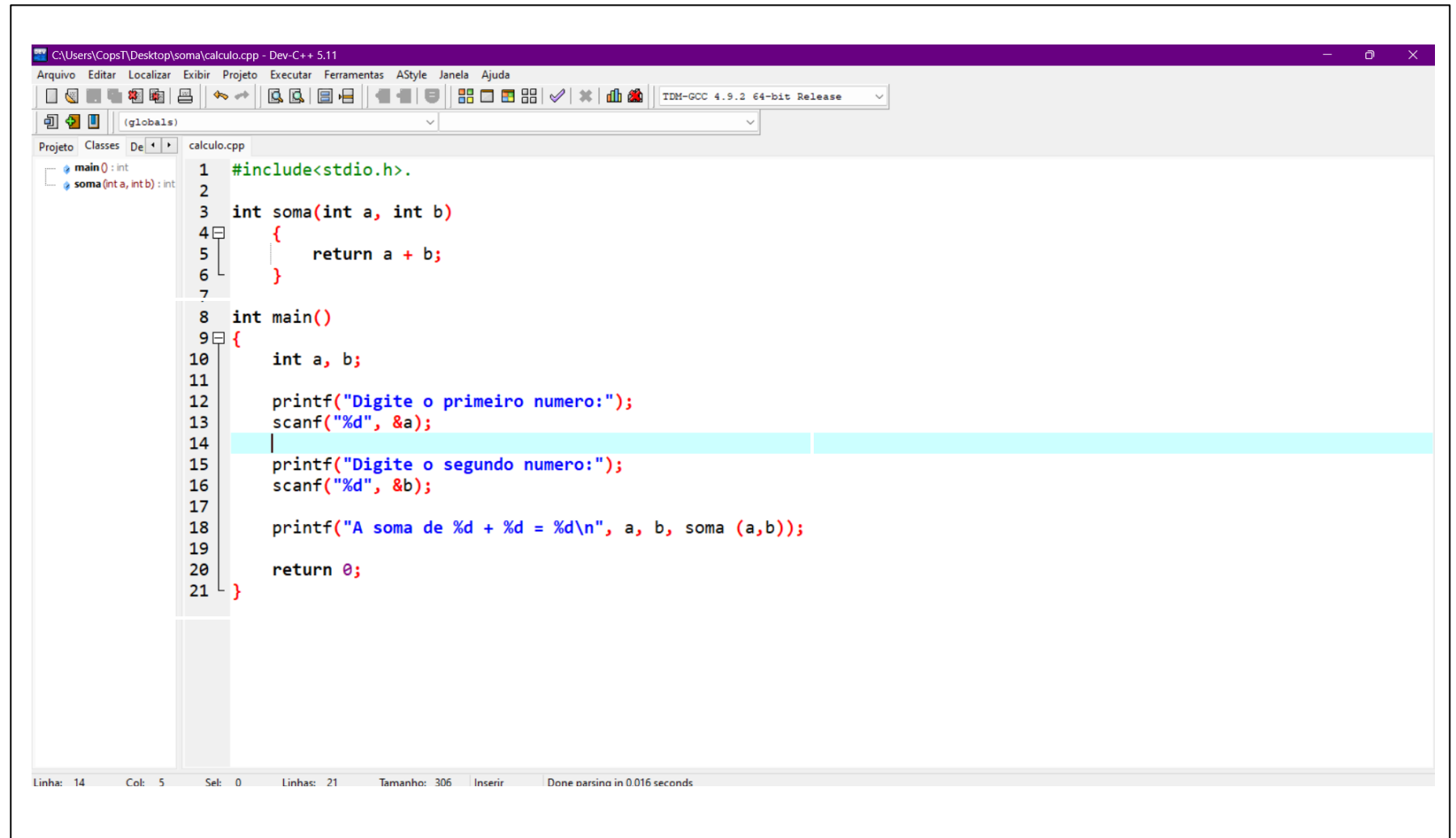


The screenshot shows the Dev-C++ IDE with a C++ file named `calculo.cpp`. The code is as follows:

```
1 #include<stdio.h>.  
2  
3 int soma(int a, int b)  
4 {  
5     return a + b;  
6 }
```

A blue arrow points to the function declaration on line 3. The IDE interface includes a menu bar (Arquivo, Editar, Localizar, Exibir, Projeto, Executar, Ferramentas, AStyle, Janela, Ajuda), a toolbar, a compiler selection dropdown (TDM-GCC 4.9.2 64-bit Release), and a status bar at the bottom showing line and column information (Linha: 5, Col: 22, Sel: 0, Linhas: 6, Tamanho: 78).

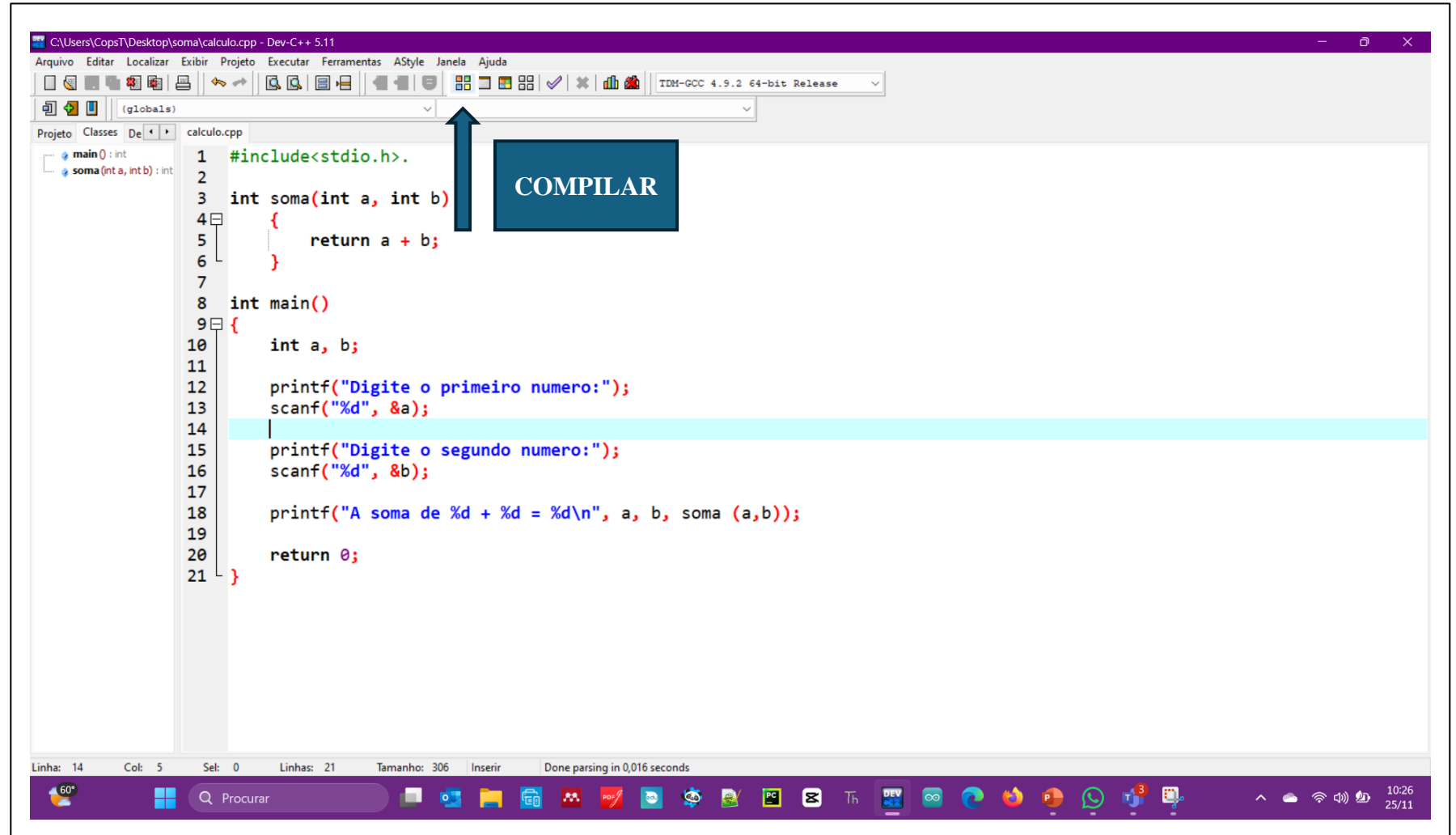
Passo 8 – Vamos implementar a função soma na função main()



```
C:\Users\CopsT\Desktop\soma\calculo.cpp - Dev-C++ 5.11
Arquivo Editar Localizar Exibir Projeto Executar Ferramentas AStyle Janela Ajuda
(globals)
Projeto Classes De calculo.cpp
main() : int
soma(int a, int b) : int
1 #include<stdio.h>.
2
3 int soma(int a, int b)
4 {
5     return a + b;
6 }
7
8 int main()
9 {
10     int a, b;
11
12     printf("Digite o primeiro numero:");
13     scanf("%d", &a);
14
15     printf("Digite o segundo numero:");
16     scanf("%d", &b);
17
18     printf("A soma de %d + %d = %d\n", a, b, soma (a,b));
19
20     return 0;
21 }
```

Linha: 14 Col: 5 Sel: 0 Linhas: 21 Tamanho: 306 Inserir Done parsing in 0.016 seconds

Passo 9 – Agora vamos compilar o código para verificarmos se existem erros

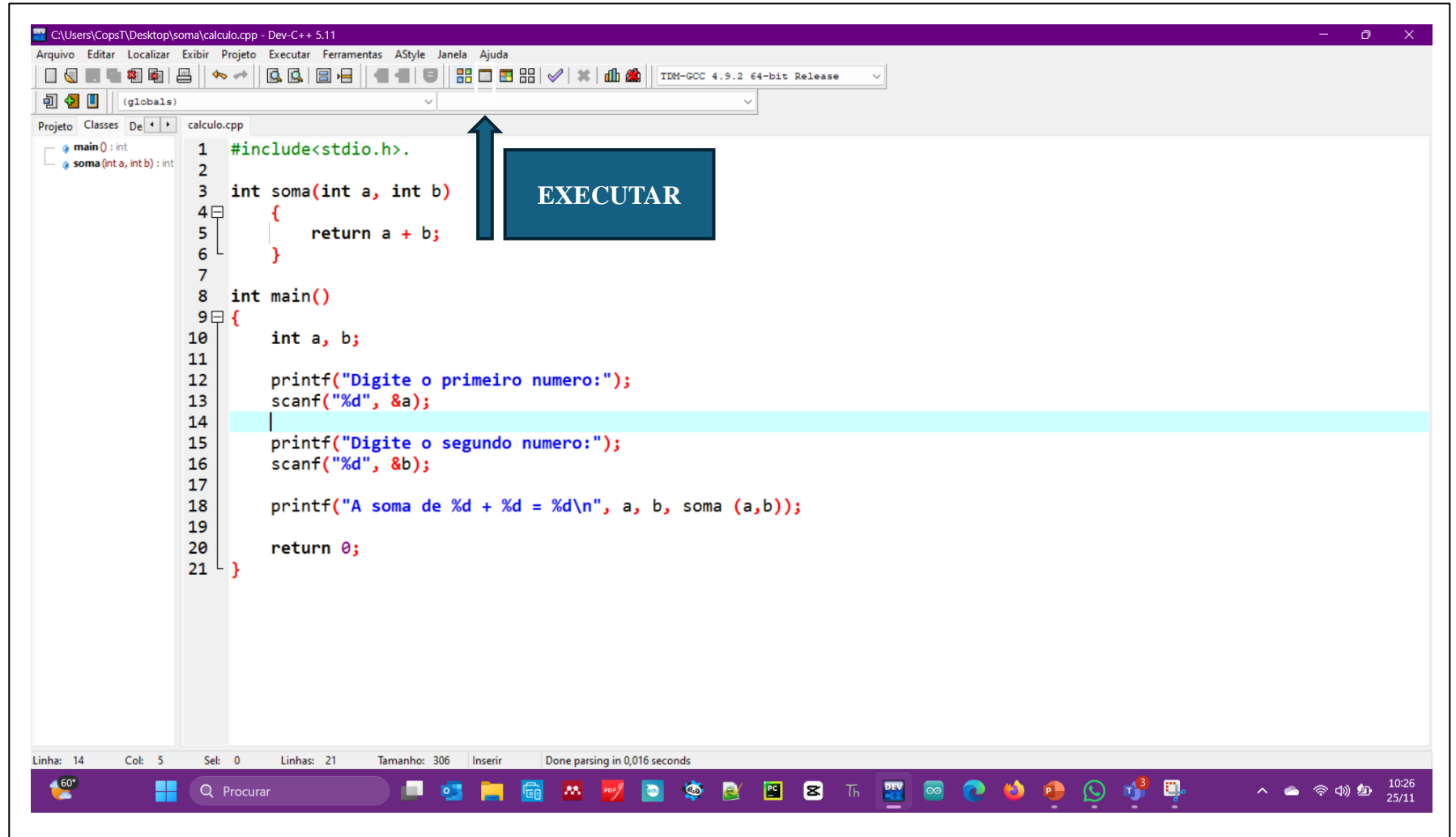


The screenshot shows the Dev-C++ IDE with a C++ program named `calculo.cpp`. The code is as follows:

```
1 #include<stdio.h>.
2
3 int soma(int a, int b)
4 {
5     return a + b;
6 }
7
8 int main()
9 {
10     int a, b;
11
12     printf("Digite o primeiro numero:");
13     scanf("%d", &a);
14
15     printf("Digite o segundo numero:");
16     scanf("%d", &b);
17
18     printf("A soma de %d + %d = %d\n", a, b, soma (a,b));
19
20     return 0;
21 }
```

The IDE interface includes a menu bar (Arquivo, Editar, Localizar, Exibir, Projeto, Executar, Ferramentas, AStyle, Janela, Ajuda), a toolbar with various icons, and a status bar at the bottom showing "Linha: 14 Col: 5 Sel: 0 Linhas: 21 Tamanho: 306 Inserir Done parsing in 0,016 seconds". A Windows taskbar is visible at the bottom of the screen.

Passo 10 – Agora vamos executar o nosso código

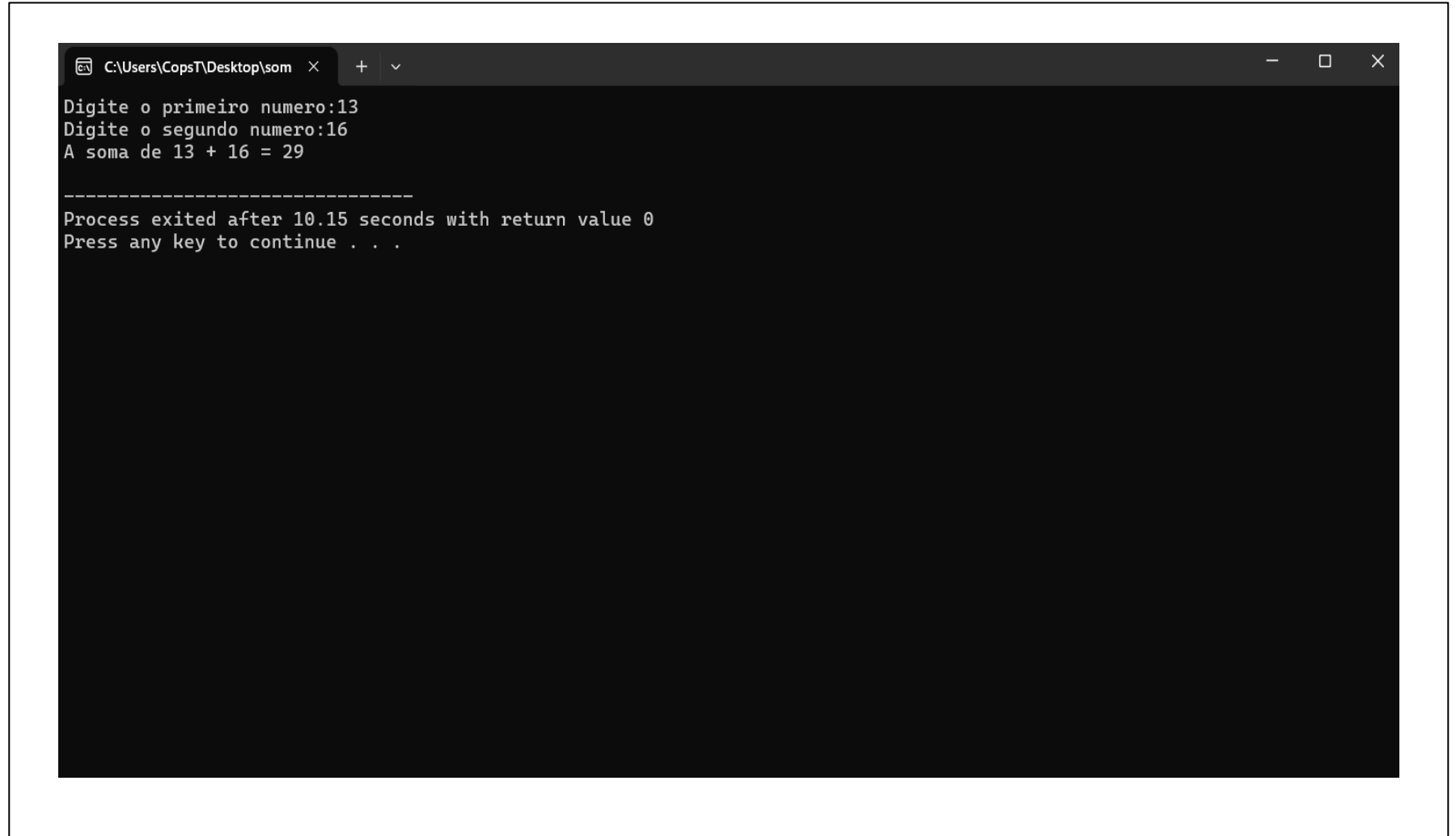


```
1 #include<stdio.h>.
2
3 int soma(int a, int b)
4 {
5     return a + b;
6 }
7
8 int main()
9 {
10     int a, b;
11
12     printf("Digite o primeiro numero:");
13     scanf("%d", &a);
14
15     printf("Digite o segundo numero:");
16     scanf("%d", &b);
17
18     printf("A soma de %d + %d = %d\n", a, b, soma (a,b));
19
20     return 0;
21 }
```

EXECUTAR

Linha: 14 Col: 5 Sel: 0 Linhas: 21 Tamanho: 306 Inserir Done parsing in 0,016 seconds

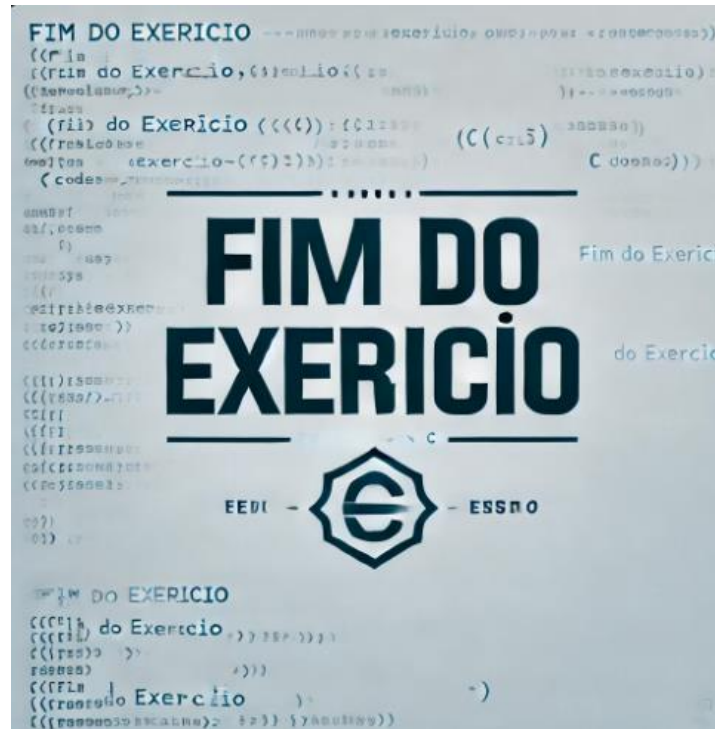
Passo 11 – Programa a correr na consola



```
C:\Users\CopsT\Desktop\som x + v
Digite o primeiro numero:13
Digite o segundo numero:16
A soma de 13 + 16 = 29

-----
Process exited after 10.15 seconds with return value 0
Press any key to continue . . .
```

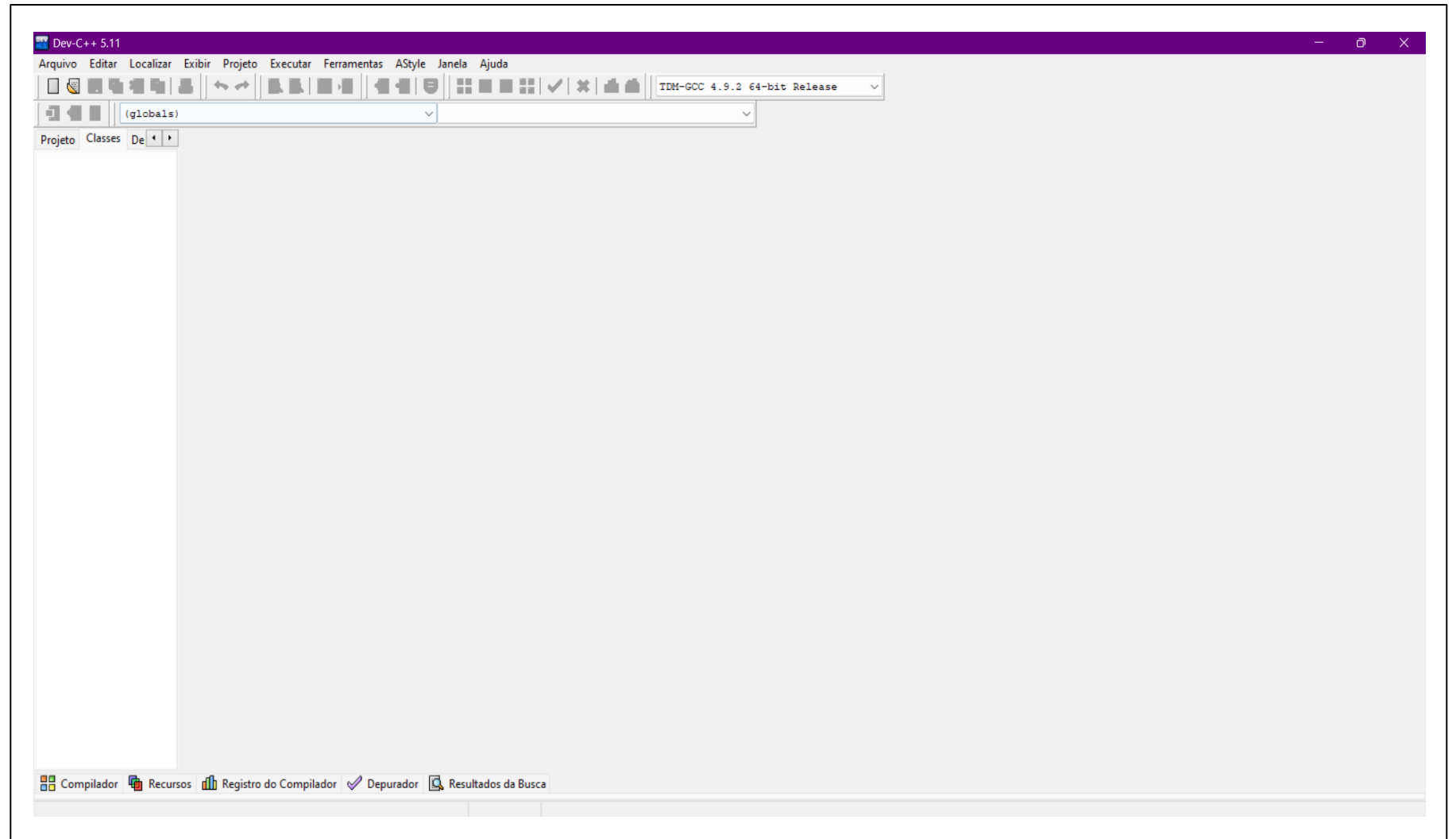
FIM DO EXERCICIO 1 CONDUZIDO STEP BY STEP



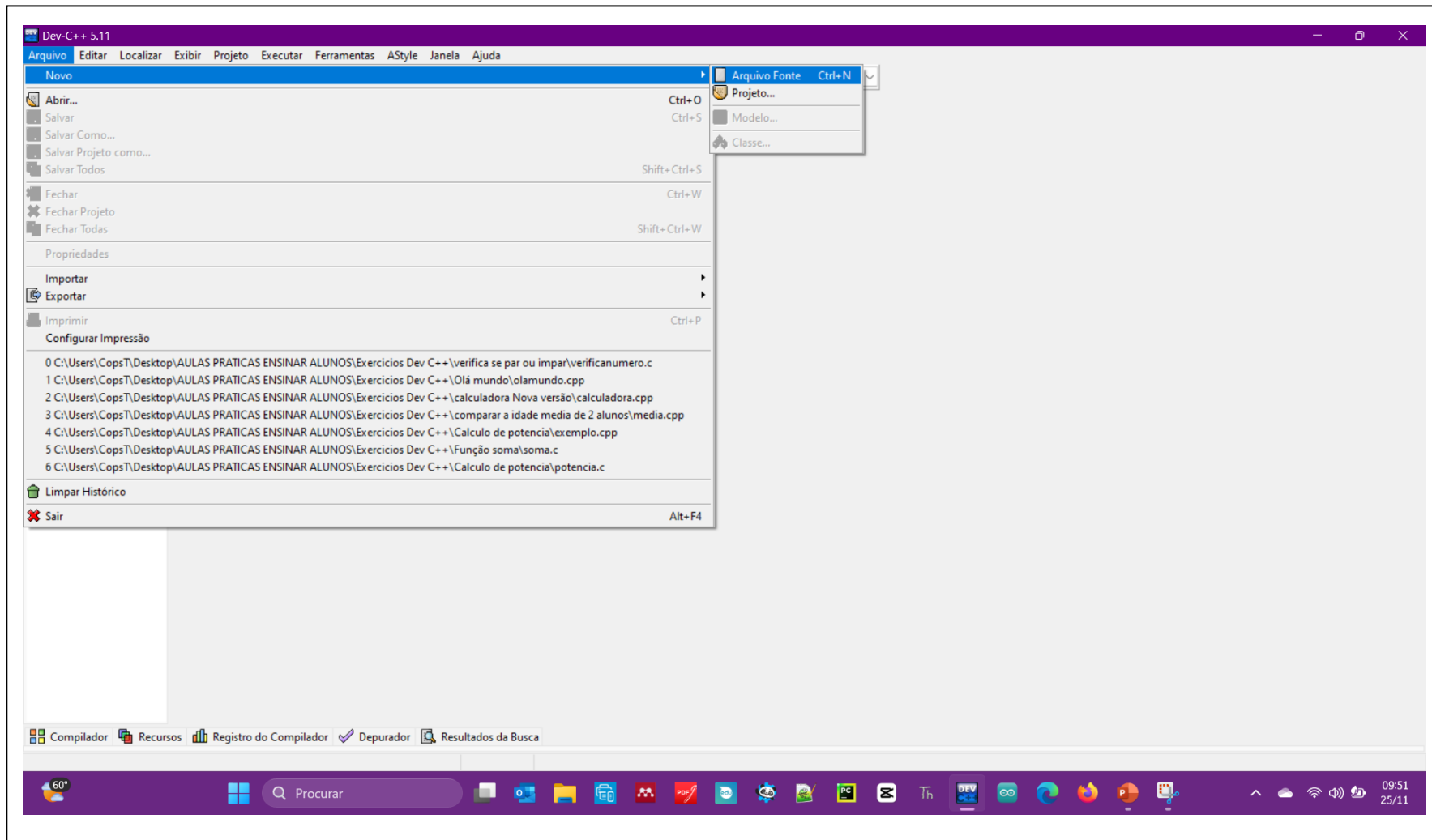
INICIO DO EXERCICIO 2 CONDUZIDO STEP BY STEP

- Desenvolva um programa que calcule a média de três notas de um aluno. Crie uma função com o nome `calculaMedia()` e que retorne o valor da média. O programa deve exibir a média no ecrã

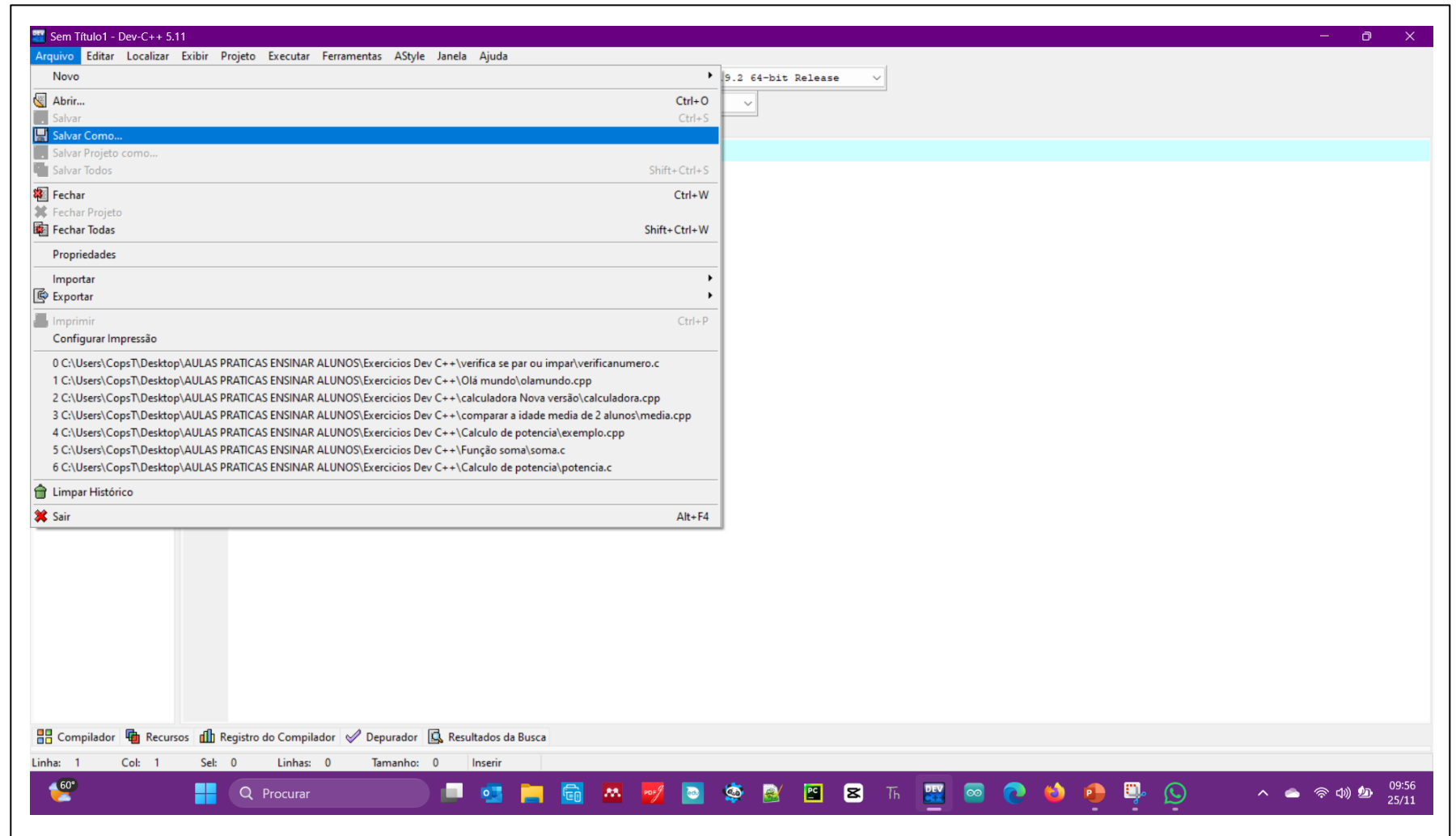
Passo 1 – Abrir o program Dev C++



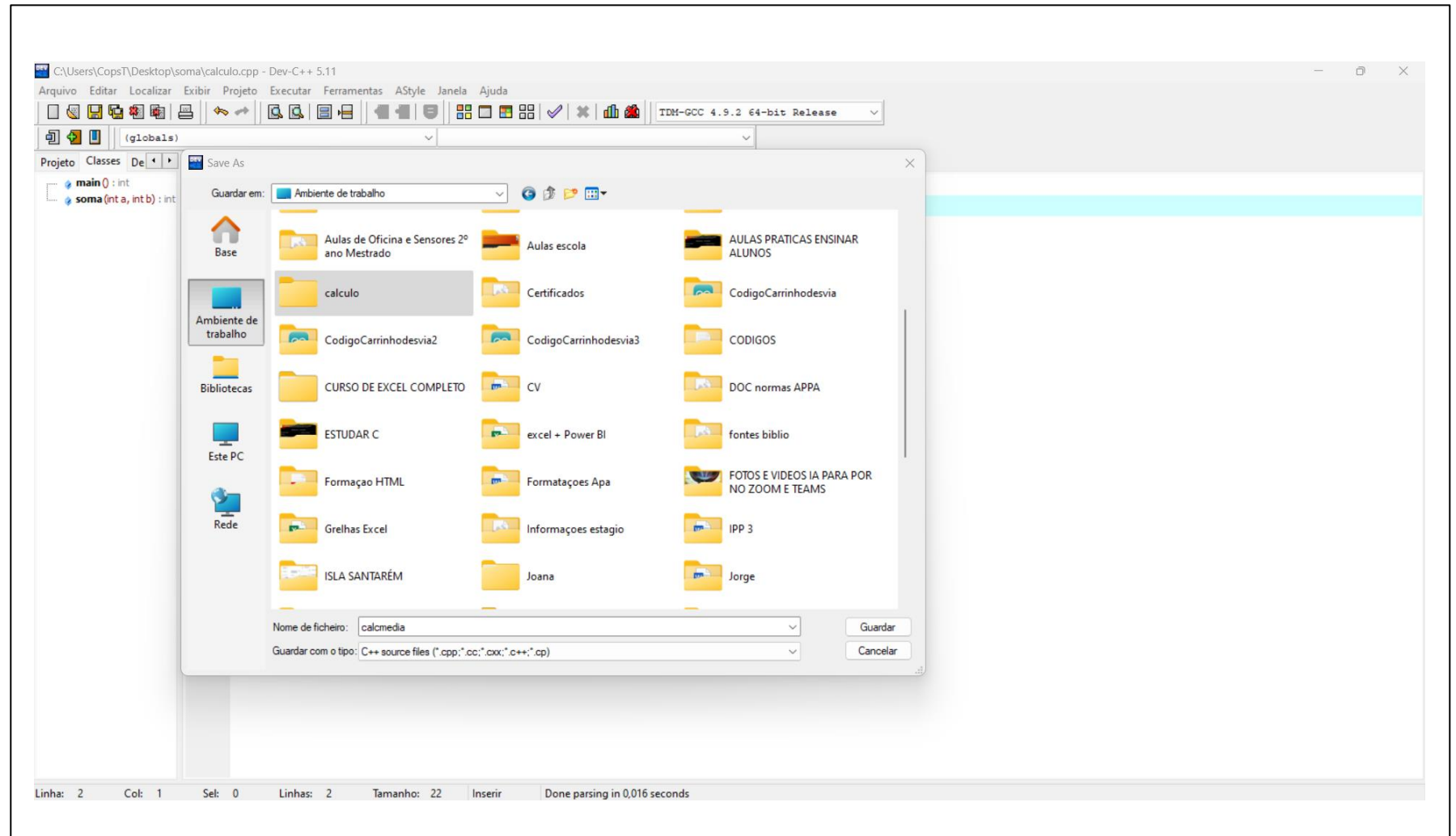
Passo 2 – Vamos a arquivo – Novo – Arquivo Fonte



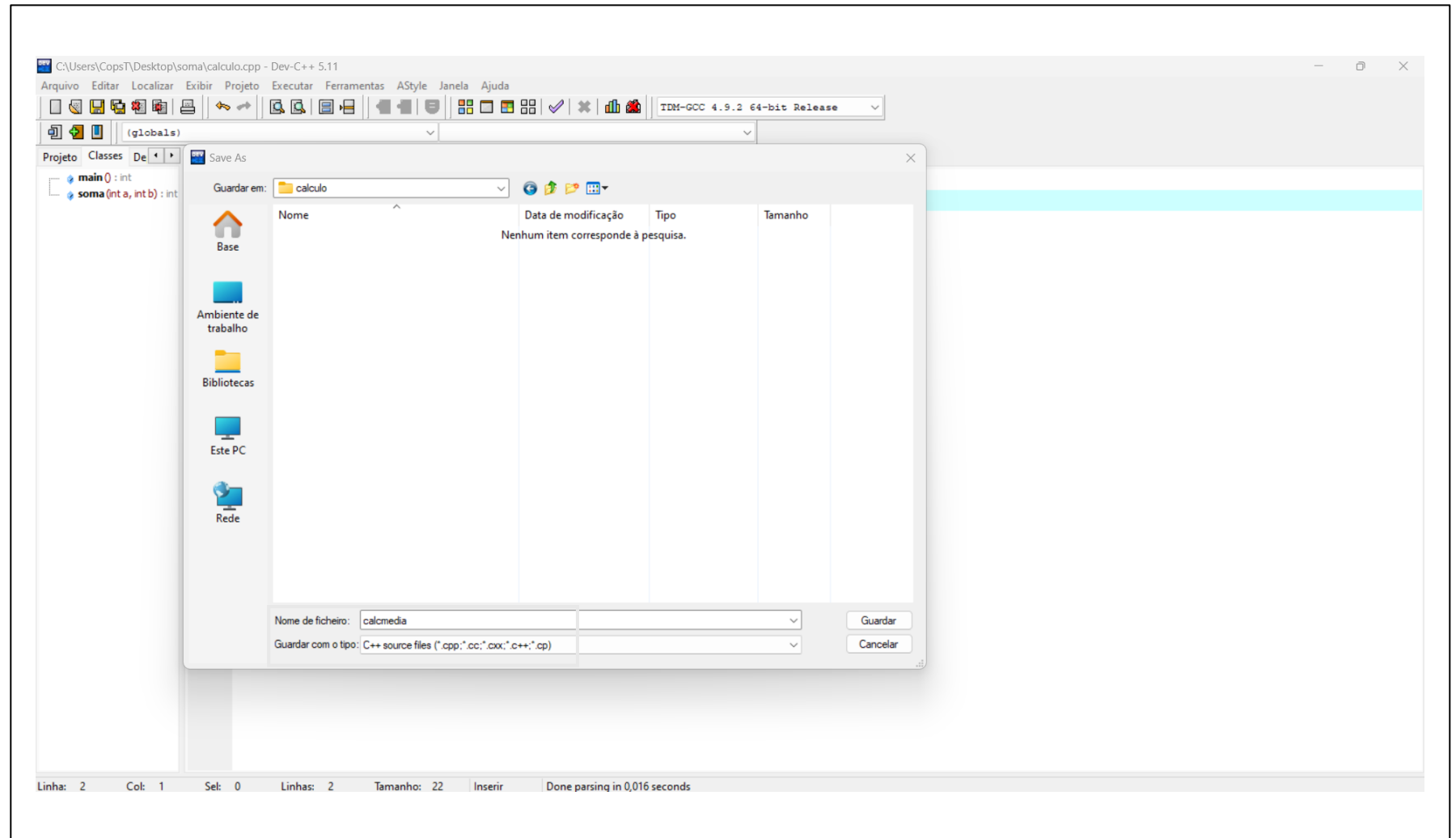
Passo 3 – Vamos agora salvar o ficheiro em salvar como...



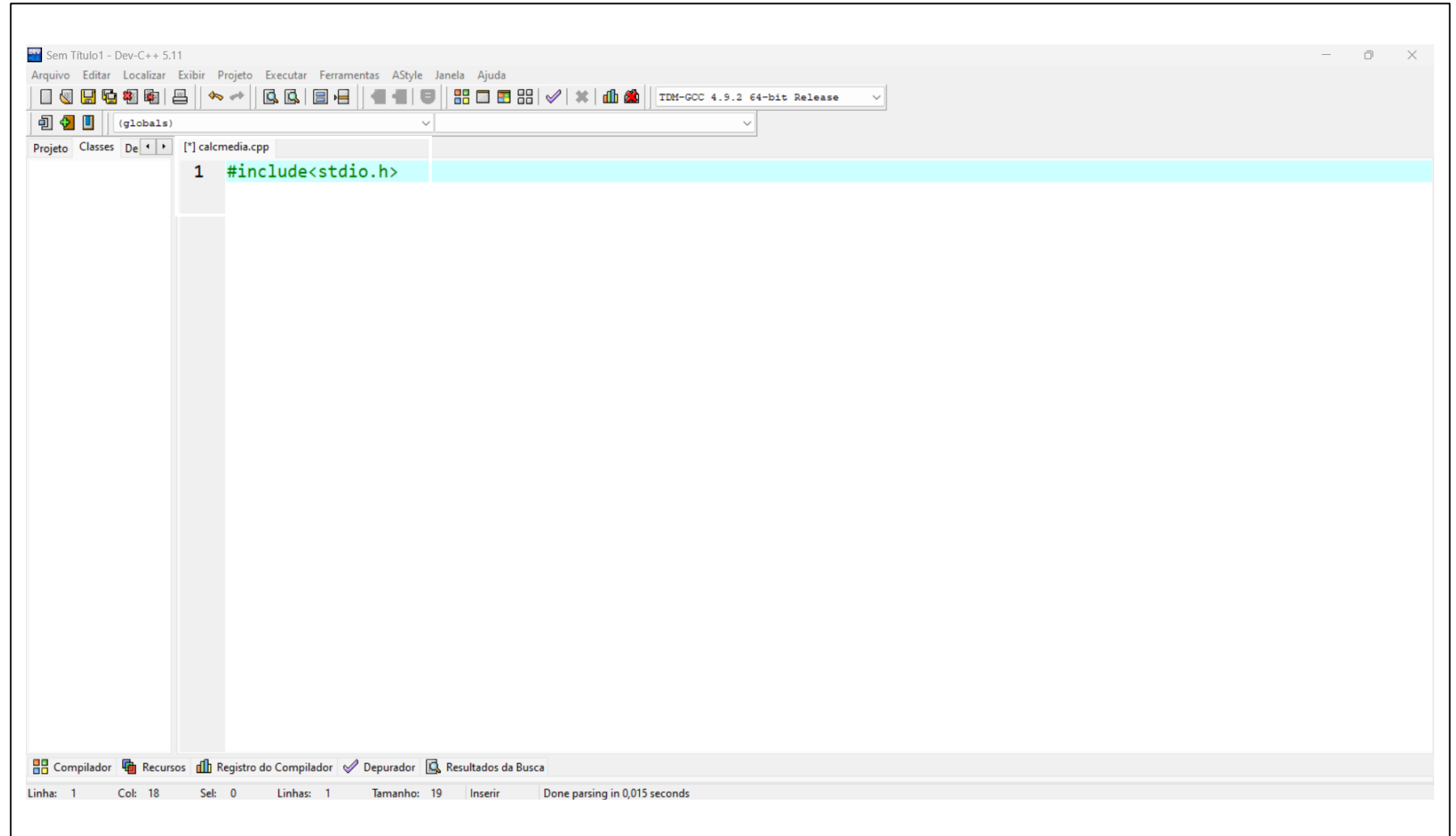
Passo 4 – Vamos agora salvar o ficheiro numa pasta chamada “calcmedia”



Passo 5 – Vamos guardar o ficheiro com o nome “calcmedia”

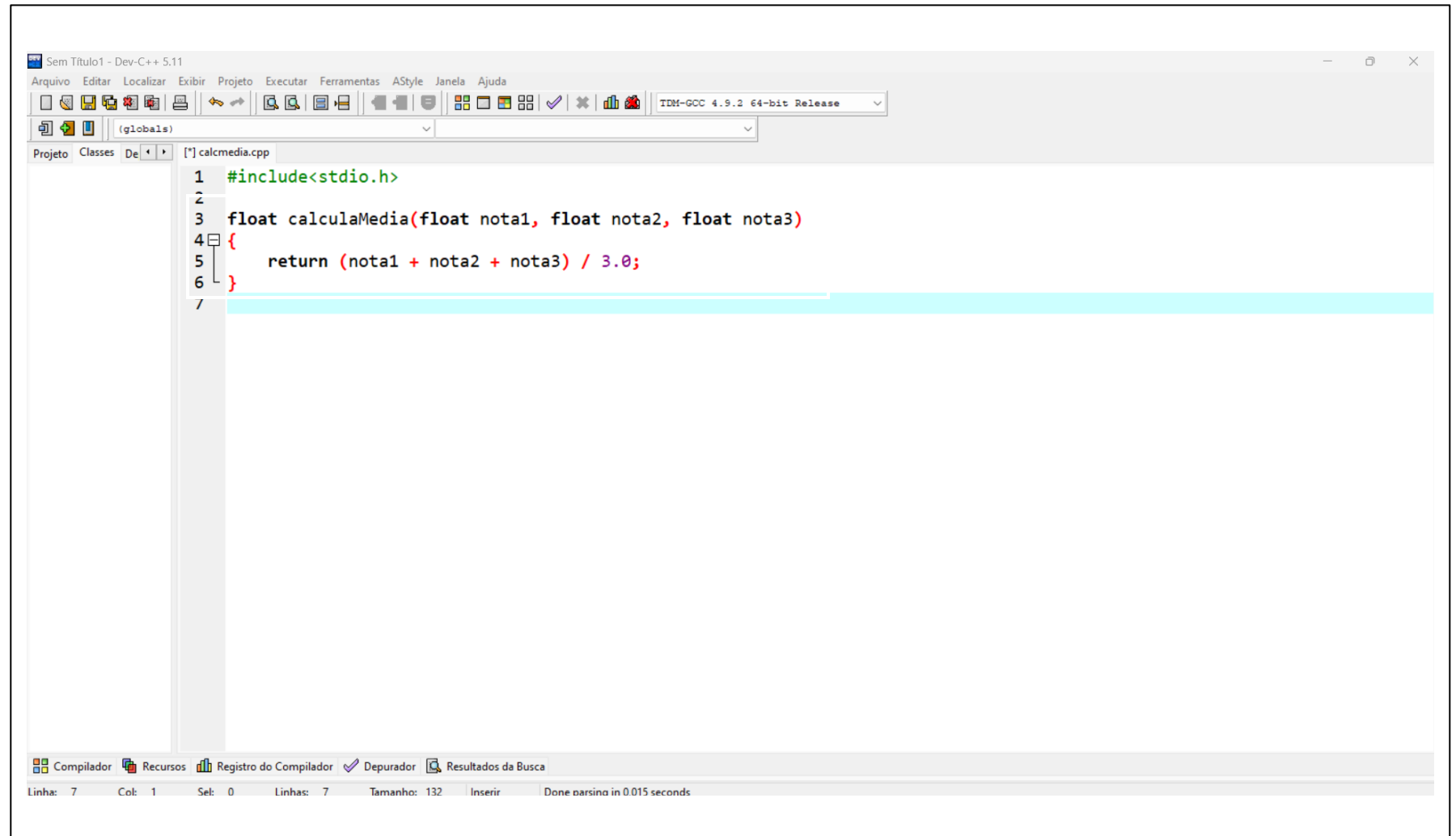


Passo 6 – No topo do programa do Dev C++ linha 1 devemos incluir a biblioteca - `#include<stdio.h>`



The screenshot shows the Dev-C++ IDE interface. The main window displays a C++ source file named 'calcmidia.cpp'. The first line of the code is highlighted in light blue and reads: `1 #include<stdio.h>`. The IDE's menu bar includes 'Arquivo', 'Editar', 'Localizar', 'Exibir', 'Projeto', 'Executar', 'Ferramentas', 'AStyle', 'Janela', and 'Ajuda'. The toolbar contains various icons for file operations and compilation. The status bar at the bottom shows 'Linha: 1 Col: 18 Sel: 0 Linhas: 1 Tamanho: 19 Inserir Done parsing in 0,015 seconds'.

Passo 7 – Vamos declarar a função float calculaMedia(float nota1, float nota2, float nota3) antes da função main()

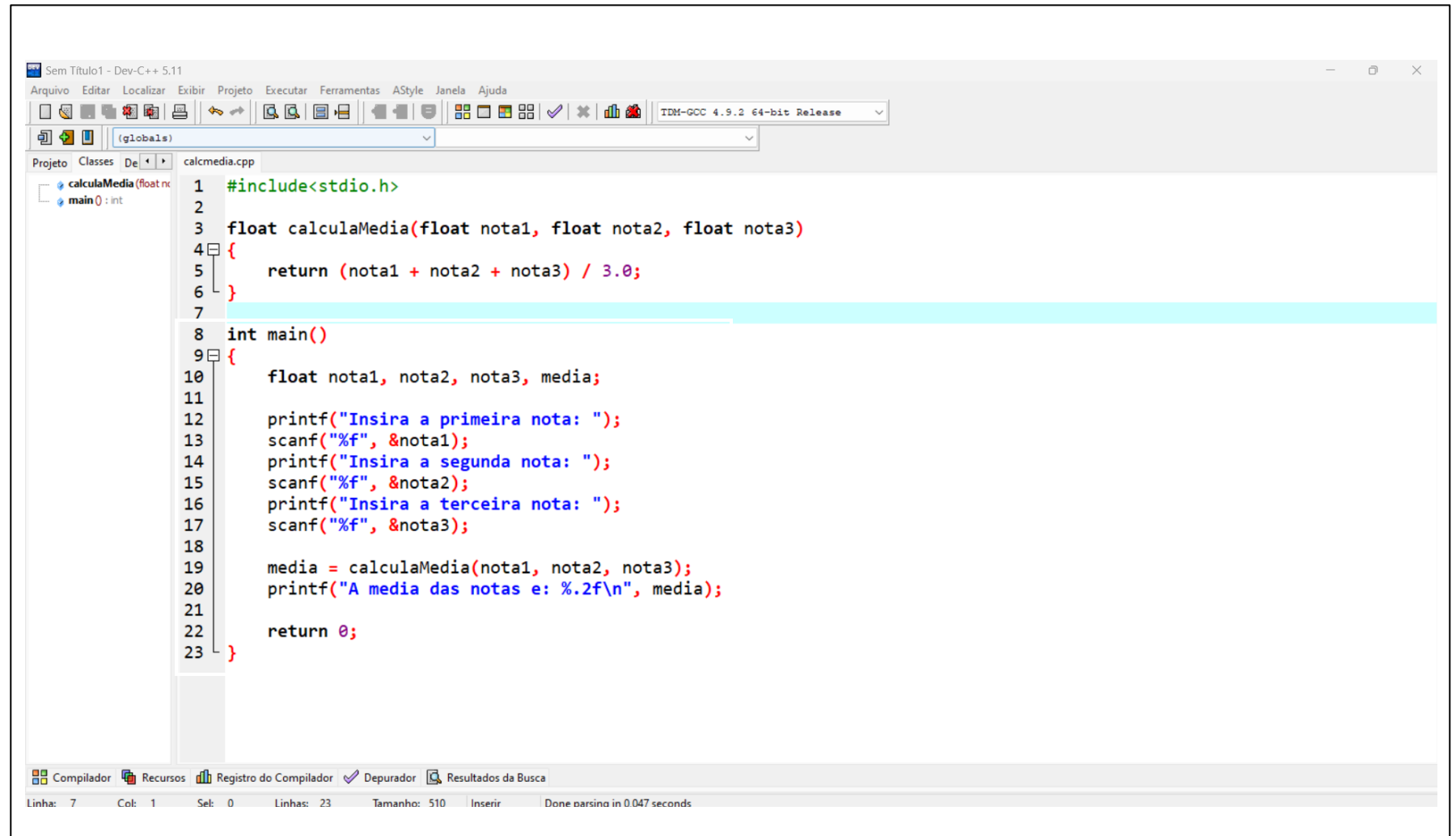


The screenshot shows the Dev-C++ IDE interface. The main editor window displays the following C++ code in a file named `calcmidia.cpp`:

```
1 #include<stdio.h>
2
3 float calculaMedia(float nota1, float nota2, float nota3)
4 {
5     return (nota1 + nota2 + nota3) / 3.0;
6 }
7
```

The code is color-coded: `#include` is green, `float` is blue, `calculaMedia` is black, `float` is blue, `nota1`, `nota2`, and `nota3` are black, `{` and `}` are black, `return` is black, `(`, `+`, `+`, `+`, `)` are black, `/` is black, `3.0` is black, and `;` is black. The IDE's status bar at the bottom indicates: Linha: 7 Col: 1 Sel: 0 Linhas: 7 Tamanho: 132 Inserir Done parsing in 0.015 seconds.

Passo 8 – Vamos agora escrever o resto do programa utilizando a função main()

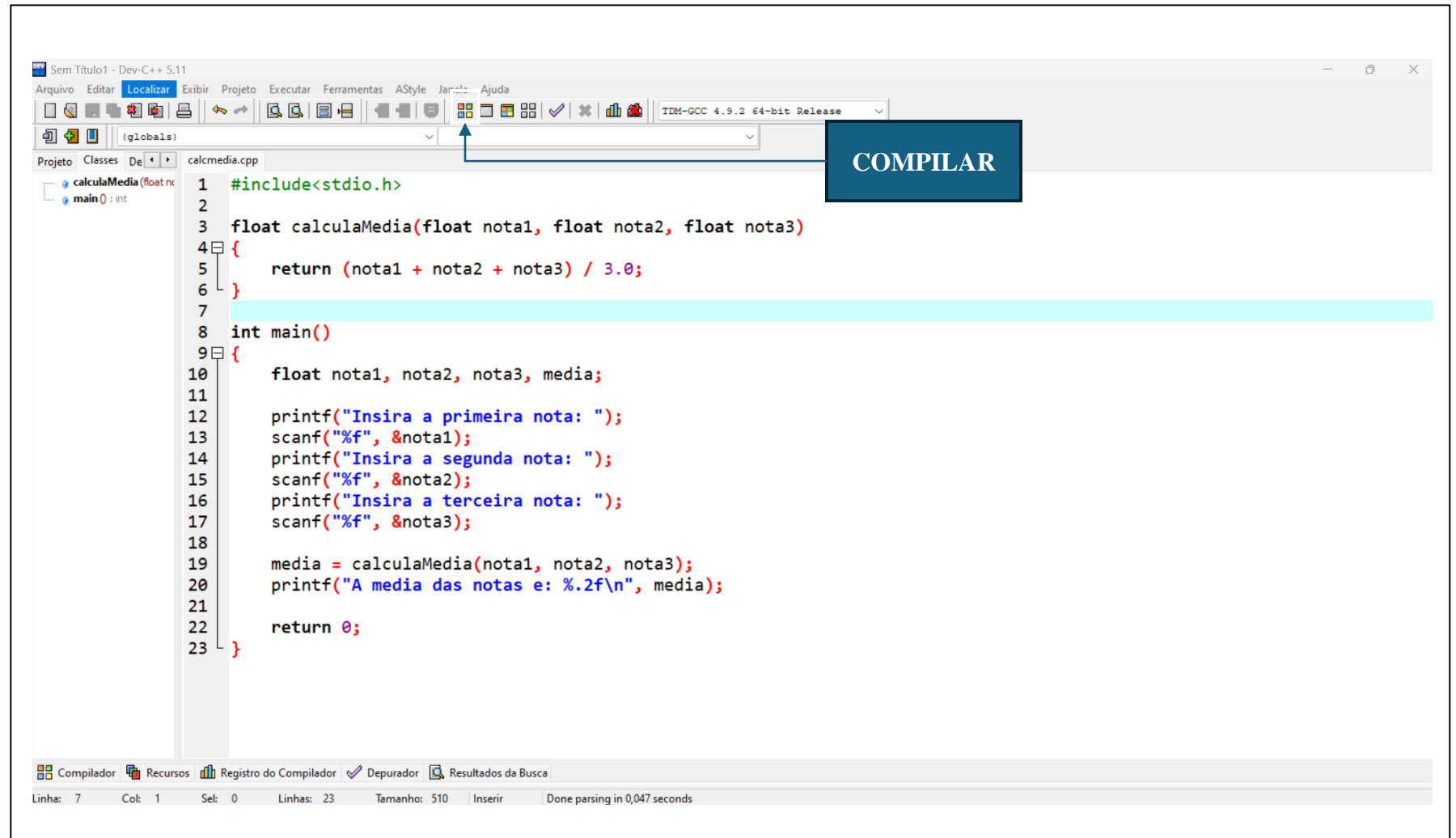


```
1 #include<stdio.h>
2
3 float calculaMedia(float nota1, float nota2, float nota3)
4 {
5     return (nota1 + nota2 + nota3) / 3.0;
6 }
7
8 int main()
9 {
10     float nota1, nota2, nota3, media;
11
12     printf("Insira a primeira nota: ");
13     scanf("%f", &nota1);
14     printf("Insira a segunda nota: ");
15     scanf("%f", &nota2);
16     printf("Insira a terceira nota: ");
17     scanf("%f", &nota3);
18
19     media = calculaMedia(nota1, nota2, nota3);
20     printf("A media das notas e: %.2f\n", media);
21
22     return 0;
23 }
```

Compilador Recursos Registro do Compilador Depurador Resultados da Busca

Linha: 7 Col: 1 Sel: 0 Linhas: 23 Tamanho: 510 Inserir Done parsing in 0.047 seconds

Passo 9 – Vamos agora compilar o programa para verificarmos se existem erros no código



The image shows a screenshot of the Dev-C++ IDE. The main window displays a C++ program named `calcmidia.cpp`. The code defines a function `calculaMedia` that takes three floating-point numbers and returns their average, and a `main` function that prompts the user for three numbers, calculates the average, and prints the result. A blue box with the text "COMPILAR" is positioned over the toolbar, with an arrow pointing to the compile button (represented by a gear icon).

```
1 #include<stdio.h>
2
3 float calculaMedia(float nota1, float nota2, float nota3)
4 {
5     return (nota1 + nota2 + nota3) / 3.0;
6 }
7
8 int main()
9 {
10    float nota1, nota2, nota3, media;
11
12    printf("Insira a primeira nota: ");
13    scanf("%f", &nota1);
14    printf("Insira a segunda nota: ");
15    scanf("%f", &nota2);
16    printf("Insira a terceira nota: ");
17    scanf("%f", &nota3);
18
19    media = calculaMedia(nota1, nota2, nota3);
20    printf("A media das notas e: %.2f\n", media);
21
22    return 0;
23 }
```

At the bottom of the IDE, the status bar shows: Linha: 7 Col: 1 Sel: 0 Linhas: 23 Tamanho: 510 Inserir Done parsing in 0,047 seconds.

Passo 10 – Vamos agora executar o programa e ver o resultado

```
#include <stdio.h>

float calcularMedia(float a, float b, float c);

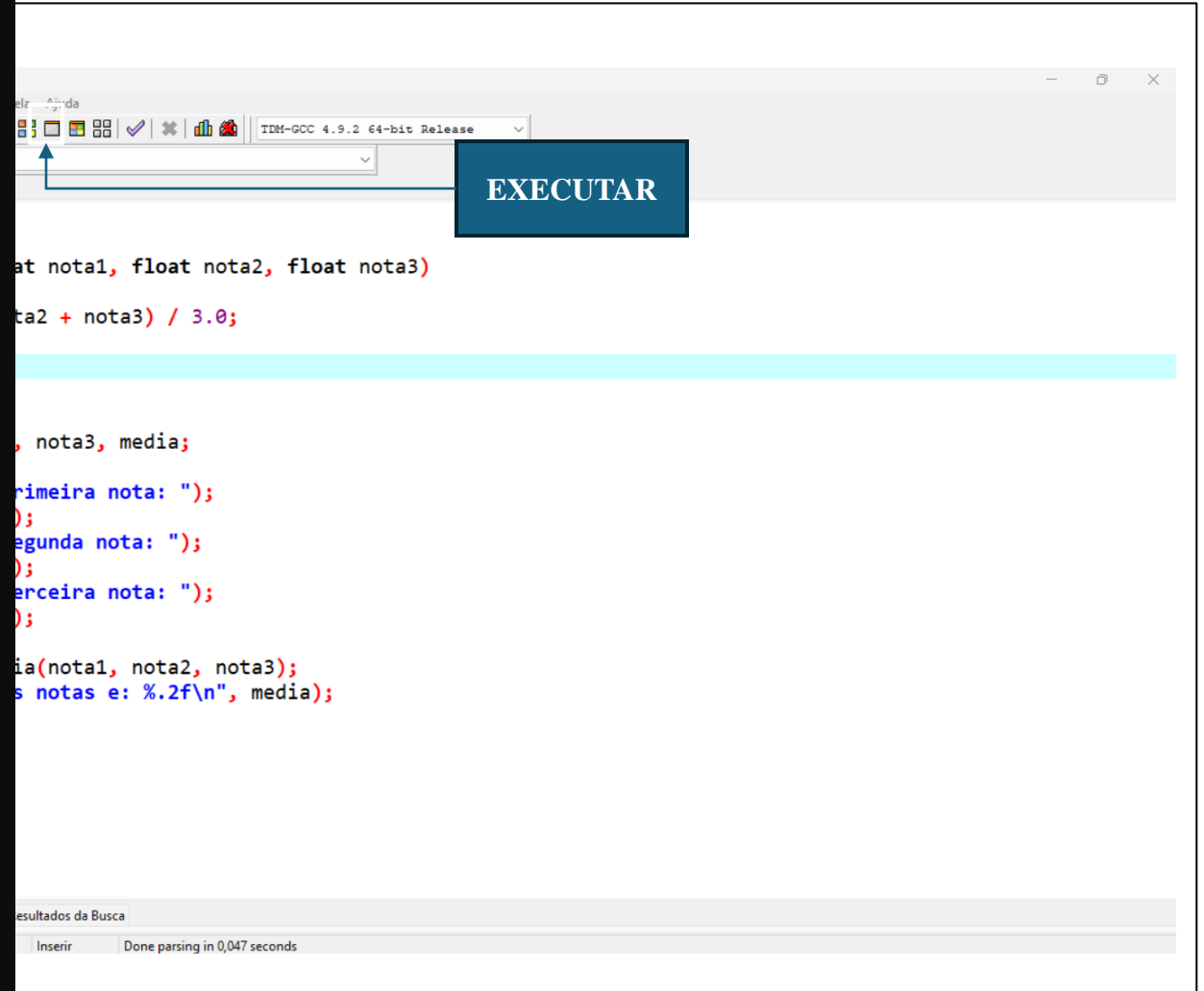
int main() {
    float nota1, nota2, nota3;

    printf("Digite as três notas separadas por espaço: ");
    scanf("%f %f %f", &nota1, &nota2, &nota3);

    printf("A média das notas é: %.2f\n", calcularMedia(nota1, nota2, nota3));

    return 0;
}

float calcularMedia(float a, float b, float c) {
    return (a + b + c) / 3;
}
```

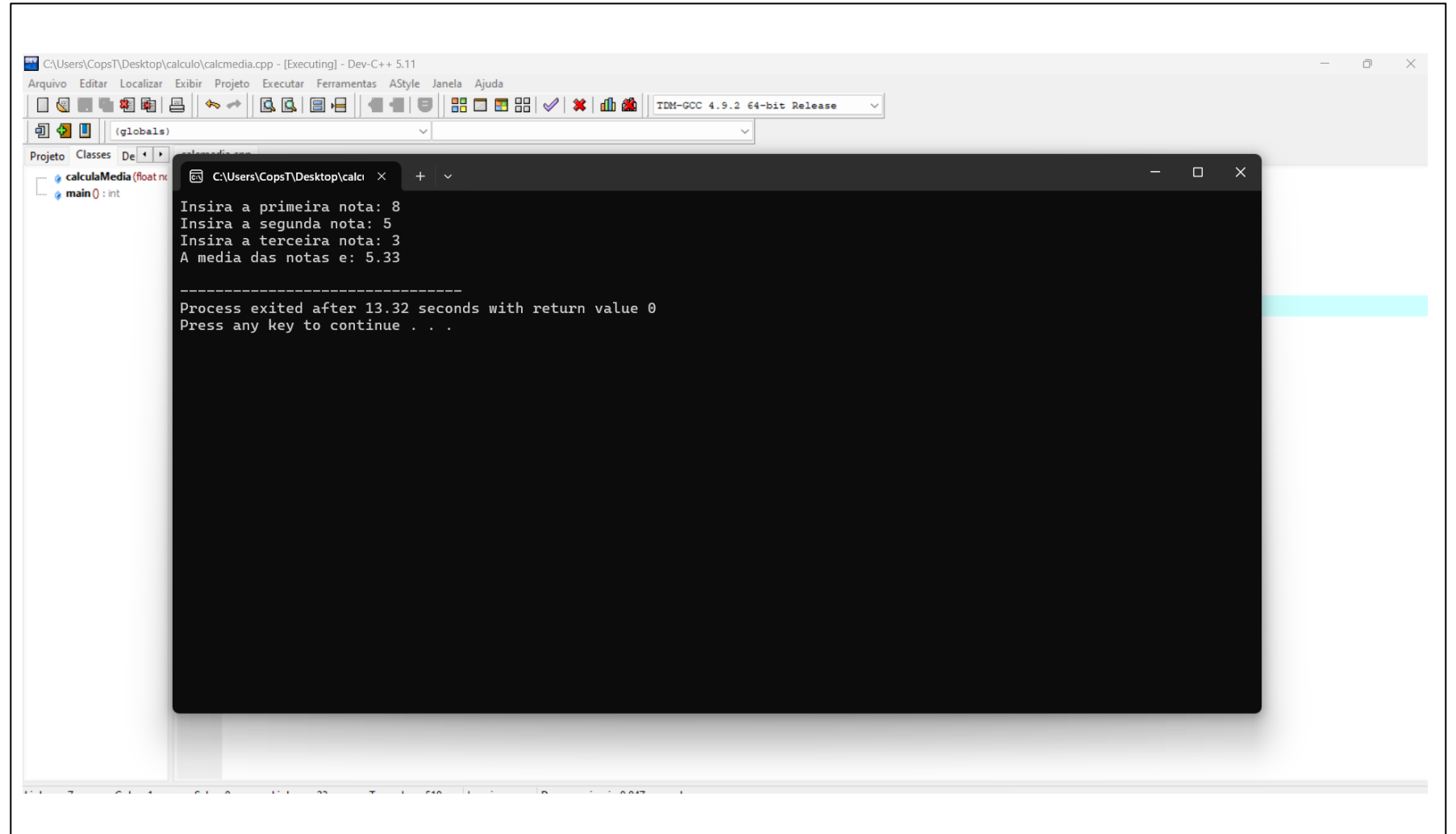


```
elr - Ajuda
IDM-GCC 4.9.2 64-bit Release
EXECUTAR
at nota1, float nota2, float nota3)
ta2 + nota3) / 3.0;

, nota3, media;
primeira nota: ");
);
segunda nota: ");
);
terceira nota: ");
);
ia(nota1, nota2, nota3);
s notas e: %.2f\n", media);

resultados da Busca
Inserir Done parsing in 0,047 seconds
```

Passo 10 – Vamos agora executar e ver a nossa consola



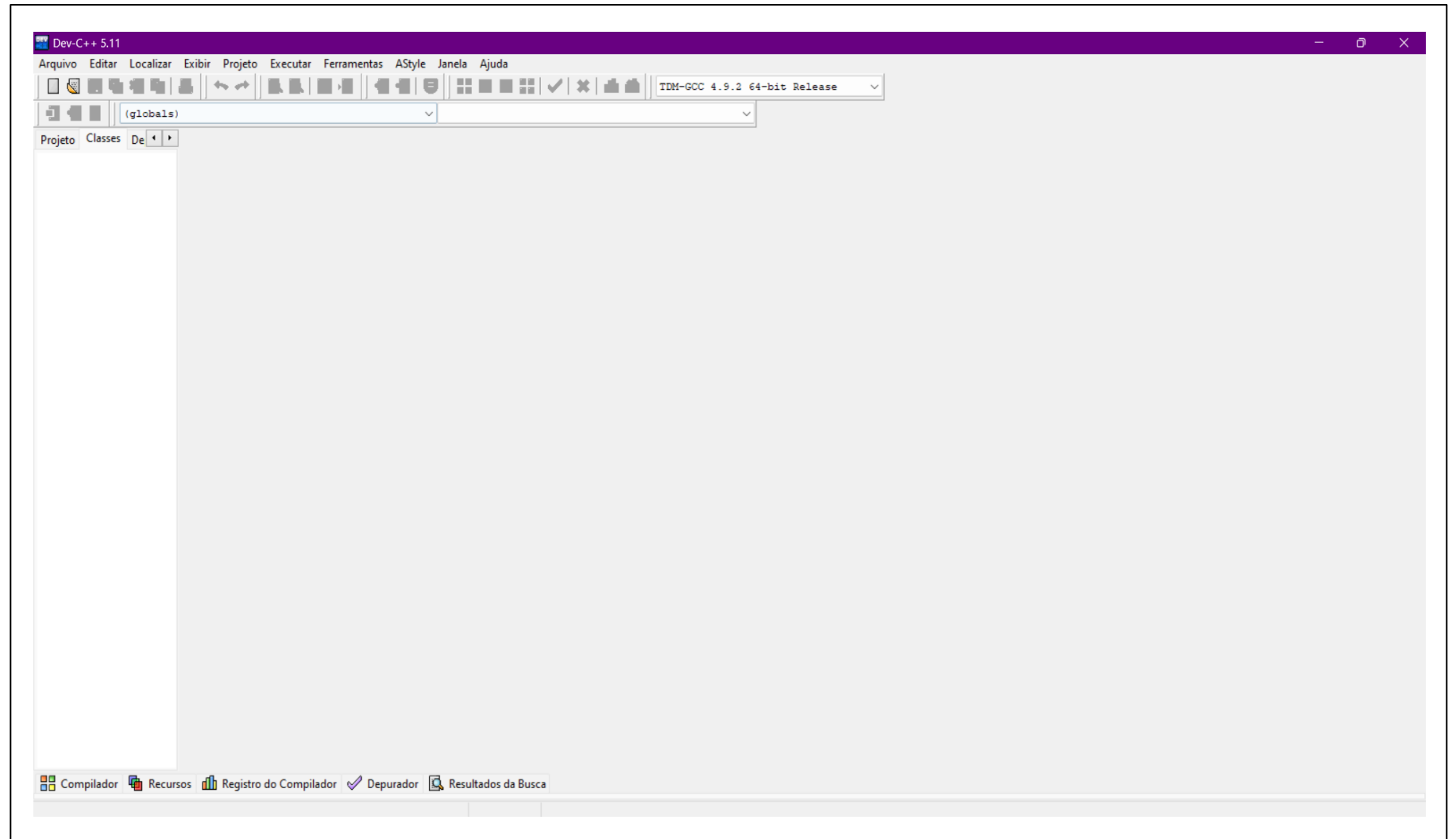
```
C:\Users\CopsT\Desktop\calculo\calcmidia.cpp - [Executing] - Dev-C++ 5.11
Arquivo Editar Localizar Exibir Projeto Executar Ferramentas AStyle Janela Ajuda
(globals)
Projeto Classes De
calculaMedia(float n
main() : int
C:\Users\CopsT\Desktop\calcu x + v
Insira a primeira nota: 8
Insira a segunda nota: 5
Insira a terceira nota: 3
A media das notas e: 5.33

-----
Process exited after 13.32 seconds with return value 0
Press any key to continue . . .
```

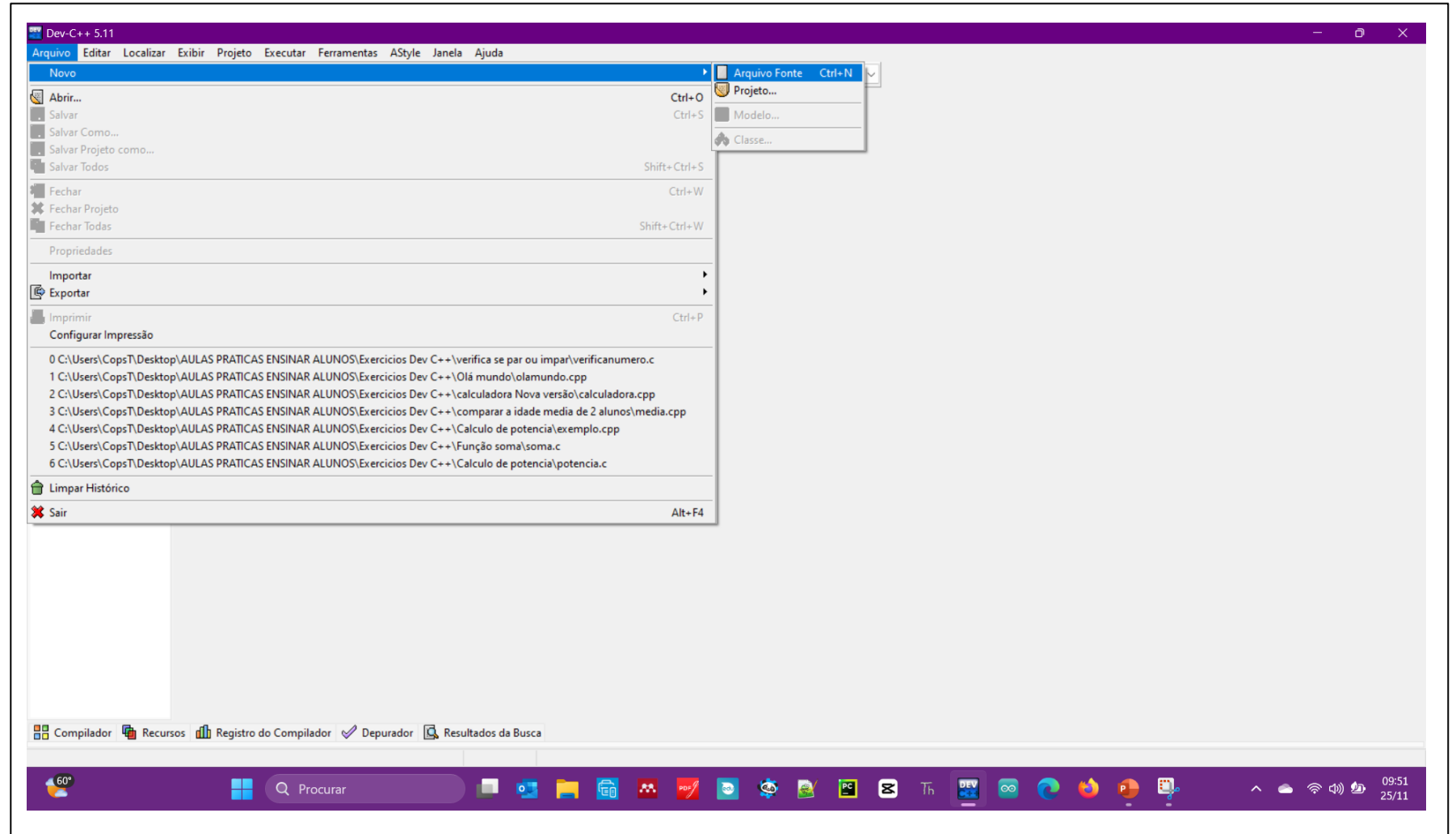

EXERCICIO 3 CONDUZIDO STEP BY STEP

- Desenvolva um programa que calcule a média de três ponderações diferentes sendo que o trabalho vale (25%), a ficha de trabalho (15%) e um teste (60%).
- Crie uma função com o nome calculaMedia() e que retorne o valor da média. O programa deve exibir a média no ecrã.

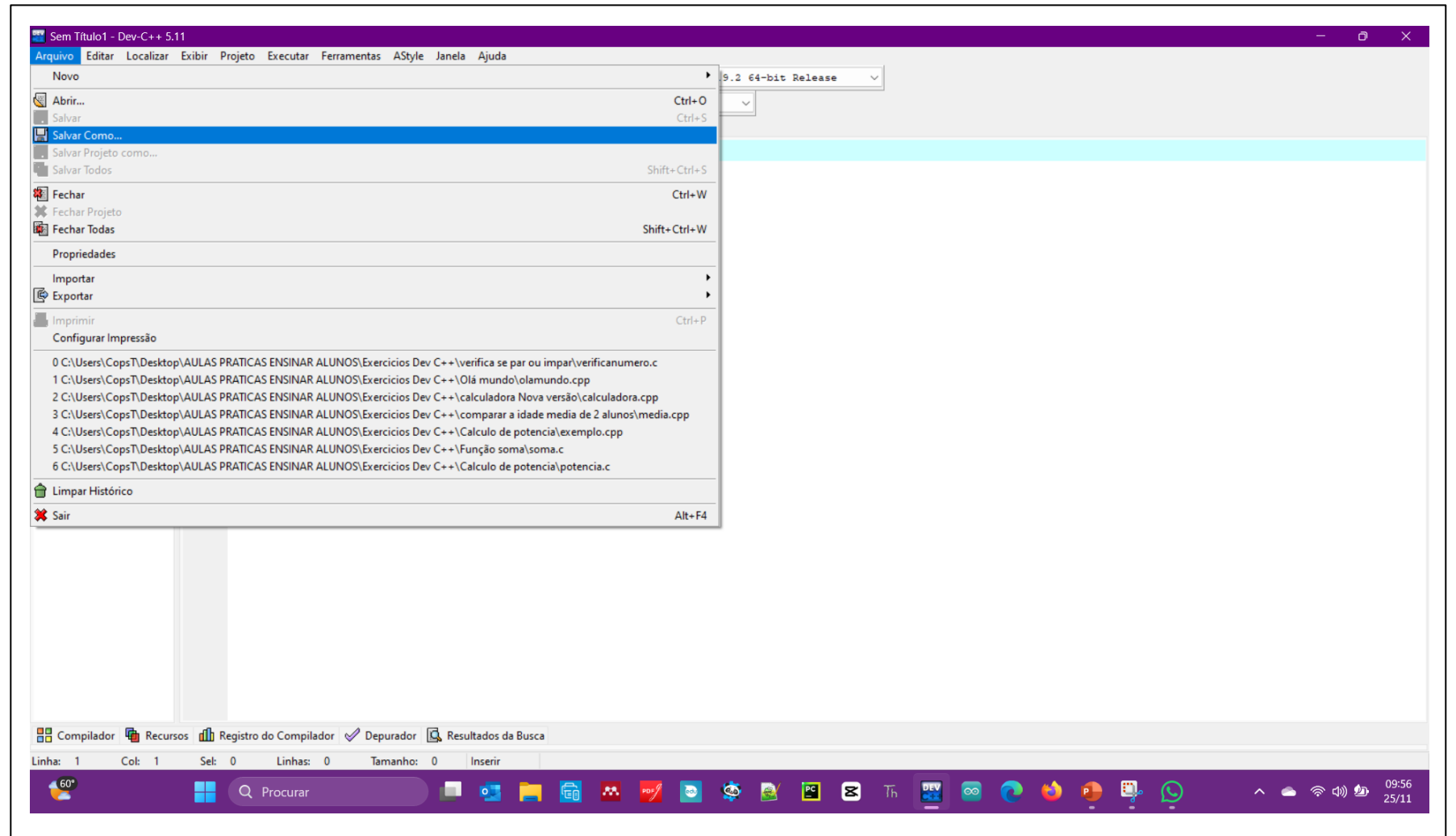
Passo 1 – Abrir o program Dev C++



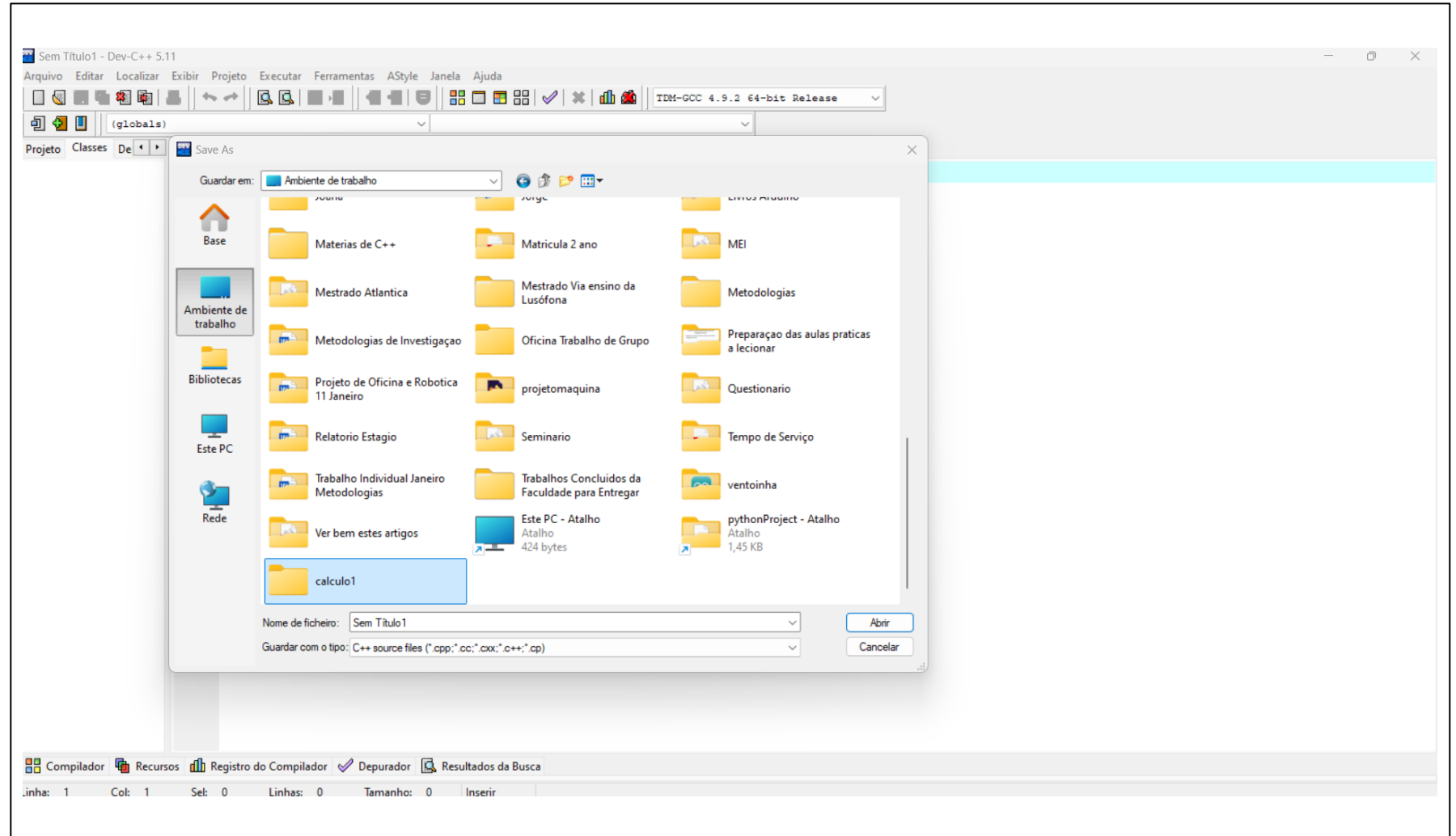
Passo 2 – Vamos a arquivo – Novo – Arquivo Fonte



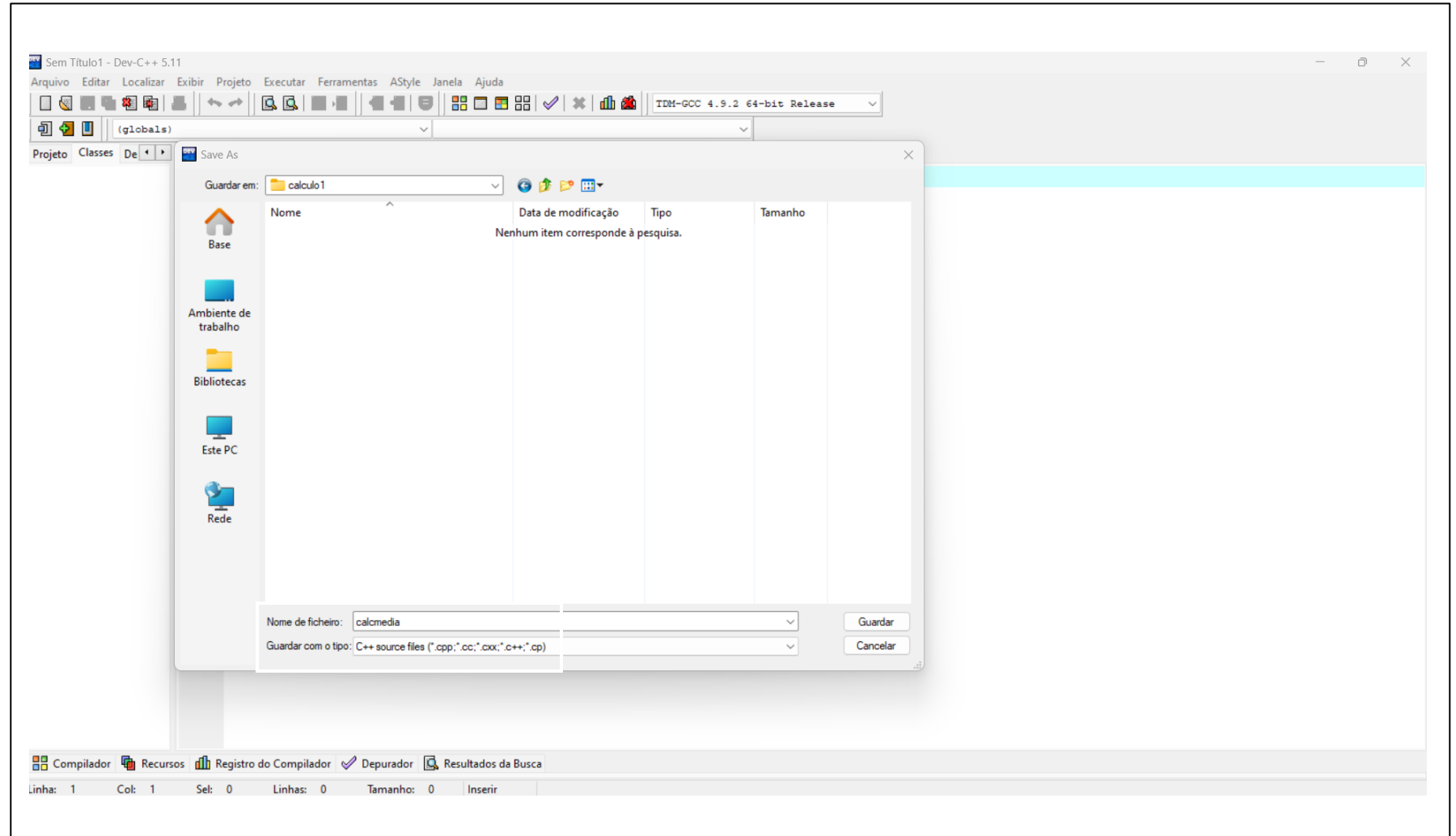
Passo 3 – Vamos agora salvar o ficheiro e vamos a salvar como...



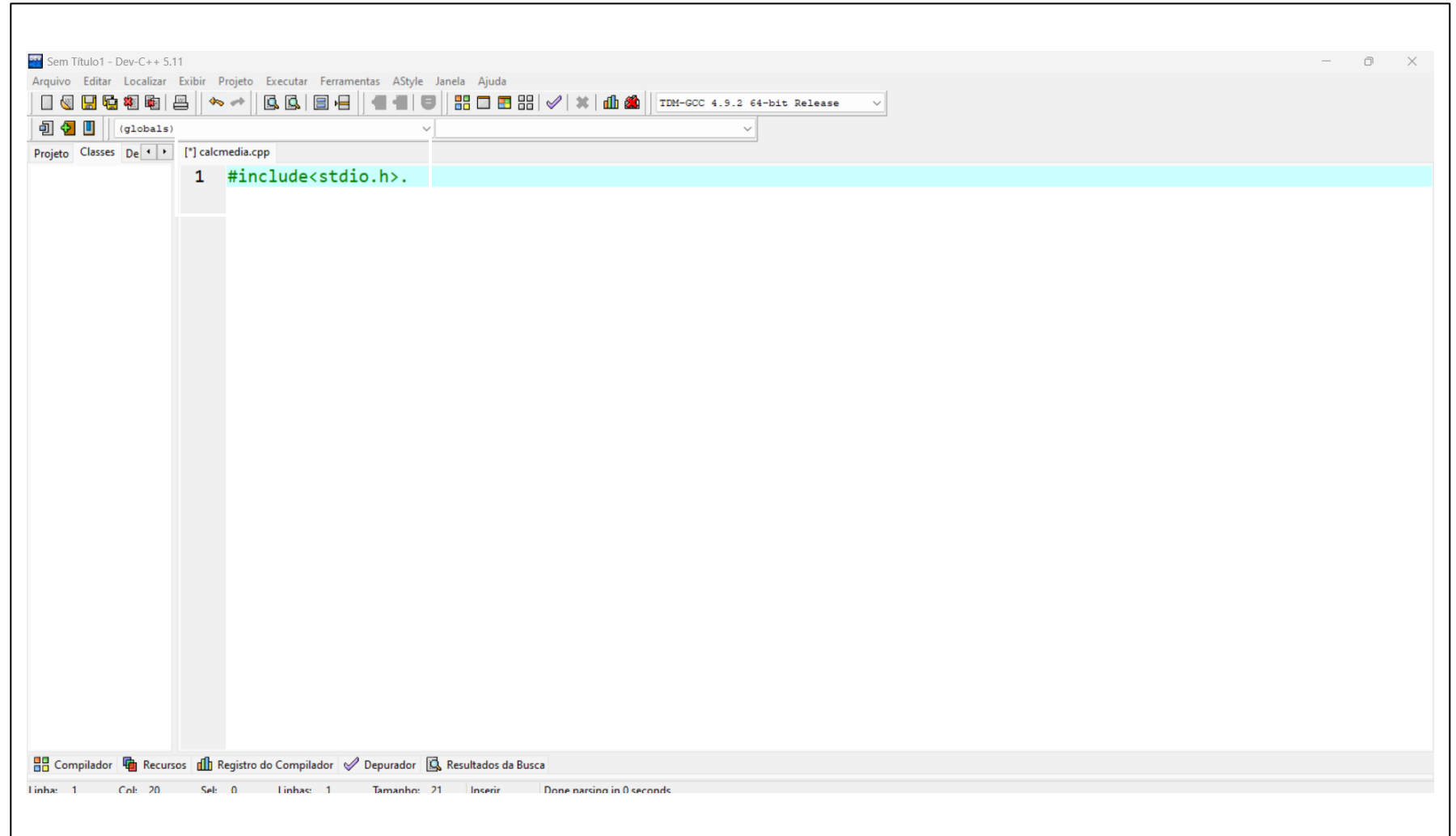
Passo 4 – Vamos agora salvar o ficheiro com o nome “calcmedia” numa pasta no ambiente de trabalho dentro da pasta “calculo1”



Passo 5 – Vamos agora salvar o ficheiro com o nome “calcmedia” numa pasta no ambiente de trabalho dentro da pasta “calculo1”

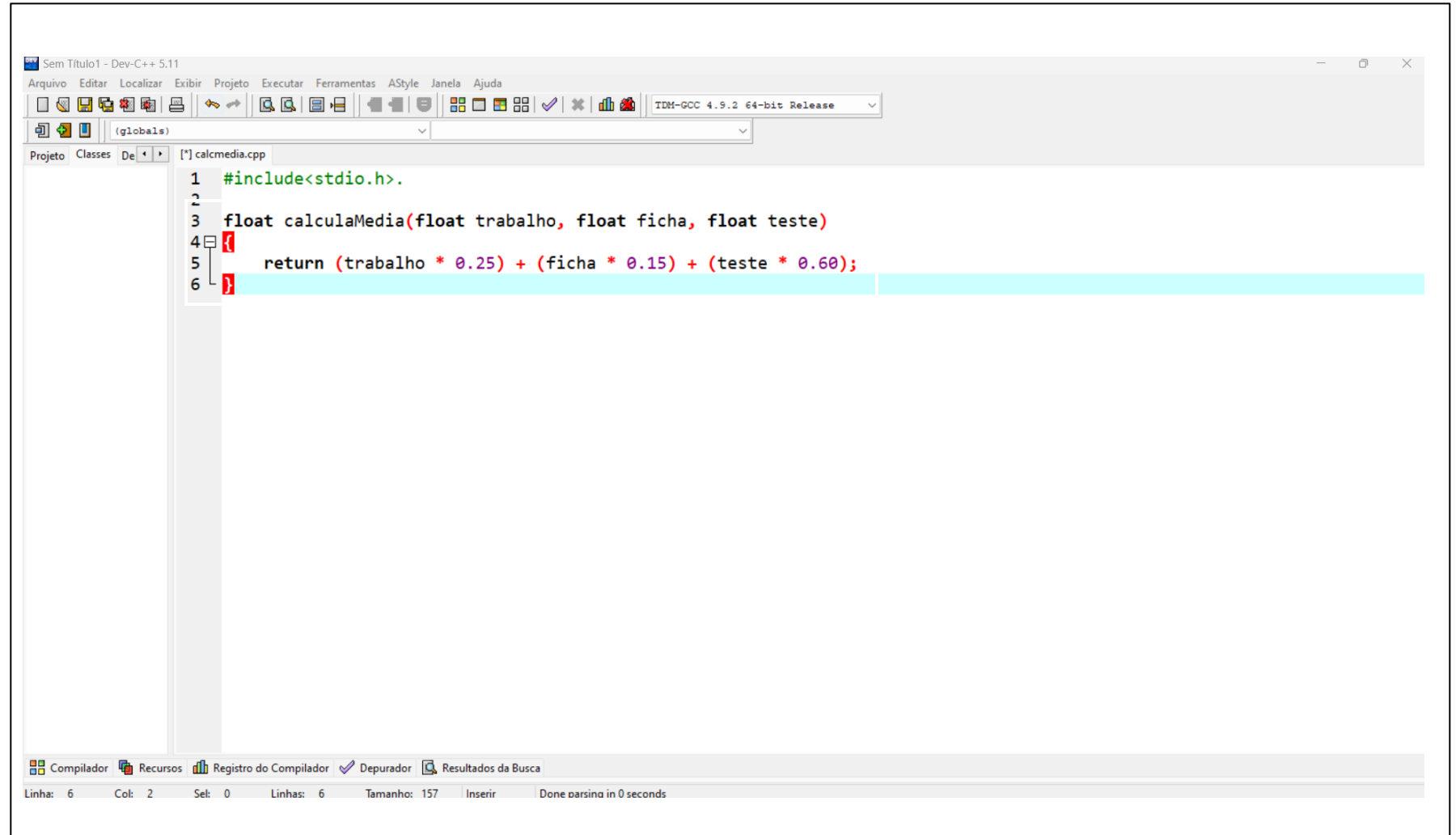


Passo 6 – No topo do programa do Dev C++ linha 1 devemos incluir a biblioteca - `#include<stdio.h>`



The image shows a screenshot of the Dev-C++ IDE. The window title is "Sem Título1 - Dev-C++ 5.11". The menu bar includes "Arquivo", "Editar", "Localizar", "Exibir", "Projeto", "Executar", "Ferramentas", "AStyle", "Janela", and "Ajuda". The toolbar contains various icons for file operations and execution. The compiler is set to "TDM-GCC 4.9.2 64-bit Release". The project name is "(globals)". The file name is "[*] calcmidia.cpp". The main editor window shows the first line of code: `1 #include<stdio.h>.` The status bar at the bottom indicates "Linha: 1", "Col: 20", "Sel: 0", "Linhas: 1", "Tamanho: 21", and "Done parsing in 0 seconds".

Passo 7 – Vamos declarar a função float calculaMedia(float trabalho, float ficha, float teste) antes da função main()

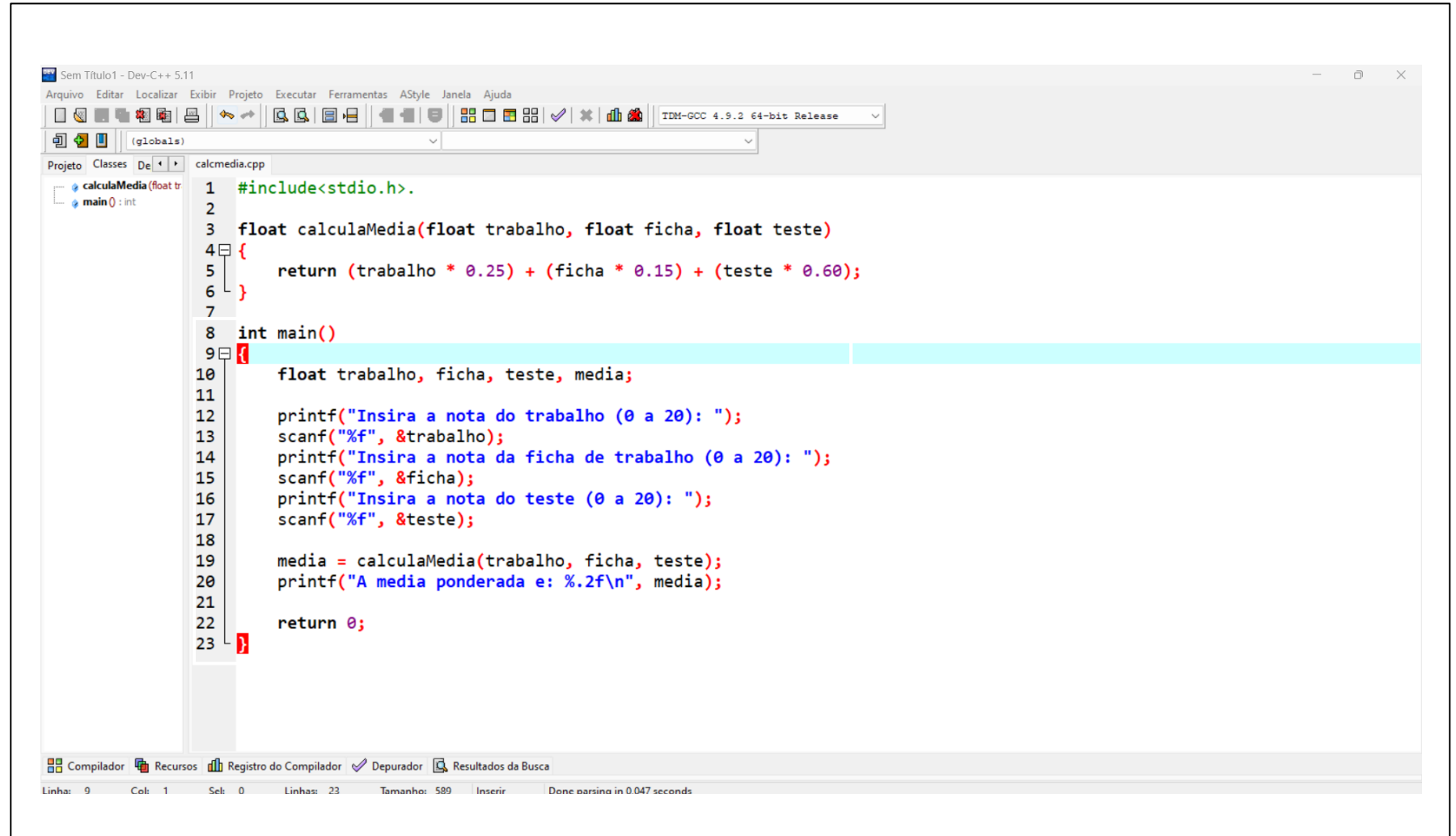


The screenshot shows a C++ IDE window titled "Sem Título1 - Dev-C++ 5.11". The menu bar includes "Arquivo", "Editar", "Localizar", "Exibir", "Projeto", "Executar", "Ferramentas", "AStyle", "Janela", and "Ajuda". The toolbar contains various icons for file operations and execution. The compiler is set to "TDM-GCC 4.9.2 64-bit Release". The project name is "(globals)". The active file is "calcmedia.cpp". The code in the editor is as follows:

```
1 #include<stdio.h>.  
2  
3 float calculaMedia(float trabalho, float ficha, float teste)  
4 {  
5     return (trabalho * 0.25) + (ficha * 0.15) + (teste * 0.60);  
6 }
```

The status bar at the bottom shows: "Compilador", "Recursos", "Registro do Compilador", "Depurador", and "Resultados da Busca". The status text reads: "Linha: 6 Col: 2 Set: 0 Linhas: 6 Tamanho: 157 Inserir Done parsing in 0 seconds".

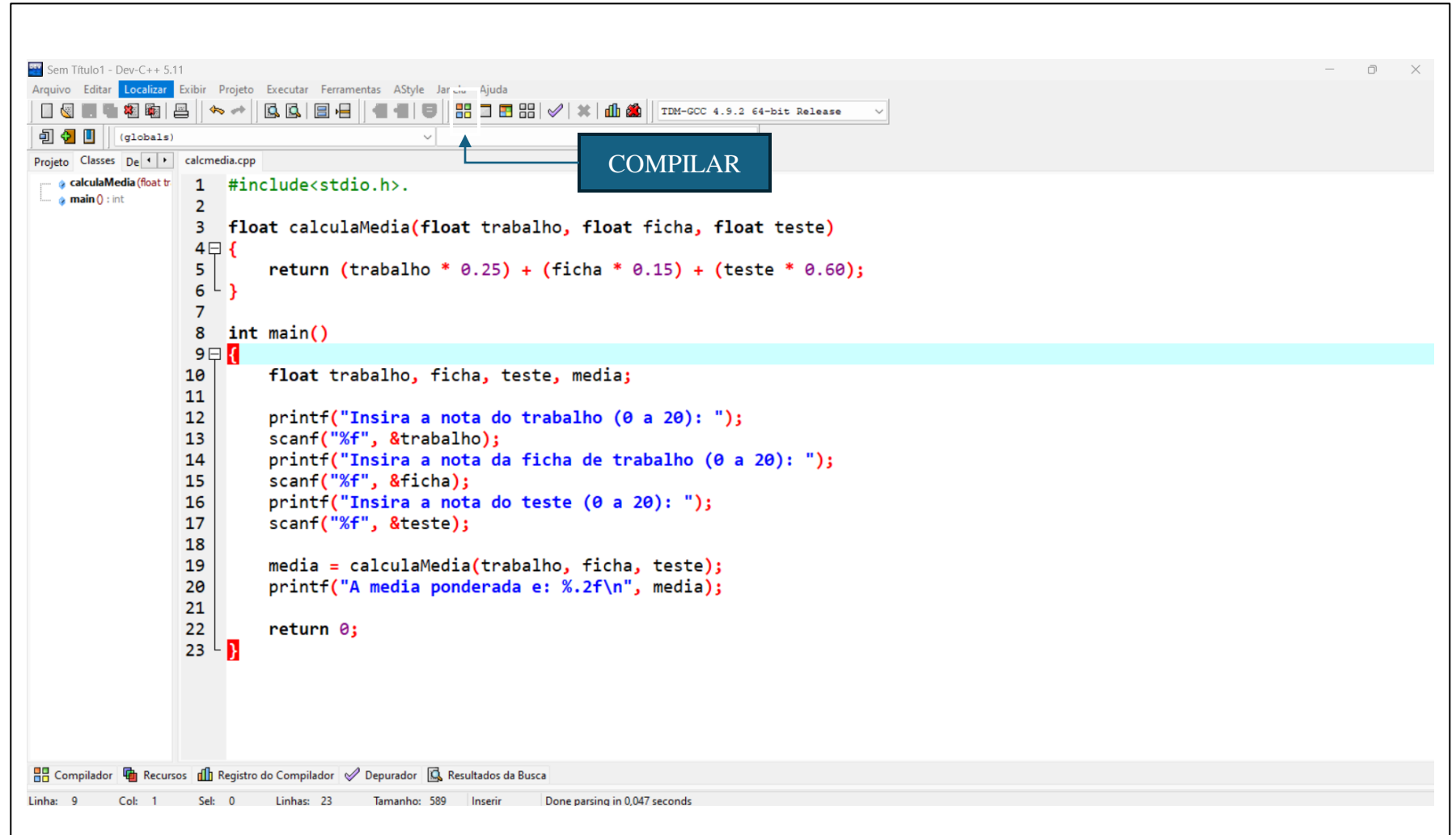
Passo 8 – Vamos agora escrever o resto do programa utilizando a função main()



```
1 #include<stdio.h>.
2
3 float calculaMedia(float trabalho, float ficha, float teste)
4 {
5     return (trabalho * 0.25) + (ficha * 0.15) + (teste * 0.60);
6 }
7
8 int main()
9 {
10     float trabalho, ficha, teste, media;
11
12     printf("Insira a nota do trabalho (0 a 20): ");
13     scanf("%f", &trabalho);
14     printf("Insira a nota da ficha de trabalho (0 a 20): ");
15     scanf("%f", &ficha);
16     printf("Insira a nota do teste (0 a 20): ");
17     scanf("%f", &teste);
18
19     media = calculaMedia(trabalho, ficha, teste);
20     printf("A media ponderada e: %.2f\n", media);
21
22     return 0;
23 }
```

Linhas: 9 Col: 1 Sel: 0 Linhas: 23 Tamanho: 589 Inserir Done parsing in 0.047 seconds

Passo 9 – Vamos agora compilar o programa para verificarmos se existem erros no código



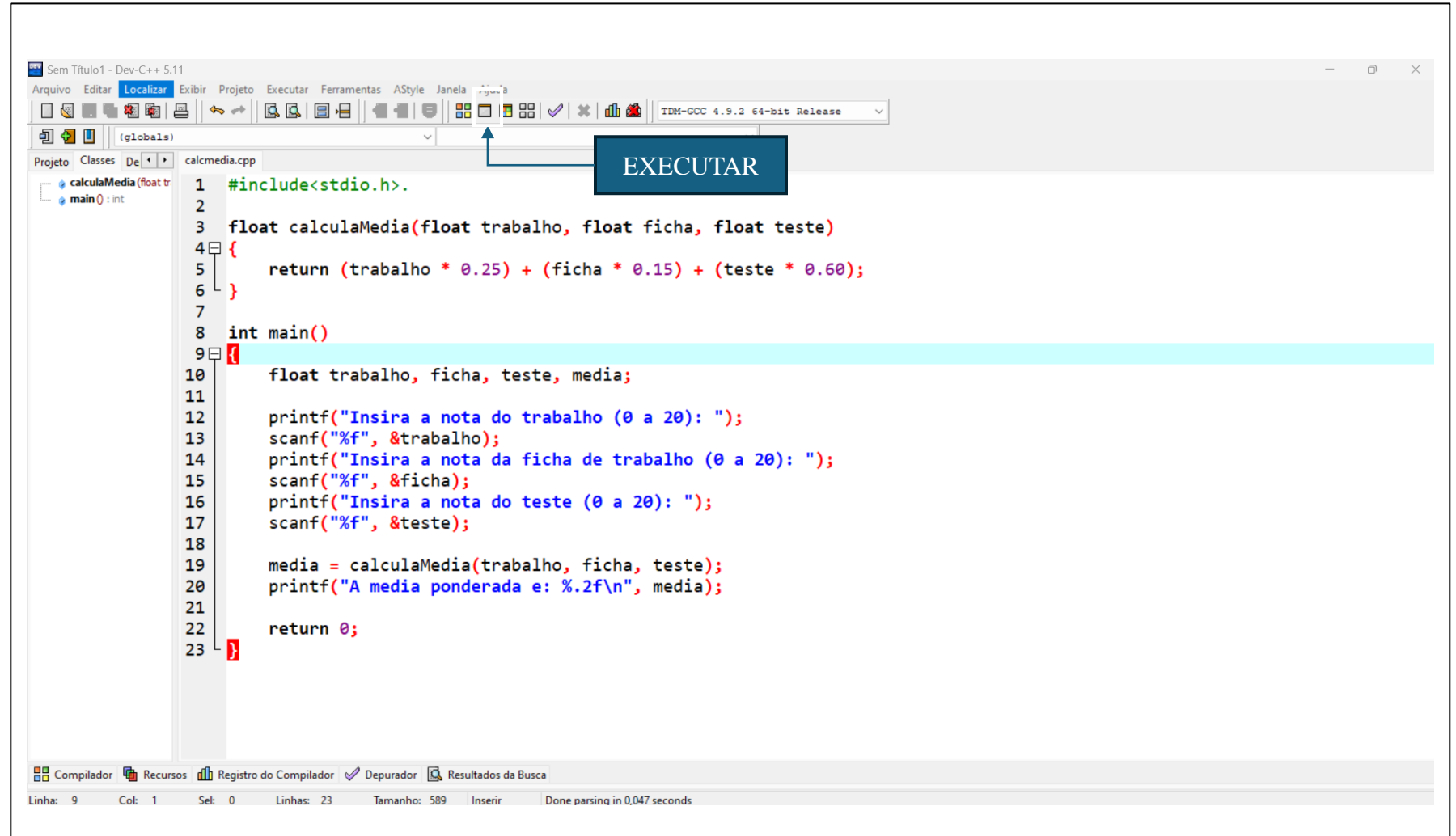
```
1 #include<stdio.h>.
2
3 float calculaMedia(float trabalho, float ficha, float teste)
4 {
5     return (trabalho * 0.25) + (ficha * 0.15) + (teste * 0.60);
6 }
7
8 int main()
9 {
10     float trabalho, ficha, teste, media;
11
12     printf("Insira a nota do trabalho (0 a 20): ");
13     scanf("%f", &trabalho);
14     printf("Insira a nota da ficha de trabalho (0 a 20): ");
15     scanf("%f", &ficha);
16     printf("Insira a nota do teste (0 a 20): ");
17     scanf("%f", &teste);
18
19     media = calculaMedia(trabalho, ficha, teste);
20     printf("A media ponderada e: %.2f\n", media);
21
22     return 0;
23 }
```

COMPILAR

Compilador Recursos Registro do Compilador Depurador Resultados da Busca

Linha: 9 Col: 1 Sel: 0 Linhas: 23 Tamanho: 589 Inserir Done parsing in 0,047 seconds

Passo 10 – Vamos agora executar o programa e ver o resultado



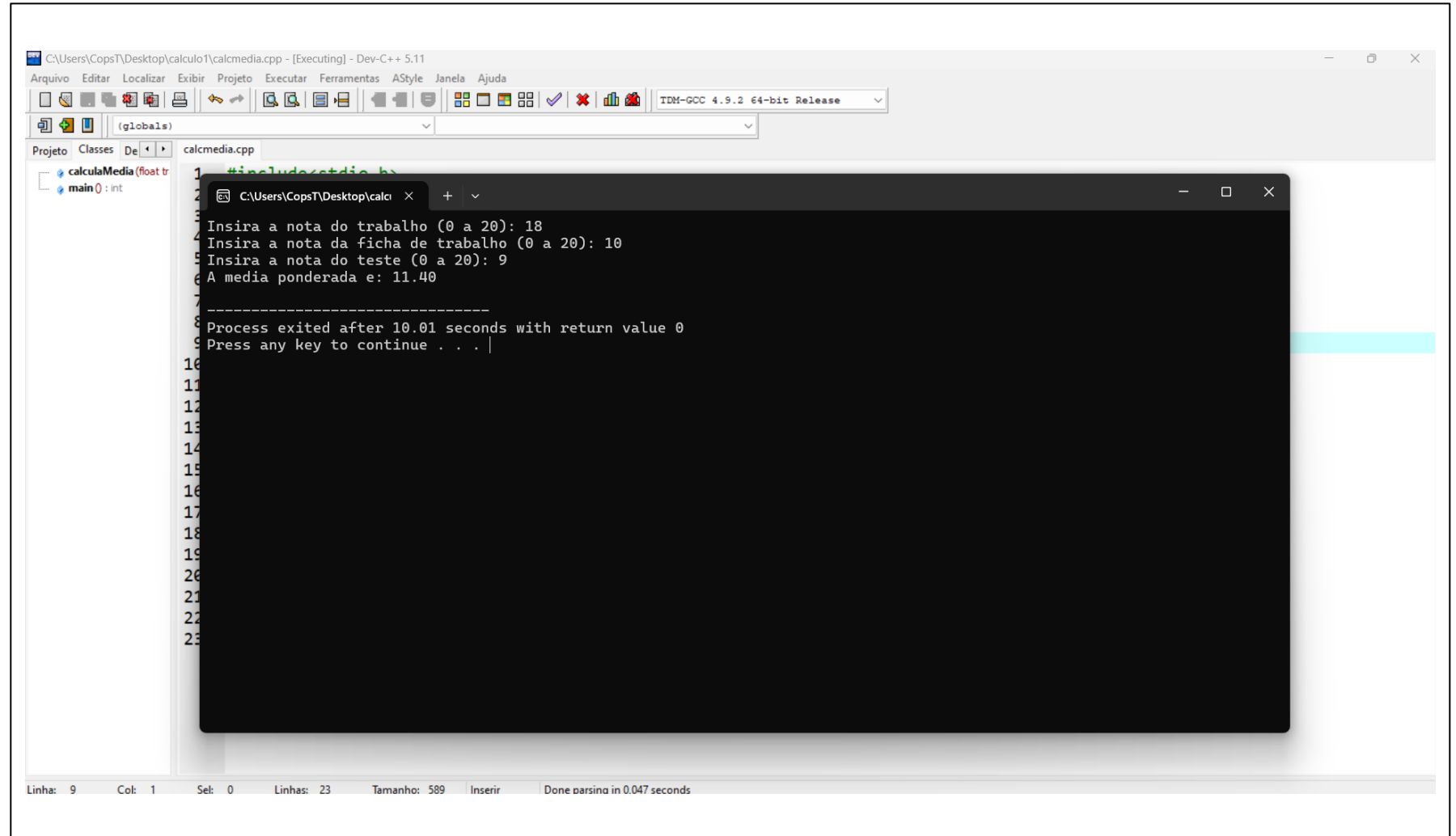
```
1 #include<stdio.h>.
2
3 float calculaMedia(float trabalho, float ficha, float teste)
4 {
5     return (trabalho * 0.25) + (ficha * 0.15) + (teste * 0.60);
6 }
7
8 int main()
9 {
10     float trabalho, ficha, teste, media;
11
12     printf("Insira a nota do trabalho (0 a 20): ");
13     scanf("%f", &trabalho);
14     printf("Insira a nota da ficha de trabalho (0 a 20): ");
15     scanf("%f", &ficha);
16     printf("Insira a nota do teste (0 a 20): ");
17     scanf("%f", &teste);
18
19     media = calculaMedia(trabalho, ficha, teste);
20     printf("A media ponderada e: %.2f\n", media);
21
22     return 0;
23 }
```

EXECUTAR

Compilador Recursos Registro do Compilador Depurador Resultados da Busca

Linha: 9 Col: 1 Sel: 0 Linhas: 23 Tamanho: 589 Inserir Done parsing in 0,047 seconds

Passo 11 – Vamos agora executar e ver a nossa consola



```
C:\Users\CopsT\Desktop\calculo1\calcmidia.cpp - [Executing] - Dev-C++ 5.11
Arquivo Editar Localizar Exibir Projeto Executar Ferramentas AStyle Janela Ajuda
(globals)
Projeto Classes De
calcmidia.cpp
calculaMedia(float tr
main() : int
1 #include <stdio.h>
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
C:\Users\CopsT\Desktop\calcu
+
-
Insira a nota do trabalho (0 a 20): 18
Insira a nota da ficha de trabalho (0 a 20): 10
Insira a nota do teste (0 a 20): 9
A media ponderada e: 11.40
-----
Process exited after 10.01 seconds with return value 0
Press any key to continue . . . |
Linha: 9 Col: 1 Sel: 0 Linhas: 23 Tamanho: 589 Inserir Done parsing in 0.047 seconds
```

INICIO DO EXERCICIO 3 CONDUZIDO STEP BY STEP

