Funções e Procedimentos

ECM404

Funções

- Funções são blocos de código que podem ser nomeados e chamados em um programa:
  - **printf() –** função que escreve um valor na tela;
  - **scanf() –** função que lê um valor digitado no teclado.
- **Estruturação:** programas grandes e complexos são constituídos bloco a bloco;
- **Reutilização:** usar funções evita a cópia desnecessária de trechos de código que realizam a mesma tarefa.

Funções

```c
float media(int x, int y, float z);

int main (){
  int a,b;
  float m;
  printf("Digite o valor de a: ");
  scanf("%i",&a);
  printf("Digite o valor de b: ");
  scanf("%i",&b);
  m = media(a,b,2.0);
  return 0;
}

float media(int x, int y, float z){
  return (x+y)/z;
}
```

Funções – Protótipo da função

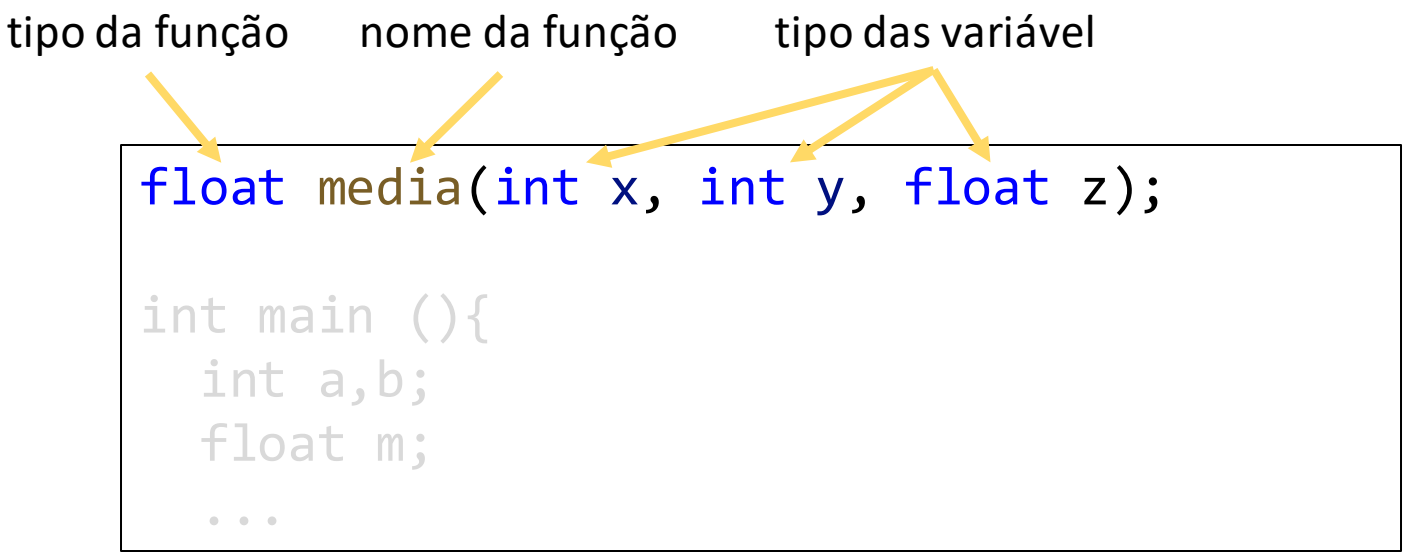
- Protótipo da função:
  - Funções devem ser declaradas antes de serem utilizadas. Assim, definir globalmente os protótipos das funções, permite chamá-las no programa principal (ou dentro de outras funções).
  - Itens obrigatórios: tipo da função, nome da função e tipo das variáveis.

```
float media(int x, int y, float z);

int main (){
  int a,b;
  float m;
  ...
```

Funções – Protótipo da função

- Protótipo da função:
  - Itens obrigatórios: tipo da função, nome da função e tipo das variáveis.



Funções – Chamada da função

- Chamada da função media na função main:

```
float media(int x, int y, float z);

int main (){
  int a,b;
  float m;
  printf("Digite o valor de a: ");
  scanf("%i",&a);
  printf("Digite o valor de b: ");
  scanf("%i",&b);
  m = media(a,b,2.0);
  return 0;
}

float media(int x, int y, float z){
  return (x+y)/z;
}
```

Funções – Declaração da função

- Declaração da função:
  - Deve-se colocar:
    - tipo da função;
    - nome da função;
    - tipo das variáveis;
    - nome das variáveis;
    - corpo da função (comandos);
    - return.

```
...
return 0;
}


float media(int x, int y, float z){
  return (x+y)/z;
}
```

Funções – Corpo da função

- Corpo da função:
  - Variáveis locais: são aquelas que só tem validade dentro da função na qual foram declaradas.

```
...
return 0;
}


float media(int x, int y, float z){
    float m;
    m = (x+y)/z;
    return m;
}
```

Funções – Return

- Uma função pode ou não retornar um valor:
  - Se ela retornar um valor, alguém deverá receber este valor na chamada;
  - Uma função que não retorna um valor é chamada de **procedimento** e seu tipo deve ser declarado como **void**.

```
void nome da função ( parâmetros ){
  sequência de comandos;
  return;  // pode-se omitir
}
```

Funções – Return

- O valor de retorno tem que ser compatível com o tipo declarado para a função. Pode-se retornar qualquer valor válido em C:
  - Tipos pré-definidos: **int, float, char** e **double**;
  - Tipos definidos pelo usuário: **struct**.
- Uma função pode ter mais que um **return**:
  - Quando o comando **return** é executado, a função termina imediatamente;
  - Todos os comandos restantes são **ignorados**.

Funções – Boas Práticas

- De modo geral, evita-se fazer operações de leitura e escrita dentro de uma função. Isso assegura que a função possa ser utilizada nas mais diversas aplicações, garantindo a sua generalidade.
- Evite variáveis globais! Elas podem ser alteradas por todas as funções do programa. Entretanto, quando houver uma variável local como mesmo nome que a global, a função dará preferência para a variável local.

Depuração de código

Como depurar para iniciantes absolutos

Artigo • 12/01/2022 • 8 minutos para o fim da leitura • 2 colaboradores

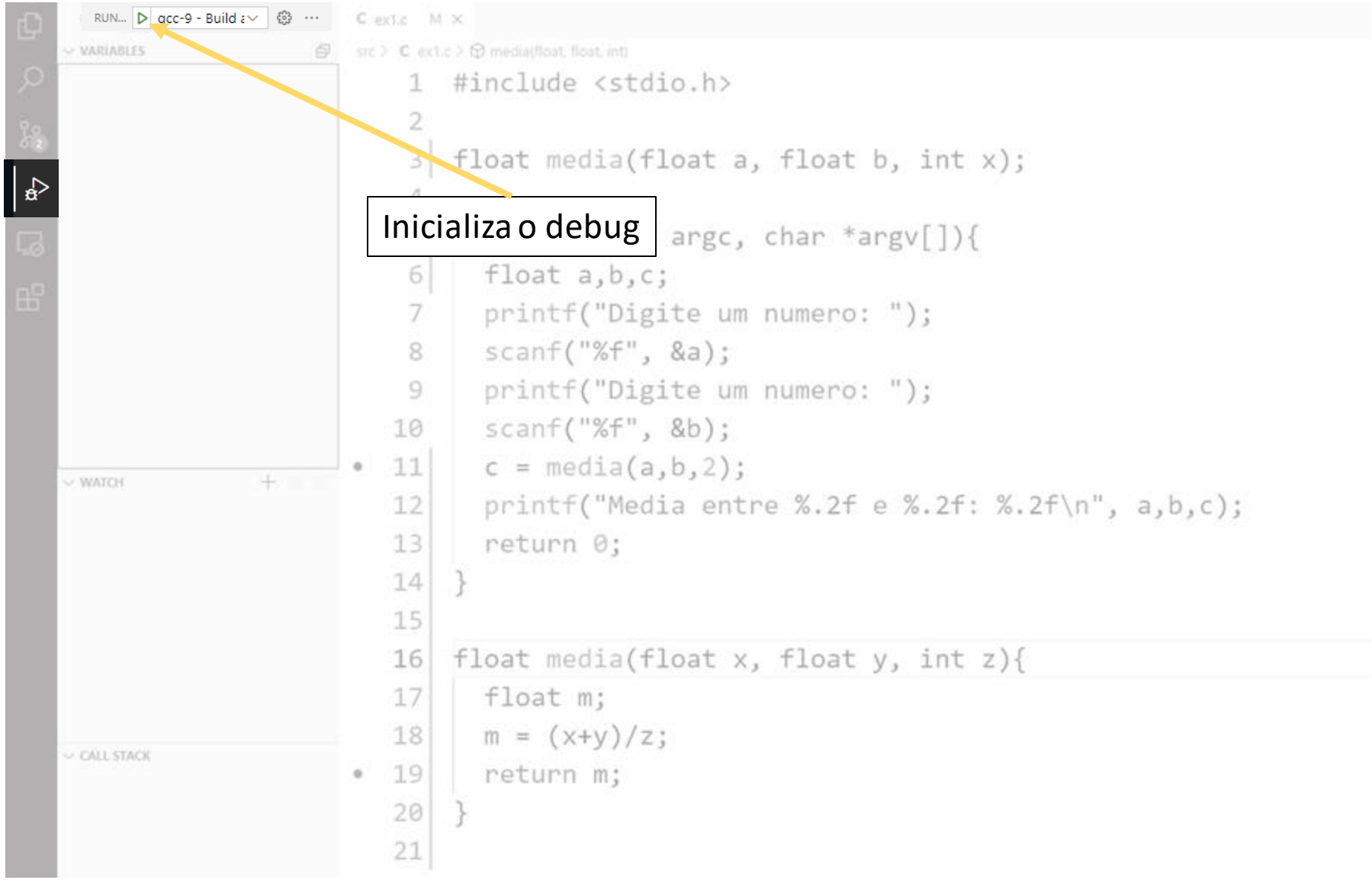
Sem falha, o código que escrevemos como desenvolvedores de software nem sempre faz o que esperamos. Às vezes, ele faz algo completamente diferente! Quando isso acontece, a próxima tarefa é descobrir por que e, embora possamos ficar tentados a simplesmente encarar nosso código por horas, é muito mais fácil e eficiente usar uma ferramenta de depuração ou depurador.

Um depurador, infelizmente, não é algo que possa revelar magicamente todos os problemas ou "bugs" em nosso código. *Depuração* significa executar o código passo a passo em uma ferramenta de depuração, como o Visual Studio, para localizar o ponto exato em que você cometeu um erro de programação. Você entenderá quais correções são necessárias para fazer em seu código e as ferramentas de depuração geralmente permitem que você faça alterações temporárias para poder continuar executando o programa.

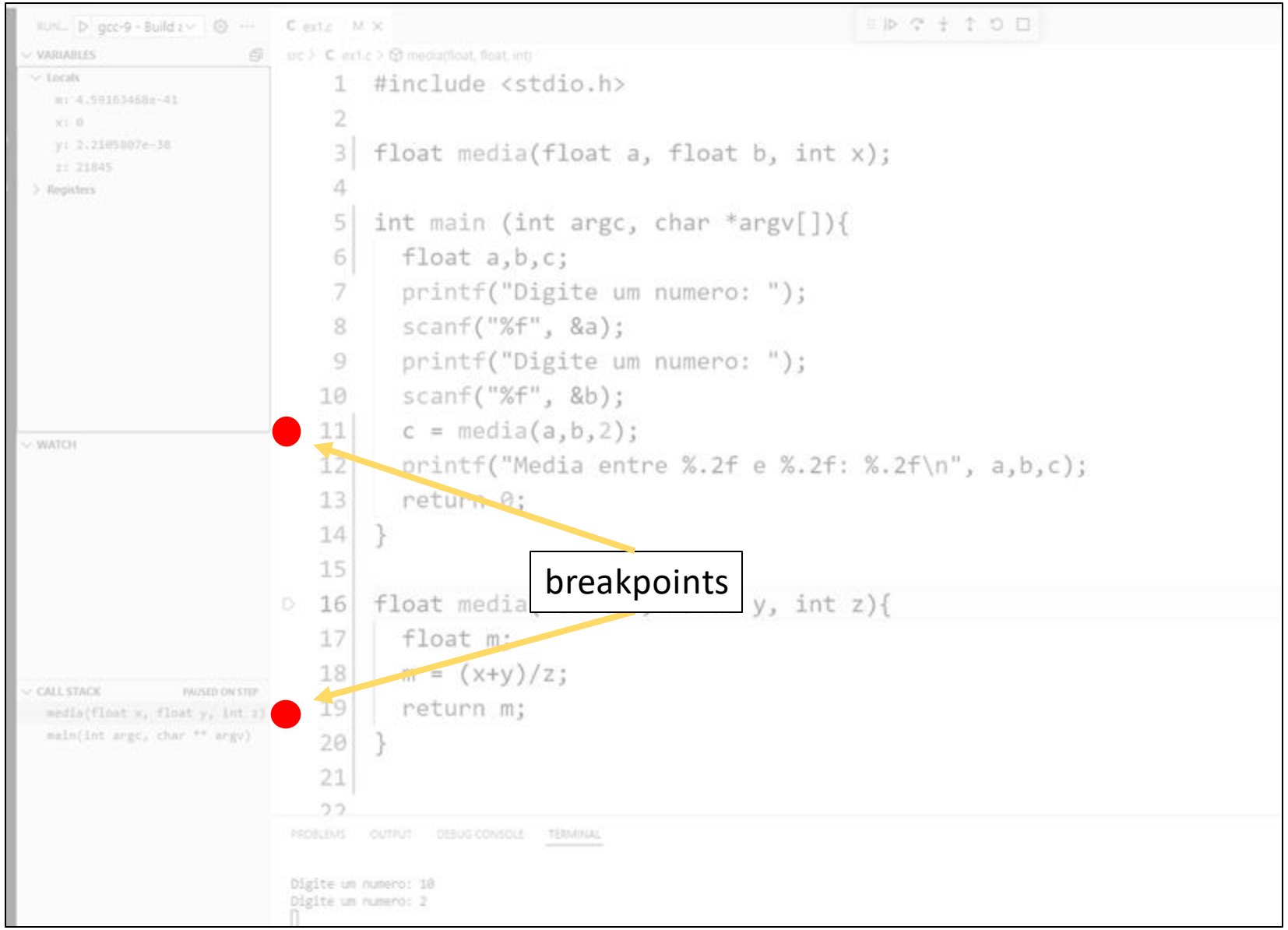
Usar um depurador com eficiência é também uma habilidade que leva tempo e prática, mas é definitivamente uma tarefa fundamental para todo desenvolvedor de software. Neste artigo, introduziremos os princípios fundamentais da depuração e forneceremos dicas para você começar.

Como depurar para iniciantes absolutos

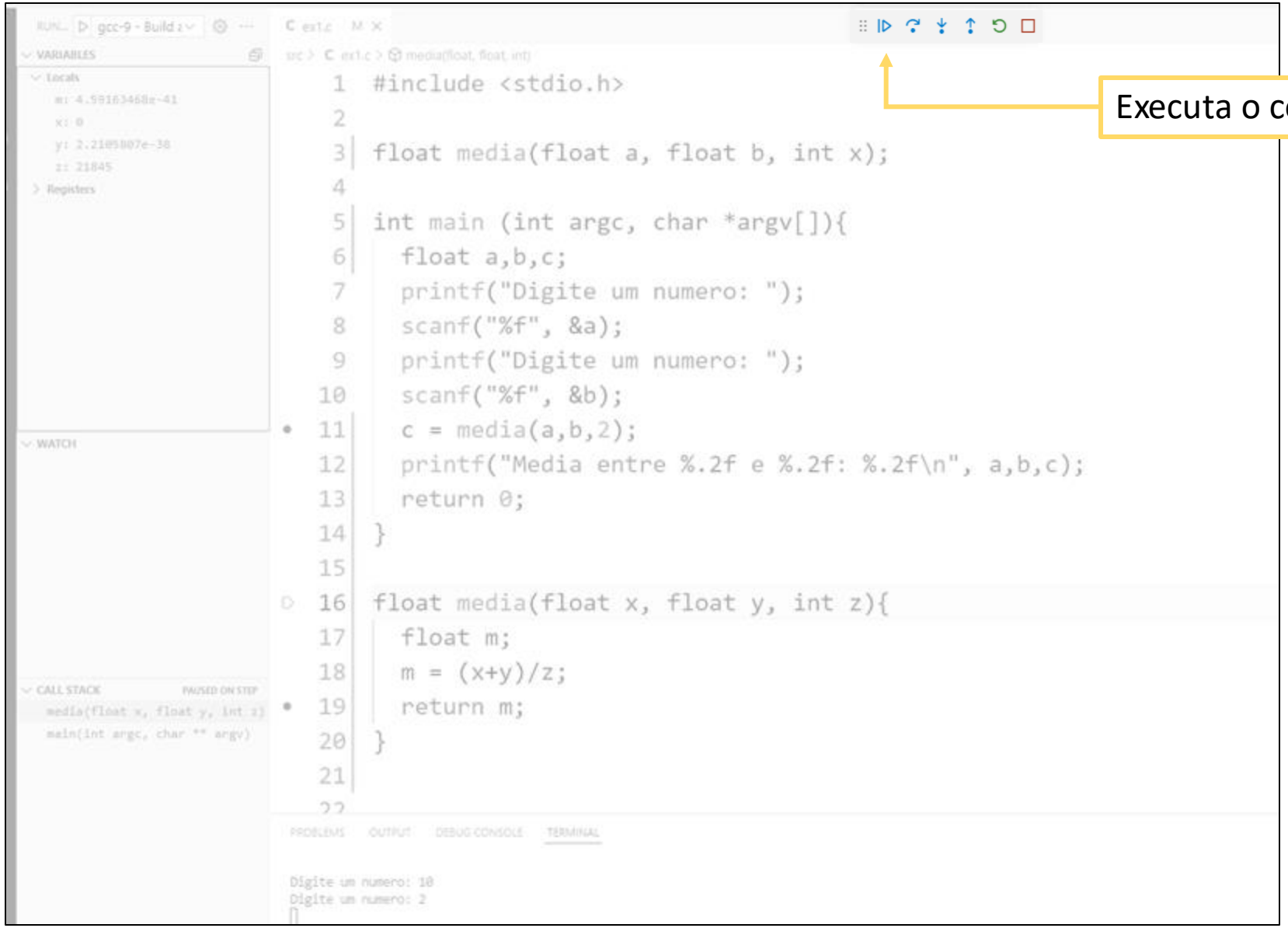
Depuração de código



Depuração de código

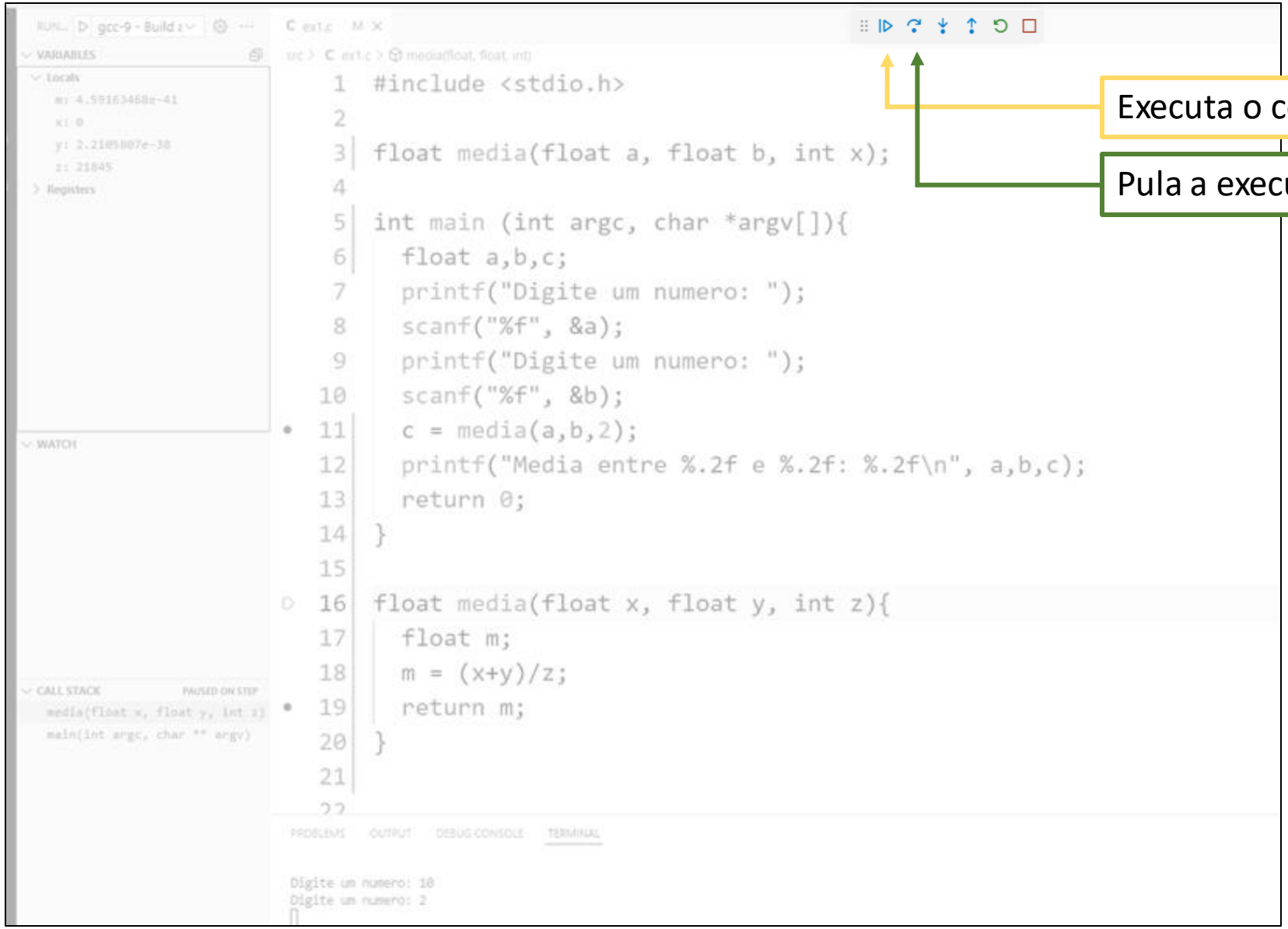


Depuração de código



Executa o código até um breakpoint

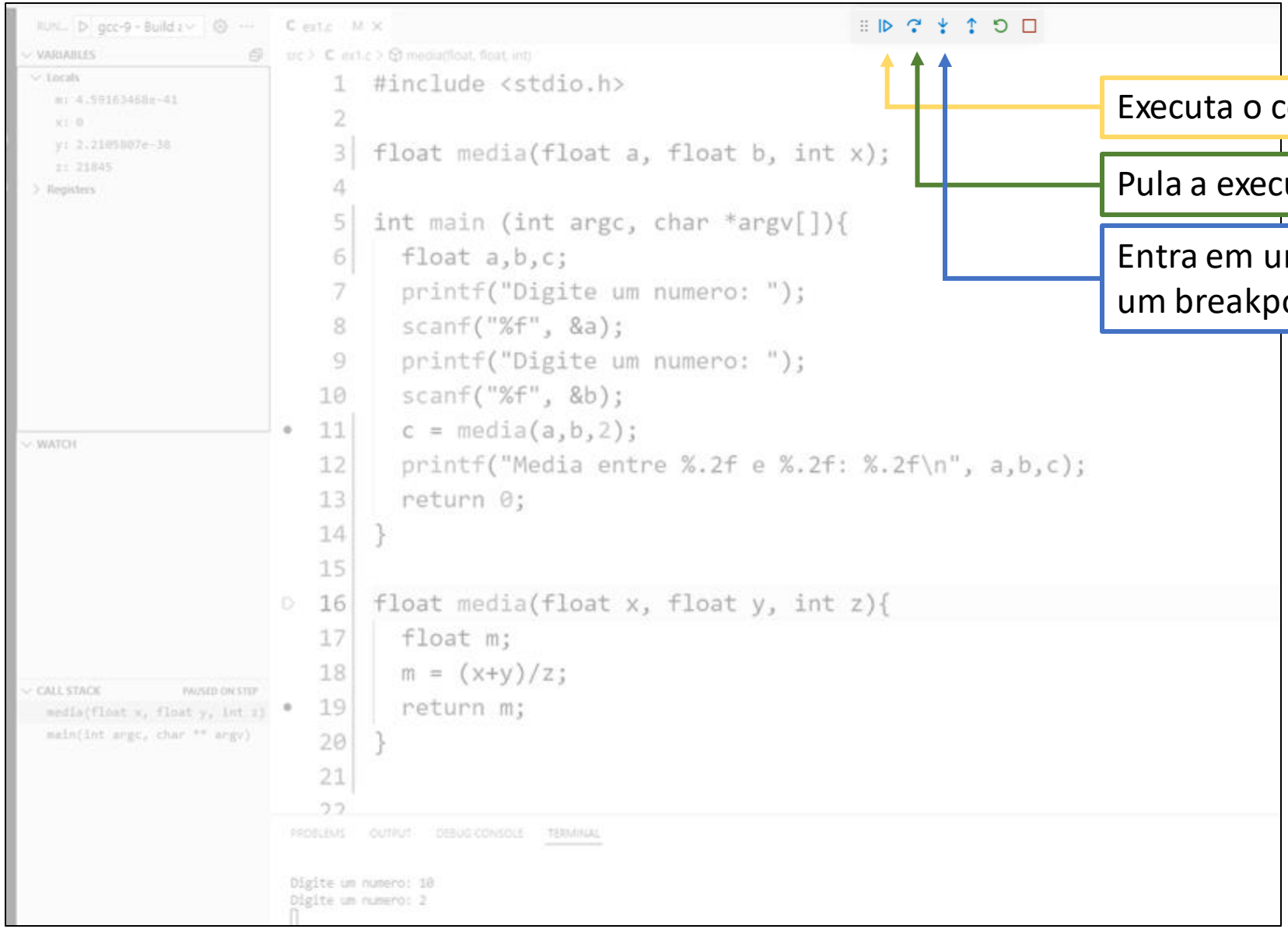
Depuração de código



Executa o código até um breakpoint

Pula a execução de um breakpoint

Depuração de código

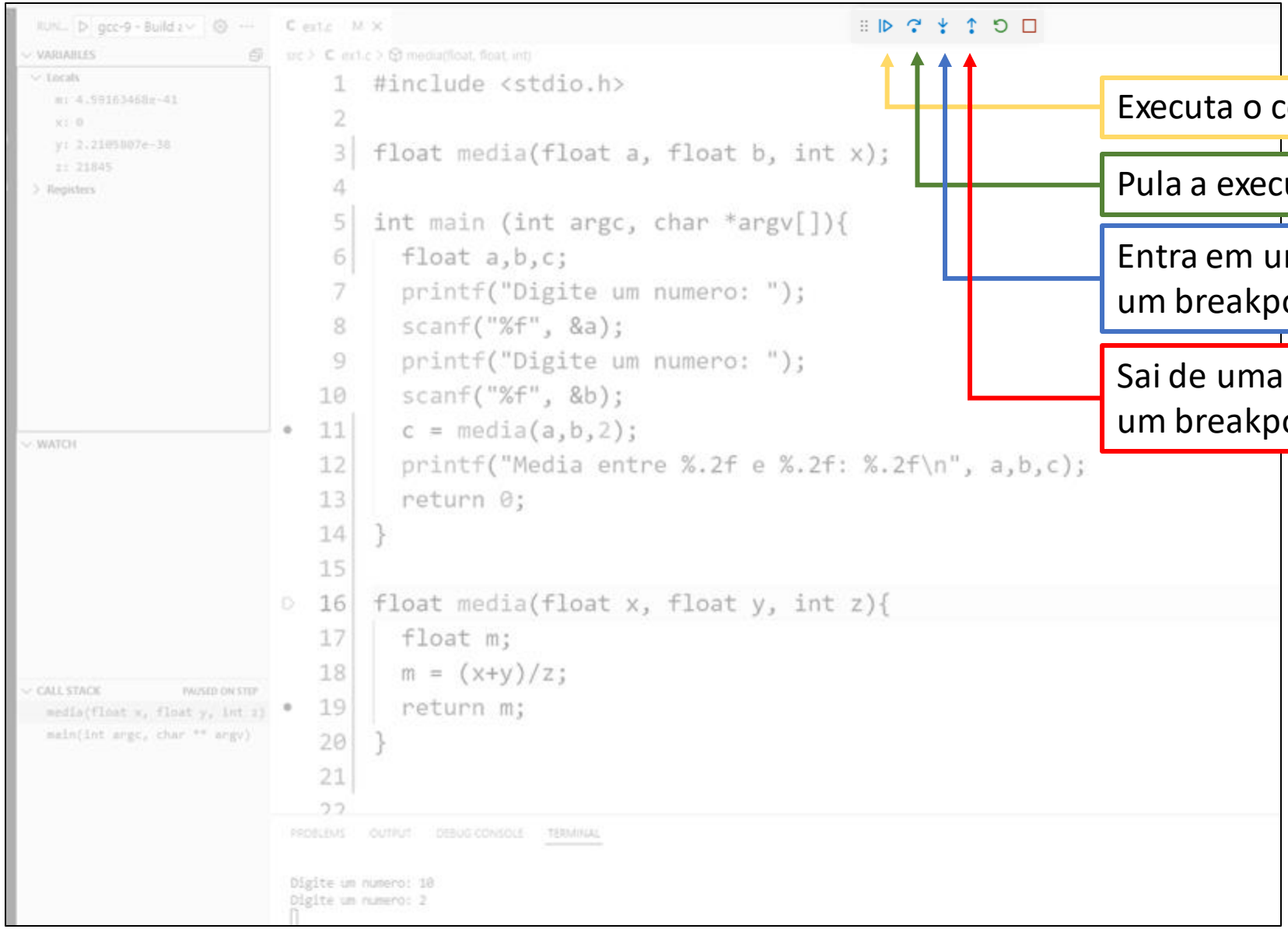


Executa o código até um breakpoint

Pula a execução de um breakpoint

Entra em uma sub-rotina marcada por um breakpoint

Depuração de código



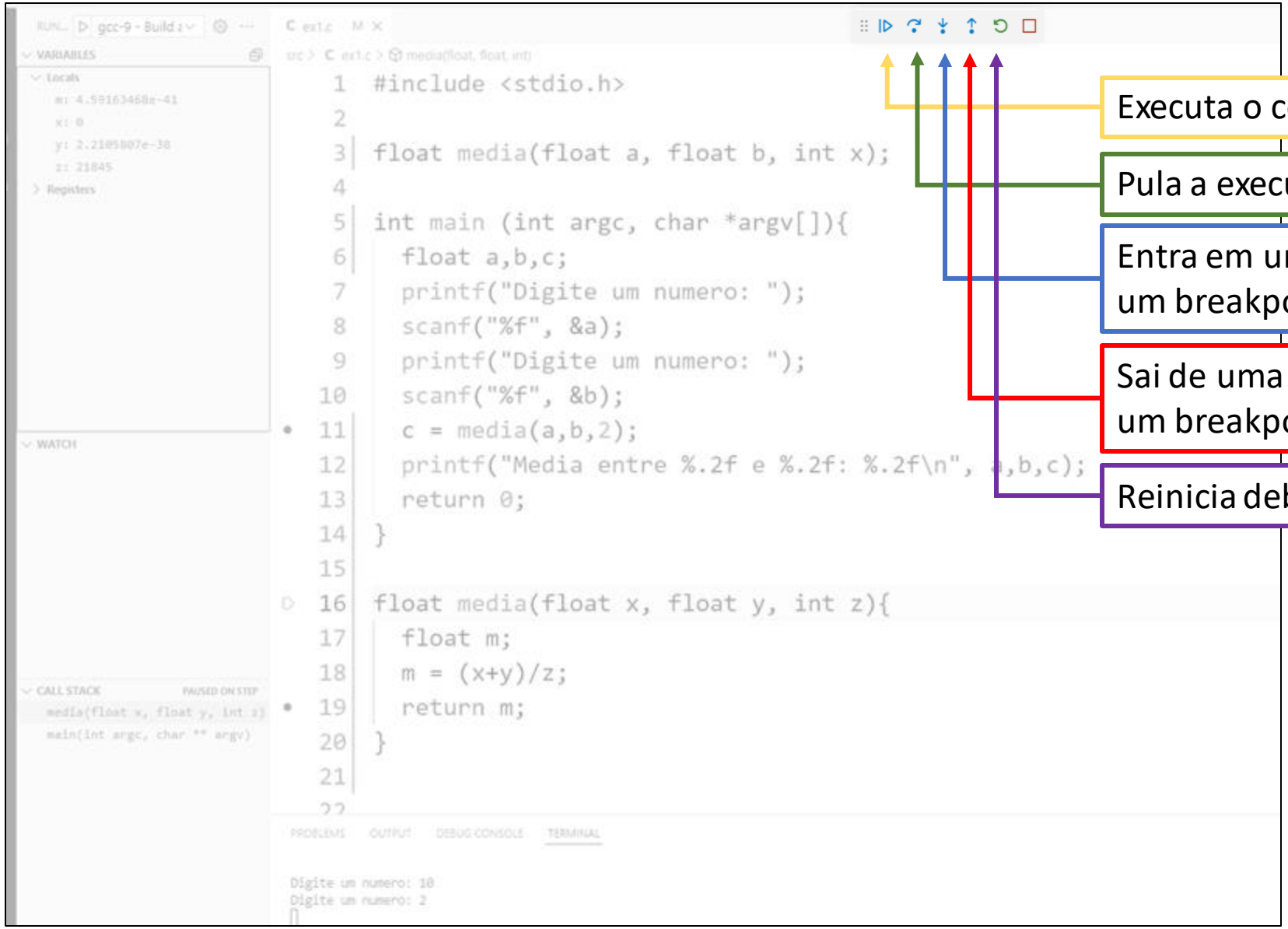
Executa o código até um breakpoint

Pula a execução de um breakpoint

Entra em uma sub-rotina marcada por um breakpoint

Sai de uma sub-rotina marcada por um breakpoint

Depuração de código



Executa o código até um breakpoint

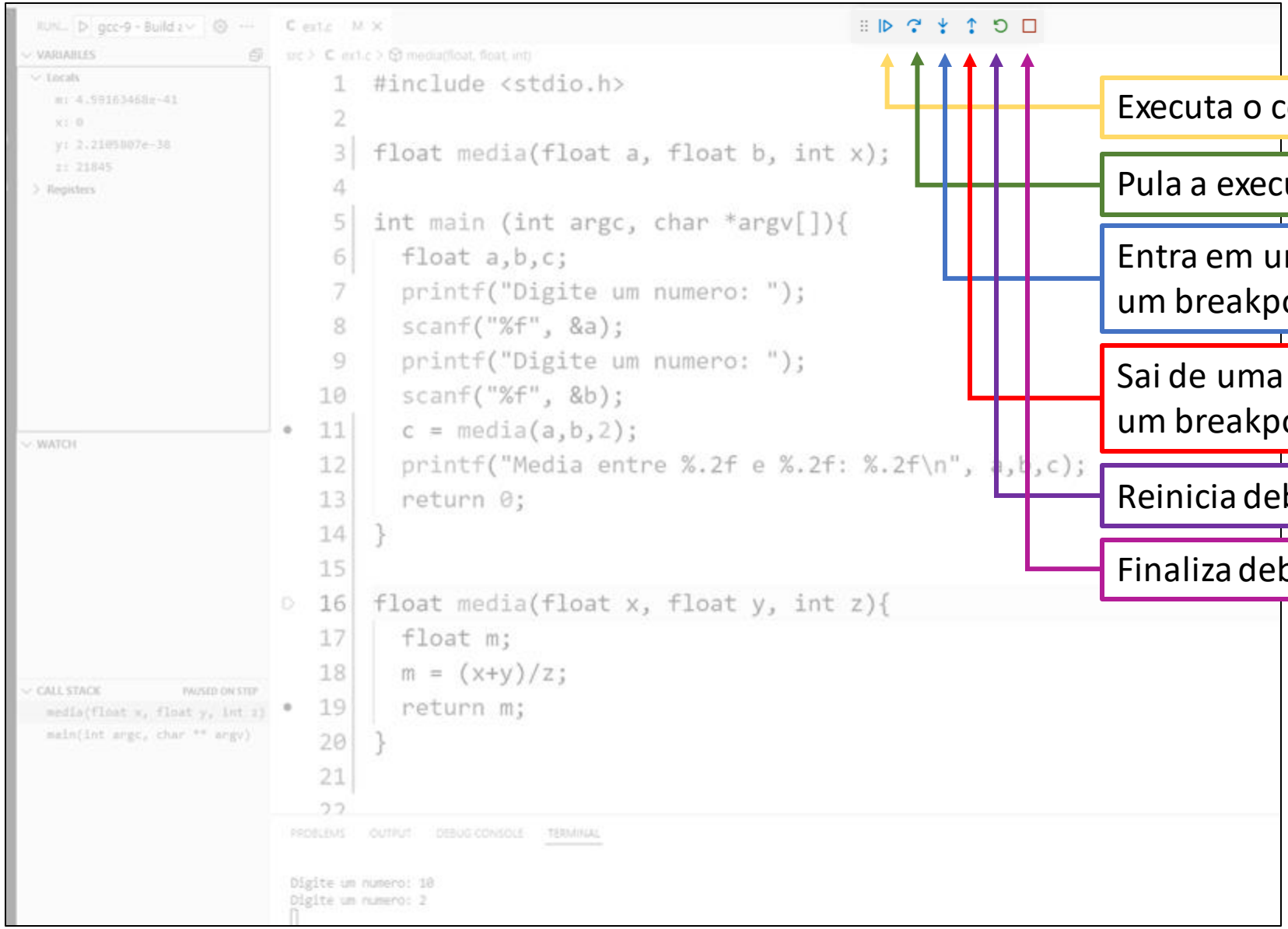
Pula a execução de um breakpoint

Entra em uma sub-rotina marcada por um breakpoint

Sai de uma sub-rotina marcada por um breakpoint

Reinicia debug

Depuração de código



Executa o código até um breakpoint

Pula a execução de um breakpoint

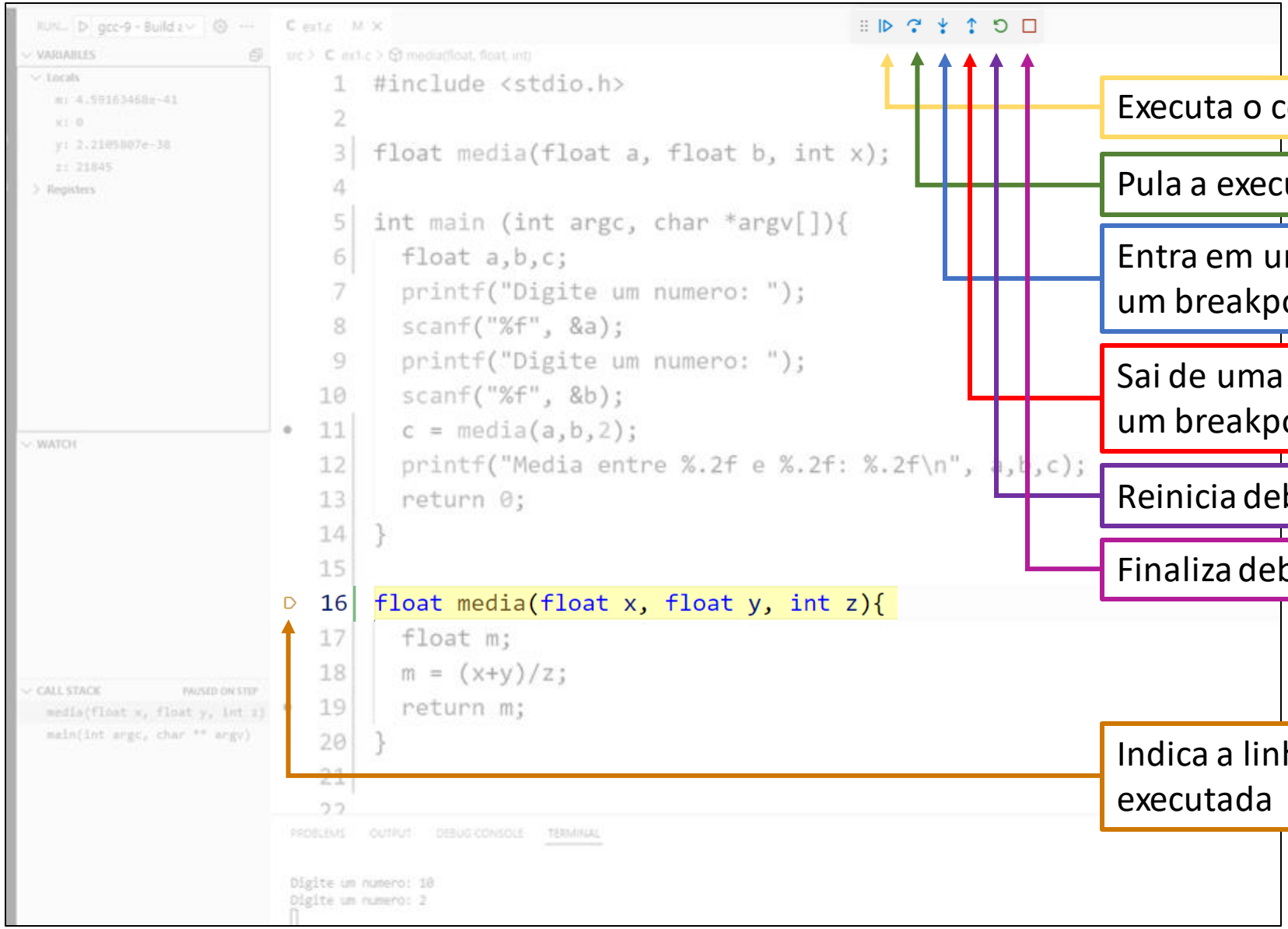
Entra em uma sub-rotina marcada por um breakpoint

Sai de uma sub-rotina marcada por um breakpoint

Reinicia debug

Finaliza debug

Depuração de código



Executa o código até um breakpoint

Pula a execução de um breakpoint

Entra em uma sub-rotina marcada por um breakpoint

Sai de uma sub-rotina marcada por um breakpoint

Reinicia debug

Finaliza debug

Indica a linha que está sendo executada

Lista de Exercícios

- Abra a pasta Downloads no VS Code
  - Utilize o comando git clone https://github.com/ECM404/lista3.git --recursive
  - Entre na pasta utilizando o comando cd lista3
  - Inicialize o diretório com o comando make install