

CURSO BÁSICO

# Programação em C++

**Kaio Christaldo**  
**Fabricio Matsunaga**

# Apresentação

**C++ é uma linguagem de programação de alto desempenho, amplamente utilizada para desenvolvimento de sistemas, jogos, aplicações embarcadas e software de alta eficiência. Derivada do C, ela adiciona recursos de programação orientada a objetos, tornando-a poderosa e flexível. Sua popularidade se deve à combinação de desempenho, controle de hardware e suporte a múltiplos paradigmas de programação.**

# Diferença com a linguagem C

**C++ é uma extensão do C, trazendo suporte à programação orientada a objetos, enquanto C é puramente procedural. Em C++, é possível usar classes, herança e polimorfismo, permitindo um código mais modular e reutilizável. Além disso, C++ oferece recursos como tratamento de exceções, sobrecarga de funções e templates, que não existem em C. Outra diferença importante é o uso de new e delete para alocação dinâmica de memória, substituindo malloc e free do C. Essas melhorias tornam o C++ mais versátil para projetos modernos.**

# Configurando seu Ambiente de Desenvolvimento

**Escolha do compilador**


**Instale um compilador C++ como g++, MinGW ou MSVC.**

**IDEs recomendadas**

**Use Visual Studio Code (com extensão C++), Code::Blocks ou Dev-C++.**

**Crie seu primeiro programa. Compile e execute!**

**Exercício prático: Modifique o programa para exibir seu nome.**



```
1  #include <iostream>
2
3  using namespace std;
4
5  int main() {
6
7      cout << "Hello SBC";
8
9      return 0;
10 }
```

# Conceitos Básicos

## Estrutura Básica de um Programa



```
1  #include <iostream> // Inclusão de Bibliotecas
2
3  using namespace std; // Biblioteca Padrão
4
5  int main() { // Função Principal
6
7      return 0;
8  }
```

# Conceitos Básicos

## Tipos de dados – (Variáveis)

Tipo	Descrição	Exemplo	Limite de Valor
int	Números inteiros (32 bits).	<code>int x = 100;</code>	-2,147,483,648 a 2,147,483,647
long long	Números inteiros grandes (64 bits).	<code>long long x = 10000000000;</code>	-9,223,372,036,854,775,808 a 9,223,372,036,854,775,807
unsigned int	Números inteiros positivos (32 bits).	<code>unsigned int x = 500;</code>	0 a 4,294,967,295
unsigned long long	Números inteiros grandes positivos (64 bits).	<code>unsigned long long x = 10000000000;</code>	0 a 18,446,744,073,709,551,615
float	Números de ponto flutuante (precisão simples).	<code>float pi = 3.14f;</code>	Aproximadamente $\pm 3.4e-38$ a $\pm 3.4e+38$
double	Números de ponto flutuante (precisão dupla).	<code>double pi = 3.14159265358979;</code>	Aproximadamente $\pm 1.7e-308$ a $\pm 1.7e+308$

# Conceitos Básicos

## Tipos de dados – (Variáveis)

Tipo	Descrição	Exemplo	Limite de Valor
char	Um único caractere (1 byte).	<code>char c = 'A';</code>	-128 a 127
bool	Valores booleanos (verdadeiro ou falso).	<code>bool flag = true;</code>	<code>true</code> ou <code>false</code>
array	Conjunto de elementos do mesmo tipo (tamanho fixo).	<code>int arr[1000];</code>	Depende do tamanho do array e da memória disponível
string	Sequência de caracteres (utiliza a biblioteca <code>&lt;string&gt;</code> ).	<code>std::string nome = "João";</code>	Depende da memória disponível

# Conceitos Básicos

## Váriaveis



```
1  int a; // Declara com valor lixo
2  int b = 0; // Declara e inicializa
3  int c{0}; // Não é muito comum
4
5  int d, e = 0, f{0}; //Multi Declaração (Rápido)
6
7  long long g = 1e9; // 1 seguido de 9 zeros;
8
9  double h = 1.600;
10
11  string nome = "Mikhail"; // Uma Frase...
12
13  char letra = 'a'; // Apenas um caracter
14
15  bool isCorrect = false;
```



# Conceitos Básicos

## Entrada e Saída



```
1  int x; // Declara a variável
2  cin >> x; // Escreve
3
4  int x, y; // Declara as variáveis
5  cin >> x, y; // Escreve
```

# Conceitos Básicos

## Entrada e Saída



```
1  string nome, sobrenome;  
2  cin >> nome >> sobrenome; // Digite separado por espaço  
3  
4  string nomeCompleto;  
5  getline(cin, nomeCompleto); // Digite uma Frase...
```

# Conceitos Básicos

## Entrada e Saída



```
1 cout << "Curso de Programação Competitiva" << endl;
```



```
1 cout << sobrenome << ", " << nome << endl;
```

# Conceitos Básicos

## Entrada e Saída



```
1 // #include <iomanip>
2 double pi = 3.141592653589793;
3
4 cout << "Pi com precisão de 5 casas decimais: " << setprecision(5) << pi << endl;
5 cout << "Pi com precisão de 3 casas decimais: " << fixed << setprecision(3) << pi << endl;
6
```

# Conceitos Básicos

## Operadores

- **Aritméticos**
- **Comparativos**

Operador	Descrição	Exemplo
+	Soma	<code>int x = 5 + 3;</code>
-	Subtração	<code>int x = 5 - 3;</code>
*	Multiplicação	<code>int x = 5 * 3;</code>
/	Divisão	<code>int x = 5 / 3;</code>
%	Módulo (resto da divisão)	<code>int x = 5 % 3;</code>
++	Incremento (aumenta o valor da variável em 1)	<code>int x = 5; x++;</code>
--	Decremento (diminui o valor da variável em 1)	<code>int x = 5; x--;</code>
==	Igualdade (verifica se dois valores são iguais)	<code>if (x == 5) {}</code>
!=	Desigualdade (verifica se dois valores são diferentes)	<code>if (x != 5) {}</code>
>	Maior que	<code>if (x &gt; 5) {}</code>
<	Menor que	<code>if (x &lt; 5) {}</code>
>=	Maior ou igual a	<code>if (x &gt;= 5) {}</code>
<=	Menor ou igual a	<code>if (x &lt;= 5) {}</code>
&&	E lógico (AND)	<code>if (x &gt; 5 &amp;&amp; y &lt; 10) {}</code>
**		**
!	Não lógico (NOT)	<code>if (!(x &gt; 5)) {}</code>

# Conceitos Básicos

## Estruturas de Controle: Condicionais



```
1  if (numero % 2 == 0) {  
2      cout << "O Número é Par" << endl;  
3  }  
4  else {  
5      cout << "O Número é Ímpar" << endl;  
6  }
```

# Conceitos Básicos

## Estruturas de Controle: Laços (loops)



```
1 for (int i = 1; i <= 10; i++) {  
2     std::cout << "Número: " << i << std::endl;  
3 }
```



```
1 int i = 1;  
2  
3 while (i <= 10) {  
4     std::cout << "Número: " << i << std::endl;  
5     i++;  
6 }
```

# Conceitos Básicos

**Funções: Modularizando seu Código!**



```
1 void imprimirMsg(string msg) {  
2     std::cout << msg << std::endl;  
3 }
```



# Exercícios

1. **Verificação de Paridade:** Crie um programa que solicite ao usuário um número inteiro e informe se ele é par ou ímpar.
2. **Soma dos Números Pares:** Escreva um programa que peça ao usuário um número inteiro positivo,  $N$ , e calcule a soma de todos os números pares de 0 até  $N$  (inclusive).
3. **Verificação de Número Primo:** Crie um programa que solicite ao usuário um número inteiro positivo,  $N$ , e determine se esse número é primo ou não. Lembre-se de que um número primo é aquele maior que 1 que só é divisível por 1 e por ele mesmo.

# Exercícios do Beecrowd

- 1001 – **Extremamente Básico**
- 1002 – **Área do Círculo**
- 1010 – **Cálculo Simples**
- 1044 – **Múltiplos**
- 1060 – **Números Positivos**
- 1074 – **Par ou Ímpar**
- 1113 – **Jogo de Par ou Ímpar**
- 1164 – **Somando Divisores**
- 1186 – **Área da Pirâmide**
- 1274 – **A Maior Sequência**

# Resolução de Exercícios



**Fim.**

**Referências**