# Definição em C++

Introdução

Em C++, a definição de entidades como variáveis, funções, classes e tipos é essencial para a construção de um programa. A definição especifica não apenas a existência da entidade, mas também suas características e comportamento.

# 1. Definição de Variáveis

A definição de uma variável cria a variável e aloca memória para ela. Pode incluir a inicialização opcional da variável com um valor.

## Exemplo:

int x; // Definição de uma variável inteira

int y = 10; // Definição e inicialização de uma variável inteira

## 2. Definição de Funções

A definição de uma função inclui a especificação do tipo de retorno, nome da função, lista de parâmetros e corpo da função que contém o código a ser executado.

## Exemplo:

int soma(int a, int b) {

return a + b; // Definição de uma função que retorna a soma de dois inteiros

}

**}**;

# 3. Definição de Classes

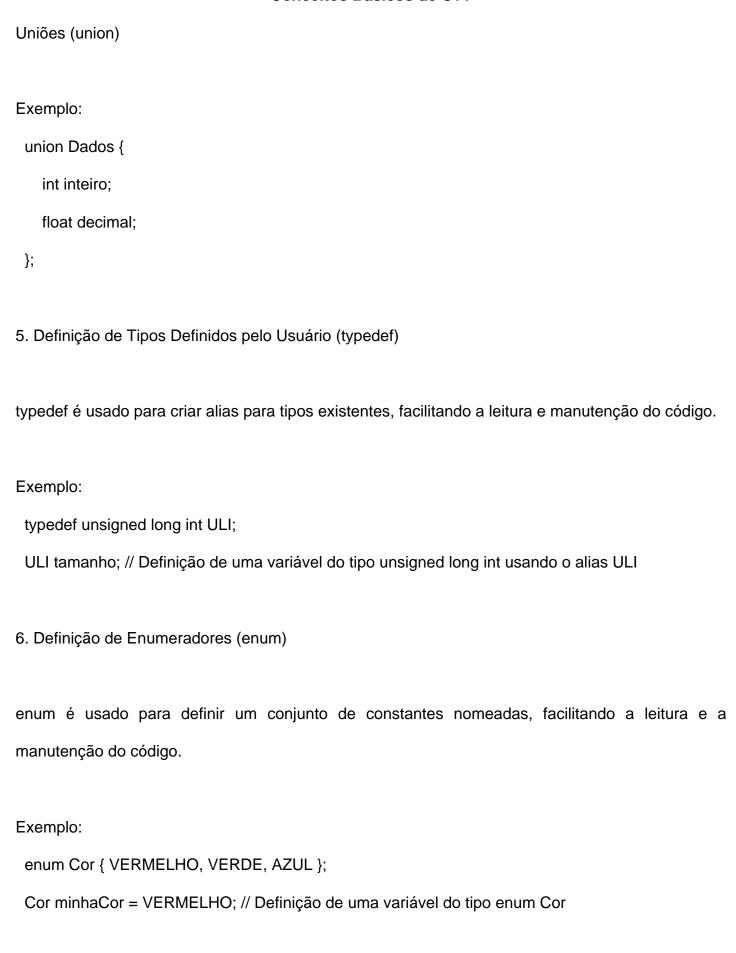
A definição de uma classe especifica as propriedades (variáveis) e métodos (funções) que as instâncias da classe possuirão. Pode incluir modificadores de acesso como public, protected e private.

```
Exemplo:
    class Carro {
    public:
        int velocidade;
        void acelerar() { velocidade++; }
    };

4. Definição de Estruturas e Uniões

Estruturas (struct)

Exemplo:
    struct Pessoa {
        int idade;
        float altura;
```



## 7. Definição de Templates

Templates permitem a definição de classes e funções genéricas, parametrizadas por tipos.

## Exemplo:

```
template<typename T>
T adicionar(T a, T b) {
  return a + b;
}
```

8. Definição de Espaços de Nomes (namespace)

Namespaces são usados para organizar o código e evitar conflitos de nomes entre diferentes partes de um programa ou entre bibliotecas.

# Exemplo:

```
namespace MeuEspaco {
  int x;
  void funcao() {}
```

Dicas de Boas Práticas

- Clareza: Use nomes descritivos para variáveis, funções e classes.
- Modularização: Divida o código em funções e classes pequenas e reutilizáveis.

- Comentários: Comente as definições complexas para facilitar o entendimento.
- Consistência: Mantenha um estilo de codificação consistente em todo o projeto.
- Espaços de Nomes: Use namespaces para organizar o código e evitar conflitos de nomes.

Esta seção abrange os conceitos sobre definição em C++. Para mais detalhes, consulte a documentação oficial: https://en.cppreference.com/w/cpp/language/definition