Funções em C++

Introdução

Funções em C++ são blocos de código que realizam tarefas específicas e podem ser reutilizados ao longo do programa. Elas podem receber parâmetros, retornar valores e são fundamentais para a organização e modularização do código.

1. Definição e Sintaxe

- Definição: Uma função é definida por um tipo de retorno, um nome, uma lista de parâmetros (opcional) e um corpo.
- Sintaxe:

```
TipoDeRetorno NomeDaFuncao(TipoParametro1 param1, TipoParametro2 param2) {
    // Corpo da função
    return valor;
}
```

2. Exemplo de Definição de Função

- Exemplo:

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

void imprimeMensagem() {
```

```
    std::cout << "Olá, mundo!" << std::endl;
}
```

3. Funções com Parâmetros

- Definição: Funções podem receber parâmetros para realizar operações com base nos valores fornecidos.
- Exemplo:

```
double multiplica(double x, double y) {
    return x * y;
}
```

4. Funções sem Parâmetros

- Definição: Funções podem ser definidas sem parâmetros, realizando operações independentes de valores de entrada.
- Exemplo:

```
void mostrarDataAtual() {
    // Código para mostrar a data atual
}
```

5. Funções com Retorno de Valor

- Definição: Funções podem retornar valores para o chamador utilizando a palavra-chave `return`.
- Exemplo:

```
int quadrado(int n) {
```

```
    return n * n;
}
```

6. Funções sem Retorno (void)

- Definição: Funções que não retornam valores utilizam o tipo de retorno `void`.
- Exemplo:

```
void saudacao() {
    std::cout << "Bem-vindo!" << std::endl;
}
```

7. Funções Inline

- Definição: Funções `inline` sugerem ao compilador que substitua a chamada da função pelo seu corpo, para evitar o overhead da chamada.
- Exemplo:

```
inline int cubo(int x) {
    return x * x * x;
}
```

8. Funções Recursivas

- Definição: Funções recursivas são aquelas que chamam a si mesmas para resolver problemas através da divisão em subproblemas menores.
- Exemplo:

```
int fatorial(int n) {
```

```
    if (n <= 1) return 1;
    else return n * fatorial(n - 1);
}
```

9. Sobrecarga de Funções

- Definição: Funções podem ser sobrecarregadas, ou seja, múltiplas funções podem ter o mesmo nome, mas com diferentes listas de parâmetros.
- Exemplo:

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

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

10. Funções Membros de Classe

- Definição: Funções podem ser membros de classes, operando sobre os dados da instância da classe.
- Exemplo:

```
class Retangulo {
public:
    int largura, altura;
    int area() {
```

```
        return largura * altura;
    }
};
```

11. Funções Lambda

- Definição: Funções lambda são funções anônimas que podem capturar variáveis do escopo circundante.
- Exemplo:

```
auto soma = [](int a, int b) { return a + b; };
std::cout << soma(3, 4) << std::endl;  // Saída: 7
```

Dicas de Boas Práticas

- Modularização: Divida o código em funções pequenas e coesas para melhorar a legibilidade e a manutenção.
- Nomes Descritivos: Use nomes de função descritivos que indiquem claramente o propósito da função.
- Documentação: Documente as funções, especialmente os parâmetros e o valor de retorno, para facilitar o entendimento e a manutenção do código.

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