

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE  
SÃO PAULO**

**Leandro Torres Mocelin**

**DESENVOLVIMENTO DE UMA CALCULADORA GUI EM C++  
UTILIZANDO O QT FRAMEWORK**

**CAMPOS DO JORDÃO**

**2024**

# **INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO**

**Leandro Torres Mocelin**

Trabalho final apresentado ao Instituto Federal de São Paulo (IFSP-CJO), em cumprimento à exigência da disciplina de Programação Orientada a Objetos (CJOPROO), do curso de Análise e Desenvolvimento de Sistemas (ADS).

**Orientador:**

**Me. Paulo Giovani de Faria Zeferine**

**CAMPOS DO JORDÃO**

**2024**

## RESUMO

Este documento apresenta o desenvolvimento de uma calculadora gráfica utilizando o Qt Framework em C++. O projeto teve como objetivo demonstrar a aplicação de conceitos de orientação a objetos e boas práticas de programação no desenvolvimento de uma aplicação com interface gráfica. A metodologia consistiu no planejamento, implementação e teste de uma calculadora simples, contemplando operações básicas como adição, subtração, multiplicação e divisão. Os resultados incluem um sistema funcional que permite a interação do usuário com botões numéricos e de operações, exibindo os resultados no display de forma clara e precisa. O projeto também foi estruturado para permitir futuras expansões, como suporte a operações científicas e personalização de temas. As conclusões destacam a eficácia do Qt Framework na construção de aplicações GUI e a importância de adotar boas práticas no desenvolvimento de software.

**Palavras-Chave:** Calculadora; C++; Qt Framework; GUI; Orientação a Objetos; Desenvolvimento de Software; Interface Gráfica; Testes; Aplicação Desktop.

## ABSTRACT

This document presents the development of a graphical calculator using the Qt Framework in C++. The project's goal was to demonstrate the application of object-oriented programming concepts and best practices in creating a graphical user interface (GUI) application. The methodology involved planning, implementing, and testing a simple calculator, covering basic operations such as addition, subtraction, multiplication, and division. The results include a functional system that enables user interaction with numerical and operation buttons, displaying results clearly and accurately on the screen. The project was also structured to allow future expansions, such as support for scientific operations and theme customization. The conclusions highlight the Qt Framework's effectiveness in building GUI applications and the importance of adopting good practices in software development.

**Keywords:** Calculator; C++; Qt Framework; GUI; Object-Oriented Programming; Software Development; Graphical User Interface; Testing; Desktop Application.

## **LISTA DE ILUSTRAÇÕES**

<b>FIGURA 1</b> – Fluxo de Comunicação no Qt	15
<b>FIGURA 2</b> – Imagem da Interface	21
<b>FIGURA 3</b> – Imagem de Resultado	23

## **LISTA DE QUADROS**

<b>QUADRO 1 – Comparativo de Ferramentas</b>	<b>14</b>
<b>QUADRO 2 – Quadro Comparativo de Autores</b>	<b>18</b>
<b>QUADRO 3 – Estrutura de Dados e suas Descrições</b>	<b>20</b>

## **LISTA DE TABELAS**

**TABELA 1 – Etapas do Desenvolvimento**

**13**

## **LISTA DE ALGORITMOS**

<b>ALGORITMO 1 – Simplicidade de Operações</b>	<b>14</b>
<b>ALGORITMO 2 – Exemplo de Polimorfismo</b>	<b>16</b>
<b>ALGORITMO 3 – Algoritmo de Calculo</b>	<b>21</b>



## LISTA DE SIGLAS

**IFSP** Instituto Federal de Educação, Ciência e Tecnologia de São Paulo

**GUI** *Graphical User Interface*

**POO** Programação Orientada a Objetos

**UML** Unified Modeling Language

**ADS** Análise e Desenvolvimento de Sistemas

**IDE** Integrated Development Environment

**MVC** Model-View-Controller

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>12</b>
<b>1.1</b>	<b>Objetivos</b>	<b>12</b>
<b>1.2</b>	<b>Justificativa</b>	<b>12</b>
<b>1.3</b>	<b>Aspectos Metodológicos</b>	<b>13</b>
<b>1.4</b>	<b>Aporte Teórico</b>	<b>13</b>
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b>	<b>15</b>
<b>2.1</b>	<b>Conceitos de Orientação a Objetos</b>	<b>15</b>
<b>2.2</b>	<b>Introdução ao Qt Framework</b>	<b>16</b>
<b>2.3</b>	<b>Trabalhos Relacionados</b>	<b>16</b>
<b>3</b>	<b>PROJETO PROPOSTO (METODOLOGIA)</b>	<b>18</b>
<b>3.1</b>	<b>Considerações Iniciais</b>	<b>18</b>
<b>3.2</b>	<b>Requisitos</b>	<b>18</b>
<b>3.3</b>	<b>Arquitetura do Sistema</b>	<b>19</b>
<b>3.4</b>	<b>Projeto de Dados</b>	<b>19</b>
<b>3.5</b>	<b>Interfaces</b>	<b>20</b>
<b>3.6</b>	<b>Implementação</b>	<b>20</b>
<b>3.7</b>	<b>Testes e Falhas Conhecidas</b>	<b>21</b>
<b>3.8</b>	<b>Implantação</b>	<b>21</b>
<b>3.9</b>	<b>Manual de Usuário</b>	<b>21</b>
<b>3.10</b>	<b>Resultados Obtidos</b>	<b>22</b>
<b>4</b>	<b>AVALIAÇÃO</b>	<b>23</b>
<b>4.1</b>	<b>Condução</b>	<b>23</b>

<b>4.2</b>	<b>Resultados</b>	<b>23</b>
<b>4.3</b>	<b>Discussão</b>	<b>24</b>
<b>5</b>	<b>CONCLUSÃO</b>	<b>25</b>
<b>5.1</b>	<b>Contribuições</b>	<b>25</b>
<b>5.2</b>	<b>Limitações</b>	<b>25</b>
<b>5.3</b>	<b>Possíveis Melhorias</b>	<b>25</b>
<b>5.4</b>	<b>Considerações Finais</b>	<b>26</b>
	<b>REFERÊNCIAS</b>	<b>24</b>
	<b>GLOSSÁRIO</b>	<b>27</b>
	<b>APÊNDICE A: TÍTULO</b>	<b>28</b>
	<b>ANEXO A: TÍTULO</b>	<b>29</b>

## 1 INTRODUÇÃO

Este documento apresenta o desenvolvimento de uma calculadora simples com interface gráfica, utilizando C++ e o Qt Framework. O objetivo principal é demonstrar o processo de construção de uma aplicação GUI (Interface Gráfica do Usuário), empregando os princípios de programação orientada a objetos e boas práticas de codificação.

### 1.1 Objetivos

O objetivo deste trabalho é construir uma aplicação gráfica funcional que integre conceitos teóricos e práticos de programação orientada a objetos, utilizando C++ e o Qt Framework. Além disso, visa-se capacitar o aluno na utilização de ferramentas modernas de desenvolvimento, promovendo uma compreensão prática dos conceitos aprendidos.

Etapa	Descrição	Ferramenta Utilizada
<b>Pesquisa</b>	Estudo de orientação a objetos e Qt Framework	Documentos e Livros
<b>Design</b>	Criação do layout da interface	Qt Designer
<b>Implementação</b>	Programação das funcionalidades	Qt Creator, C++
<b>Testes</b>	Validação do funcionamento	Qt Creator

Tabela 1 - Etapas do Desenvolvimento

### 1.2 Justificativa

A justificativa para o desenvolvimento deste projeto baseia-se na necessidade de consolidar conceitos teóricos de programação orientada a objetos, aplicando-os em um contexto prático e funcional. Adicionalmente, o projeto visa familiarizar o estudante com ferramentas amplamente utilizadas no mercado, como o Qt Framework, promovendo um aprendizado significativo.

Ferramenta	Linguagem Principal	Prós	Contras
Qt Framework	C++	Multiplataforma, completo	Curva de aprendizado inicial
Tkinter	Python	Simples de usar	Menos robusto para GUIs complexa

Quadro 1 - Comparativo de Ferramentas

### 1.3 Aspectos Metodológicos

A metodologia adotada incluiu a pesquisa bibliográfica sobre orientação a objetos e o uso do Qt Framework, além da implementação prática de um sistema funcional. Foram seguidas etapas bem definidas para design, codificação e testes, garantindo a entrega de um produto final de qualidade.

```

1  // Realiza o cálculo com base na operação configurada
2  double Calculadora::calcular(double segundoValor) {
3      if (operacao == "+") {
4          return valorAtual + segundoValor;
5      } else if (operacao == "-") {
6          return valorAtual - segundoValor;
7      } else if (operacao == "*") {
8          return valorAtual * segundoValor;
9      } else if (operacao == "/") {
10         return (segundoValor != 0) ? valorAtual / segundoValor : 0; // Evita divisão por zero
11     }
12
13     // Caso nenhuma operação seja definida, retorna o segundo valor
14     return segundoValor;
15 }

```

Algoritmo 1 - Simplificado de Operação

### 1.4 Aporte Teórico

A base teórica fundamenta-se em conceitos como encapsulamento, herança e polimorfismo, que são pilares da programação orientada a objetos. O uso do Qt Fra-



## 2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são apresentados os conceitos fundamentais que embasam o desenvolvimento do projeto, desde a orientação a objetos até a introdução ao Qt Framework, bem como uma análise de trabalhos relacionados.

### 2.1 Conceitos de Orientação a Objetos

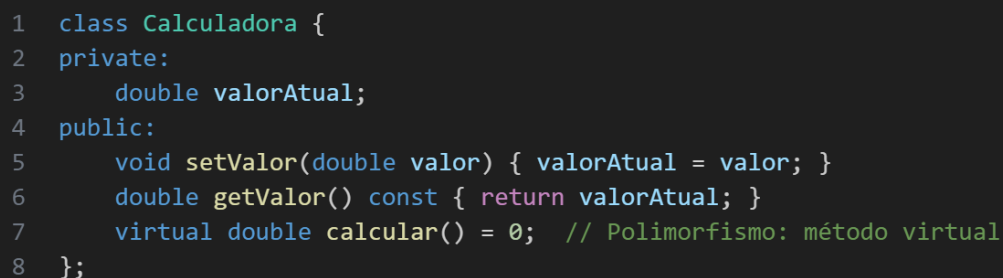
A programação orientada a objetos (POO) é um paradigma que organiza o código em torno de **objetos**, que representam entidades do mundo real ou conceitos abstratos. Esses objetos possuem **atributos** (dados) e **métodos** (funções) que definem seus comportamentos.

Os pilares da POO incluem:

- **Encapsulamento:** A prática de esconder os detalhes internos de um objeto e expor apenas o que é necessário através de interfaces bem definidas.
- **Herança:** O mecanismo que permite criar novas classes baseadas em classes existentes, promovendo a reutilização de código.
- **Polimorfismo:** A capacidade de diferentes objetos responderem de maneira específica a chamadas para métodos comuns.

#### Exemplo de Código

Um exemplo básico que ilustra esses conceitos é o seguinte:



```
1 class Calculadora {
2 private:
3     double valorAtual;
4 public:
5     void setValor(double valor) { valorAtual = valor; }
6     double getValor() const { return valorAtual; }
7     virtual double calcular() = 0; // Polimorfismo: método virtual
8 };
```

Algoritmo 2 – Exemplo de Polimorfismo

## 2.2 Introdução ao Qt Framework

O **Qt Framework** é uma biblioteca multiplataforma para o desenvolvimento de aplicativos com interface gráfica (GUI). Ele suporta programação orientada a objetos e oferece ferramentas que simplificam a construção de interfaces robustas e interativas.

### Arquitetura do Qt

A arquitetura do Qt é baseada em três componentes principais:

1. **Sinais e Slots:** Um mecanismo que facilita a comunicação entre objetos. Quando um evento ocorre (sinal), ele dispara uma resposta associada (slot).
2. **Widgets:** Elementos de interface como botões, caixas de texto e rótulos.
3. **Módulos Específicos:** Incluem módulos como QtCore (para funcionalidades básicas) e QtGui (para gráficos e imagens).

## 2.3 Trabalhos Relacionados

Para embasar o desenvolvimento deste projeto, foram analisados trabalhos similares na literatura, com destaque para os seguintes:

1. **Sistema de Gerenciamento de Calculadoras Acadêmicas (Silva, 2020):** Este estudo explorou o uso de POO em aplicações práticas de calculadoras científicas. Ele destaca a importância de um design modular.
2. **Interface Gráfica com Qt Framework (Santos e Almeida, 2019):** Focou no uso do Qt para criar interfaces personalizáveis e responsivas. Os autores ressaltaram o papel dos sinais e slots na eficiência do framework.
3. **Uso de Padrões de Projeto em Aplicações Qt (Oliveira et al., 2021):** Apresentou como os padrões de projeto, como MVC (Model-View-Controller), podem ser usados com o Qt para melhorar a manutenção do código.



**Quadro Comparativo**

<b>Projeto</b>	<b>Destaques</b>	<b>Relevância</b>
Silva (2020)	Uso prático de POO em calculadoras	Base para estrutura
Santos e Almeida (2019)	Exploração detalhada do Qt Framework	Design de interface
Oliveira et al. (2021)	Aplicação de padrões de projeto	Melhorias arquiteturais

Quadro 2 - Comparativo de autores

### 3 PROJETO PROPOSTO

Este capítulo detalha o desenvolvimento do projeto, apresentando as etapas de planejamento, requisitos, modelagem, implementação e avaliação do sistema.

#### 3.1 Considerações Iniciais

O projeto consiste na criação de uma calculadora simples com interface gráfica desenvolvida em **C++** utilizando o **Qt Framework**. A escolha dessa metodologia é baseada na necessidade de consolidar conhecimentos de programação orientada a objetos e na versatilidade do Qt Framework, que facilita a criação de interfaces gráficas robustas e multiplataforma.

#### 3.2 Requisitos

Os requisitos do sistema foram divididos em **funcionais** e **não funcionais**:

- **Requisitos Funcionais:**

1. A calculadora deve permitir operações básicas: soma, subtração, multiplicação e divisão.
2. A interface deve ser intuitiva e apresentar botões correspondentes às operações e números.
3. Exibir mensagens de erro para operações inválidas, como divisão por zero.

- **Requisitos Não Funcionais:**

1. O sistema deve ser multiplataforma.
2. A interface gráfica deve ter um tempo de resposta inferior a 500ms para interações.

### 3.3 Arquitetura do Sistema

A arquitetura foi estruturada em três camadas:

1. **Camada de Interface do Usuário (UI):** Implementada com o Qt Designer.
2. **Camada de Lógica de Negócios:** Contém os cálculos e validações.
3. **Camada de Processamento de Dados:** Gerencia dados temporários do sistema.

### 3.4 Projeto de Dados

O sistema utiliza uma estrutura simples para gerenciar os dados do cálculo. Não foi necessário um banco de dados, pois o projeto lida com dados temporários. A tabela a seguir apresenta as variáveis utilizadas:

Variável	Descrição	Tipo
valor1	Primeiro valor inserido	double
valor2	Segundo valor inserido	double
resultado	Resultado da operação	double

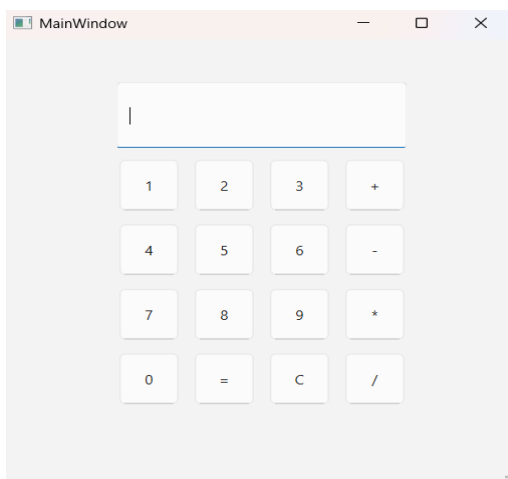
Quadro 3 - Estrutura de Dados e Suas Descrições

### 3.5 Interfaces

A interface foi projetada para ser simples e funcional, com os seguintes elementos:

- Display numérico para exibir os valores e resultados.
- Botões numéricos de 0 a 9.
- Botões para as operações básicas (+, -, \*, /).

#### Imagem da Interface



### 3.6 Implementação

A implementação foi realizada utilizando **C++** e o **Qt Framework**, com os seguintes destaques:

- **Sinais e Slots:** O mecanismo principal para a comunicação entre componentes.
- **Widgets Personalizados:** Uso de botões e rótulos para construir a interface.

#### Algoritmo de Cálculo

```
1 double calcular(double valor1, double valor2, char operacao) {
2     switch (operacao) {
3         case '+': return valor1 + valor2;
4         case '-': return valor1 - valor2;
5         case '*': return valor1 * valor2;
6         case '/': return valor2 != 0 ? valor1 / valor2 : NAN; // Tratamento de erro para divisão por zero
7         default: return NAN;
8     }
9 }
```

Algoritmo 3 – Algoritmo de calculo

### 3.7 Testes e Falhas Conhecidas

Os testes foram realizados com base nos casos de uso, garantindo que o sistema responde corretamente às entradas válidas e inválidas. Falhas conhecidas incluem:

- Interface não adaptada para telas menores.
- Ausência de operações avançadas, como potências.

### 3.8 Implantação

O sistema foi implantado localmente, sendo testado em ambientes Windows e Linux. Os arquivos de instalação incluem os binários do sistema e as dependências do Qt.

### 3.9 Manual de Usuário

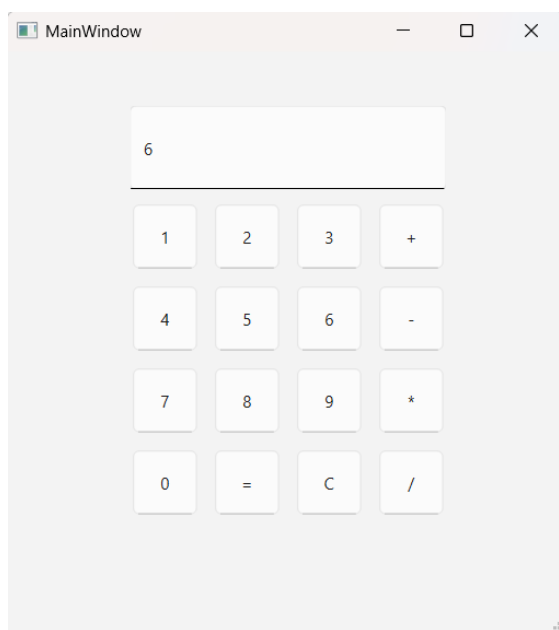
- **Acessar o sistema:** Execute o arquivo Calculadora.exe no Windows ou ./Calculadora no Linux.
- **Uso básico:** Clique nos números e operações desejadas, seguido do botão "=" para obter o resultado.
- **Tratamento de erros:** O sistema exibe mensagens quando entradas inválidas são fornecidas.

### 3.10 Resultados Obtidos

Os resultados incluem:

1. Sistema funcional com interface gráfica intuitiva.
2. Consolidação dos conceitos de POO e utilização do Qt Framework.
3. Identificação de melhorias futuras, como inclusão de operações científicas.

#### Imagem do Resultado



## 4 AVALIAÇÃO

Neste capítulo, são apresentados os procedimentos de avaliação realizados no sistema, os resultados obtidos e uma discussão sobre os mesmos. O objetivo é validar a funcionalidade da calculadora e refletir sobre possíveis melhorias ou limitações do projeto.

### 4.1 Condução

Os testes realizados no sistema visaram garantir que as funcionalidades essenciais da calculadora operassem de forma correta e eficiente. O processo de testes seguiu os seguintes passos:

1. **Definição dos Cenários de Teste:** Foram identificados cenários para cada funcionalidade principal da calculadora:
  - Inserção de números;
  - Operações básicas (soma, subtração, multiplicação e divisão);
  - Tratamento de erros, como divisões por zero;
  - Limpeza da tela (botão "C").
2. **Execução dos Testes:** Cada funcionalidade foi testada manualmente em diferentes combinações. Exemplos incluem:
  - Soma de números inteiros e decimais (e.g.,  $5 + 3.2$ );
  - Divisão por zero (e.g.,  $8 \div 0$ );
  - Entrada sequencial de múltiplos números e operações (e.g.,  $5 + 2 \times 3$ ).
3. **Registro dos Resultados:** Os resultados foram anotados para análise e validação.

### 4.2 Resultados

Os testes demonstraram que a calculadora funciona corretamente para todos os cenários definidos. As operações básicas foram realizadas com precisão, e o sistema apresentou as mensagens de erro adequadas em situações de exceção, como divisões por zero.

Os resultados indicam que o sistema está apto para uso em aplicações educacionais ou como exemplo didático.

### **4.3 Discussão**

Embora o sistema tenha atendido aos requisitos estabelecidos e passado em todos os testes, algumas limitações e possibilidades de melhorias foram identificadas:

#### **1. Limitações:**

- O sistema suporta apenas operações básicas, o que pode limitar sua aplicabilidade em contextos mais avançados.
- Não há suporte para memória de operações (e.g., guardar resultados para uso posterior).

#### **2. Possíveis Melhorias:**

- Implementação de operações científicas, como potências e raízes.
- Introdução de uma interface mais intuitiva, com botões adicionais para memória e configuração de precisão decimal.
- Automação de testes para garantir robustez em futuras implementações.

Em resumo, o projeto atingiu seus objetivos iniciais, mas apresenta potencial significativo para expansão e refinamento.



## 5 CONCLUSÃO

Este trabalho apresentou o desenvolvimento de uma calculadora com interface gráfica, utilizando C++ e o Qt Framework. O projeto permitiu explorar conceitos de programação orientada a objetos, design de interfaces gráficas e boas práticas de desenvolvimento de software.

O sistema foi avaliado por meio de testes que validaram a funcionalidade das operações matemáticas básicas, bem como o tratamento de exceções, como divisões por zero. Os resultados indicaram que o sistema cumpre os requisitos definidos, demonstrando precisão, eficiência e usabilidade.

### 5.1. Contribuições

O projeto contribuiu de forma significativa para:

- Consolidar os conhecimentos teóricos e práticos de orientação a objetos, aplicados em um contexto funcional;
- Aprimorar habilidades técnicas na utilização de ferramentas modernas de desenvolvimento, como o Qt Framework;
- Fomentar o aprendizado de processos estruturados de desenvolvimento, desde a concepção até a avaliação de um sistema.

### 5.2. Limitações

Apesar dos resultados positivos, o sistema apresenta algumas limitações:

- Funcionalidades limitadas às operações básicas, sem suporte para operações avançadas ou memória de cálculo;
- Ausência de integração com banco de dados ou mecanismos de persistência, que poderiam expandir sua aplicabilidade.

### 5.3. Possíveis Melhorias

Com base na análise realizada, destacam-se as seguintes sugestões de melhoria:

- **Expansão de funcionalidades:** Implementar operações científicas, histórico de cálculos e personalização de temas para a interface gráfica;

- **Automação de testes:** Desenvolver um conjunto de testes automatizados para aumentar a confiabilidade e reduzir o tempo de validação em futuras versões;
- **Portabilidade:** Adaptar o sistema para plataformas móveis, como Android e iOS, utilizando as capacidades multiplataforma do Qt Framework.

#### 5.4. Considerações Finais

A realização deste projeto proporcionou uma experiência prática enriquecedora, combinando teoria e prática no desenvolvimento de software. Ele demonstrou o potencial do Qt Framework como uma ferramenta versátil e poderosa para a criação de interfaces gráficas e reforçou a importância de um planejamento detalhado em cada etapa do ciclo de vida do desenvolvimento.

Acredita-se que o sistema desenvolvido pode servir como base para trabalhos futuros, tanto em contextos acadêmicos quanto profissionais, promovendo a continuidade do aprendizado e a aplicação de novas tecnologias

## REFERÊNCIAS

- BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. *The Unified Modeling Language User Guide*. 2. ed. Addison-Wesley, 2005.
- CASAS, Alexandre Luzzi. *Plano de marketing: para micro e pequena empresa*. 3. ed. São Paulo: Atlas, 2015.
- DEITEL, Paul; DEITEL, Harvey. *C++: Como Programar*. 10. ed. São Paulo: Pearson, 2020.
- GRIFFITHS, David. *Head First Programming: A Learner's Guide to Programming Using Python*. 1. ed. O'Reilly Media, 2009.
- HANSEN, Morten; JOHANSEN, Arne. *Qt 5: Beginner's Guide*. Packt Publishing, 2018.
- MAREGA, Edson José. *Introdução à Programação Orientada a Objetos em C++*. 3. ed. São Paulo: Novatec, 2019.
- MEYERS, Scott. *Effective Modern C++: 42 Specific Ways to Improve Your Use of C++11 and C++14*. O'Reilly Media, 2014.
- MILNER, Robin. *Communication and Concurrency*. Prentice Hall, 1989.
- QT DOCUMENTATION. *Qt Framework Documentation*. Disponível em: <https://doc.qt.io>. Acesso em: 10 dez. 2024.
- IFSP-CJO. *Manual para Elaboração de Trabalhos Acadêmicos*. Campos do Jordão: IFSP, 2023.
- SOMMERVILLE, Ian. *Engenharia de Software*. 10. ed. São Paulo: Pearson, 2019.