INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO

CRISTIANO DE ALMEIDA TOMAZ

DISCIPLINA: PROGRAMAÇÃO ORIENTADA A OBJETOS

PROFESSOR: PAULO GIOVANI DE FARIA ZEFERINO

CURSO: ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JOGO: JORNADA IFSP

CAMPOS DO JORDÃO

2024

RESUMO

Este relatório apresenta o desenvolvimento de um jogo 2D de tiro horizontal, criado como projeto da disciplina de Programação Orientada a Objetos no curso de Análise e Desenvolvimento de Sistemas. O jogo, intitulado "Jornada IFSP: Atire nos Obstáculos", foi implementado em C++ utilizando a biblioteca gráfica Raylib. Nele, o jogador controla um personagem que se movimenta pela tela e dispara tiros de estilingue para destruir inimigos que se aproximam. Em analogia a um estudante do Instituto Federal, que precisa vencer diversos obstáculos para permanecer no curso. A estrutura do projeto segue os princípios da programação orientada a objetos, com classes como Player, Projectile, Obstacle e GameObject, permitindo modularidade e reaproveitamento de código. Foram utilizados recursos gráficos e sonoros para tornar a experiência mais imersiva, além da aplicação de lógica para colisões, pontuação e reinício do jogo. Este projeto proporcionou o desenvolvimento de habilidades práticas na construção de jogos com C++ e Raylib, reforçando conceitos fundamentais da orientação a objetos.

Palavras-Chave: jogo, raylib, obstáculos, estudante, inimigos, orientação a objetos.

ABSTRACT

This report presents the development of a 2D side-scrolling shooting game, created as a project for the Object-Oriented Programming course in the Systems Analysis and Development program. The game, titled "Journey IFSP: Shoot the Obstacles", was implemented in C++ using the Raylib graphics library. In the game, the player controls a character who moves across the screen and shoots projectiles to destroy approaching enemies. The gameplay serves as an analogy to a student at the Federal Institute facing various challenges to stay on course. The project structure follows object-oriented programming principles, with modular and reusable classes such as Player, Projectile, Obstacle, and GameObject. Visual and audio elements were incorporated to create a more immersive experience, along with logic for collision detection, scoring, and game restart. This project fostered hands-on skills in game development with C++ and Raylib, while reinforcing key object-oriented concepts.

Keywords: game, raylib, obstacles, student, enemies, object orientation.

SUMÁRIO

1 INTRODUÇÃO

A Programação Orientada a Objetos (POO) é um dos paradigmas mais utilizados no desenvolvimento de software moderno, por permitir uma organização modular, reutilização de código e maior facilidade na manutenção de sistemas. No contexto da disciplina de Programação Orientada a Objetos, do curso de Análise e Desenvolvimento de Sistemas, este projeto foi proposto com o intuito de aplicar de forma prática os conceitos aprendidos em sala de aula por meio da criação de um jogo digital.

O jogo desenvolvido tem como temática a "Jornada IFSP: Atire nos Obstáculos", na qual o jogador controla um personagem que deve se mover pela tela e eliminar obstáculos que surgem em seu caminho, representados por diferentes inimigos. Para isso, foram implementadas mecânicas como movimentação, disparo de projéteis, detecção de colisões, sistema de pontuação, e sons de fundo e de impacto.

No contexto lúdico, o jogo mostra os obstáculos que o estudante geralmente precisa enfrentar ao ingressar em algum curso do Instituto Federal de Campos do Jordão. Entre os obstáculos propostos estão os personagens criados e denominados: “Frio”, “Professor Chato”, “Ansiedade”, “Prova” e “Sono”.

Além de reforçar os conceitos fundamentais da POO, como encapsulamento, herança e polimorfismo, o projeto também buscou estimular o raciocínio lógico, a capacidade de abstração e o trabalho com bibliotecas gráficas externas, no caso, a Raylib, utilizada para lidar com imagens, sons e entrada de dados.

Este relatório descreve detalhadamente o desenvolvimento do projeto, suas motivações, a estrutura adotada e os resultados obtidos, evidenciando como os conceitos teóricos da disciplina foram aplicados em um contexto prático e divertido.

1.1 Objetivos

O principal objetivo deste projeto foi aplicar os conceitos de Programação Orientada a Objetos na construção de um jogo digital 2D, utilizando a linguagem C++ e a biblioteca Raylib. Para alcançar esse propósito, foram estabelecidos os seguintes objetivos específicos:

- Implementar um jogo funcional no qual o jogador possa controlar um personagem que se move na tela e atira projéteis contra obstáculos inimigos.
- Modelar as entidades do jogo utilizando classes, aplicando os pilares da orientação a objetos como herança, encapsulamento e polimorfismo.
- Gerenciar recursos gráficos e sonoros, como sprites, efeitos sonoros e música de fundo, para proporcionar uma experiência interativa.
- Detectar colisões entre objetos, como projéteis e inimigos, e entre o jogador e os obstáculos, aplicando lógica condicional e controle de estado.
- Permitir reinício do jogo após o término, oferecendo ao usuário uma opção para continuar jogando sem encerrar a aplicação.
- Desenvolver habilidades práticas em C++ e no uso de bibliotecas externas voltadas para gráficos e jogos, reforçando o aprendizado da disciplina.

1.2 Justificativa

O desenvolvimento deste projeto justifica-se como uma oportunidade prática de consolidar os conhecimentos adquiridos na disciplina de Programação Orientada a Objetos (POO), por meio da criação de um jogo simples, mas funcional, que demanda o uso efetivo de classes, objetos, herança, polimorfismo e encapsulamento.

Além da aplicação direta dos conceitos teóricos, a escolha de um jogo como

produto final, torna o aprendizado mais envolvente e motivador, ao proporcionar um resultado visual e interativo. O jogo "Jornada IFSP: Atire nos Obstáculos" foi escolhido por permitir o uso de uma estrutura orientada a objetos clara, além de estimular o desenvolvimento de outras habilidades relevantes, como lógica de programação, manipulação de arquivos multimídia (imagens e sons) e controle de fluxo de execução.

A escolha do tema vai de encontro com a realidade vivida por alunos, que diariamente superam obstáculos para se manterem na jornada contínua de aprendizado. Servindo como estímulo e diversão para este público.

Outro ponto que justifica a realização deste projeto é a preparação para desafios reais da área de desenvolvimento de software, nos quais é essencial saber planejar, estruturar e implementar soluções com base nos princípios aprendidos em outras disciplinas. Trabalhar com elementos gráficos, eventos e atualizações em tempo real também permite ao estudante ter um primeiro contato com aspectos comuns do desenvolvimento de jogos e aplicações multimídia, ampliando sua visão profissional.

1.3 Aspectos Metodológicos

A metodologia adotada neste projeto seguiu um enfoque prático, baseado na aplicação dos princípios da Programação Orientada a Objetos (POO) em um ambiente de desenvolvimento real. O projeto foi implementado em linguagem C++, utilizando a biblioteca gráfica Raylib, que oferece recursos simples e eficientes para criação de jogos 2D.

O desenvolvimento foi dividido em etapas, permitindo a organização lógica do código e a separação de responsabilidades entre os componentes do sistema. Cada entidade do jogo (jogador, projéteis, obstáculos) foi representada por uma classe específica, permitindo o uso de herança e polimorfismo conforme os conceitos ensinados em sala.

A estrutura do projeto foi modularizada da seguinte forma:

- main.cpp: responsável pela execução principal do jogo, controle do loop principal e verificação de estados como “Game Over” e reinício;
- game.h: declaração das classes, variáveis globais e protótipos de funções utilizadas no jogo;

- game.cpp: implementação das funcionalidades de cada classe, lógica de movimentação, colisões, sons e desenhos na tela.

Durante o desenvolvimento, foram utilizados recursos visuais (sprites PNG) e sonoros (arquivos MP3) que enriqueceram a experiência do usuário e proporcionaram uma abordagem mais próxima do desenvolvimento profissional de jogos.

O ciclo de desenvolvimento envolveu as seguintes práticas:

1. Planejamento da estrutura orientada a objetos: definição de quais entidades seriam classes e seu relacionamento;
2. Implementação incremental: as funcionalidades foram desenvolvidas e testadas em partes;
3. Depuração e testes: o código foi executado e ajustado continuamente para garantir o bom funcionamento e eliminar erros;
4. Refinamento estético: foram aplicados recursos gráficos e sonoros para melhorar a apresentação do jogo;
5. Documentação: ao final, foi produzido o relatório contendo os fundamentos, justificativas e explicações técnicas do projeto.

1.4 Aporte Teórico

O desenvolvimento do projeto teve como base os fundamentos da Programação Orientada a Objetos (POO), um dos principais paradigmas da computação moderna. A POO permite organizar programas de maneira modular e reutilizável, por meio da definição de objetos que representam entidades do mundo real ou lógico. Os principais conceitos teóricos aplicados neste projeto foram:

Classe e Objeto: uma classe define a estrutura e o comportamento de um tipo de dado, enquanto o objeto é uma instância concreta dessa estrutura. No jogo, as classes Player, Projectile e Obstacle representam os elementos centrais da dinâmica do jogo.

Encapsulamento: promove o isolamento dos dados e comportamentos de uma classe, protegendo-os de alterações indevidas externas. Isso foi aplicado com o uso de variáveis privadas ou protegidas e métodos públicos que controlam o acesso.

Herança: permite que uma classe herde atributos e métodos de outra. A classe

base GameObject foi criada para representar qualquer entidade do jogo com posição, cor e velocidade, sendo estendida pelas classes Player, Projectile e Obstacle.

Polimorfismo: possibilita que diferentes classes derivadas da mesma base sejam tratadas de forma uniforme. Neste projeto, embora cada entidade implemente seus próprios métodos Update() e Draw(), todas seguem a mesma interface definida por GameObject.

Além dos conceitos de POO, o projeto também se fundamenta em noções básicas de desenvolvimento de jogos 2D, como:

Sprites: imagens utilizadas para representar visualmente os personagens e objetos do jogo;

Colisões: detecção de interseções entre retângulos para simular interações entre objetos (como disparos atingindo inimigos);

Game Loop: estrutura contínua de execução responsável por atualizar a lógica e renderizar a imagem do jogo na tela a cada quadro;

Gerenciamento de eventos: controle de teclado e entrada do jogador durante o jogo.

A biblioteca Raylib foi utilizada como recurso técnico para manipulação gráfica e sonora, por ser uma API de fácil aprendizado, bem documentada e com suporte multiplataforma. A Raylib facilita a criação de janelas, carregamento de texturas, reprodução de sons, detecção de colisões e controle de fluxo de jogo.

Esse embasamento teórico possibilitou não apenas a execução prática do projeto, mas também a consolidação do aprendizado dos conceitos fundamentais da POO e do desenvolvimento de jogos.

2 Metodologia

O desenvolvimento do projeto seguiu uma abordagem prática baseada nos conceitos de Programação Orientada a Objetos (POO), com o objetivo de consolidar os conhecimentos adquiridos durante a disciplina. Como ponto de partida, foram realizadas considerações iniciais sobre o escopo do jogo, suas funcionalidades

principais, mecânicas desejadas e recursos visuais e sonoros que comporiam a experiência do usuário.

Ferramentas Utilizadas

- Linguagem de Programação: C++;
- Biblioteca Gráfica: Raylib, utilizada para manipulação de gráficos, som e entrada de teclado de forma simples e eficiente.
- Ambiente de Desenvolvimento: Code::Blocks (com MinGW), por ser uma IDE leve e compatível com Raylib.
- Criador de Imagens: Para criar as imagens foi utilizado a ferramenta de inteligência artificial Chat GPT, que através de prompts específicos produziu imagens necessárias para atender o projeto.
- Editor de imagens: Para redimensionar nas imagens para os tamanho adequados foi utilizado o site “I love img” (www.iloveimg.com), que de maneira rápida deixou as imagens nas dimensões adequadas. Como as imagens do jogador e dos inimigos (player.png, enemy1.png, enemy2.png, enemy3.png, enemy4.png e background.png).

Arquivos Sonoros

Os arquivos sonoros em mp3 foram baixados do site Pixabay (www.pixabay.com) que dispõem de diversos efeitos utilizados como trilha de fundo, além de disparo e colisão.

Descrição de Funcionamento do Projeto Desenvolvido

O projeto consiste em um jogo 2D onde o jogador controla um personagem que pode se mover livremente pela tela e disparar projéteis contra obstáculos. Esses obstáculos aparecem de forma aleatória, com diferentes sprites, simulando inimigos variados. O jogador deve evitar colisões com os inimigos e eliminá-los antes que se aproximem demais.

As principais características do projeto incluem:

Sistema de Vida: O jogador possui pontos de vida e perde um ponto a cada colisão com inimigos.

Game Over e Reinício: Quando a vida chega a zero, o jogo exibe uma tela

de “Game Over” e permite reinício com a tecla R.

Disparo e Detecção de Colisão: Os projéteis lançados pelo jogador podem colidir com os obstáculos, que então desaparecem, emitindo som de impacto.

Sprites Dinâmicos: Os inimigos são sorteados entre diferentes imagens, tornando a jogabilidade mais variada.

Música e Sons: O jogo possui trilha sonora e efeitos sonoros distintos, aumentando a imersão do jogador.

O código foi estruturado de forma modular, com as responsabilidades divididas em três arquivos principais:

- main.cpp: Gerencia o ciclo principal do jogo e entrada do usuário.
- game.h: Define as classes e estruturas utilizadas.
- game.cpp: Implementa as funcionalidades do jogo, como movimentação, colisões, criação de obstáculos e sons.

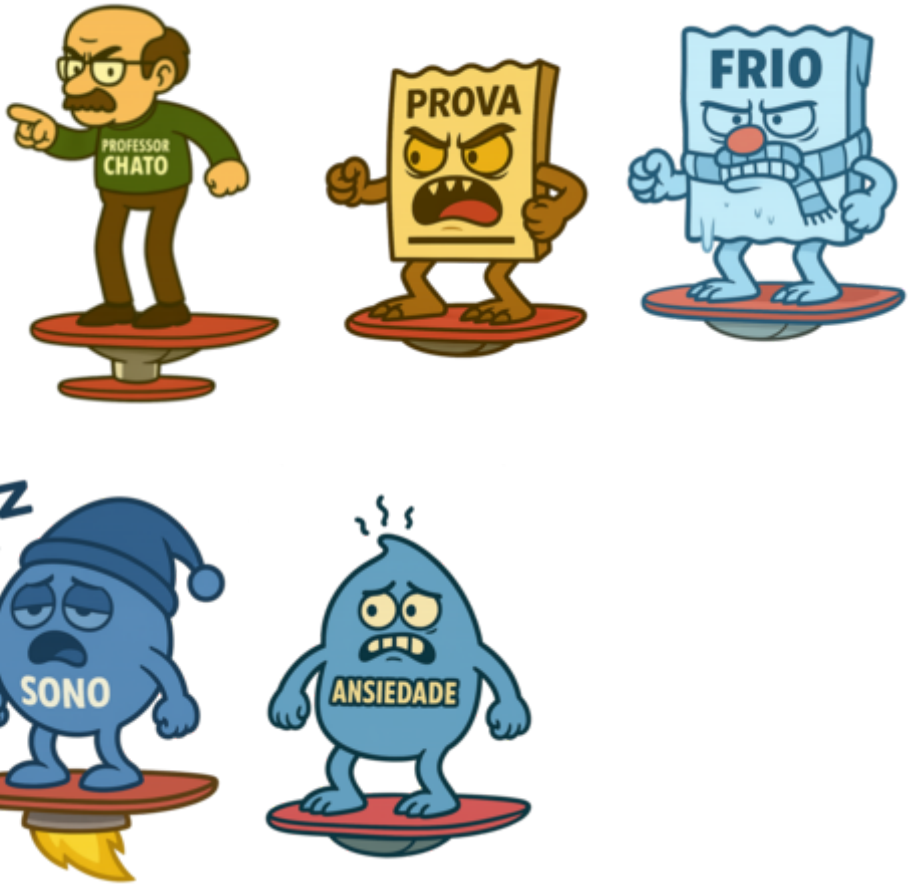
Essa organização favoreceu a manutenção e compreensão do código, além de permitir a aplicação clara dos princípios da programação orientada a objetos.

Personagens:

Jogador: No jogo “Jornada IFSP”, o jogador é representado por um garoto adolecente, que precisa vencer seus obstáculos para concluir seus objetivos como estudante. Para manter a identidade do personagem, o garoto usa blusa de moletom verde, escrito IFSP. Para se manter no jogo, ele atira com seu estilingue em seus “obstáculos” enquanto se movimenta com seu skate voador de forma lúdica.

Obstáculos: Os inimigos do jogo são tratados como obstáculos, que tentam

impedir que o jogador possa atingir seus objetivos como estudante. Estes personagens foram criados de forma descontraída para gerar identificação nos usuários. São eles: “Professor Chato”, “Prova”, “Frio”, “Sono” e “Ansiedade”.



Tela de fundo: O background foi criado também com o objetivo de gerar identificação com a cidade de Campos do Jordão-SP, cujo Instituto Federal serviu como inspiração para o jogo. Nesta imagem podemos ver uma cidade entre as montanhas, com casas em estilo europeu e árvores de araucária.

Os inimigos do jogo são tratados como obstáculos, que tentam impedir que o jogador possa atingir seus objetivos como estudante. Estes personagens foram criados de forma descontraída para gerar identificação nos usuários. São eles: “Professor Chato”, “Prova”, “Frio”, “Sono” e “Ansiedade”.

3. Resultados Obtidos

Como resultado da aplicação da metodologia proposta, foi desenvolvido com sucesso um jogo 2D completo utilizando a linguagem C++ e a biblioteca gráfica Raylib. O jogo simula uma jornada em que o jogador deve atirar em obstáculos (representados por inimigos com diferentes sprites) enquanto evita colisões que diminuem seus pontos de vida.

O jogo traz uma dinâmica animada, com imagens que geram identificação com alunos do Campus. Criando ainda uma experiência divertida e descontraída para os usuários, além de sensações sonoras empolgantes, que causam atração por sua harmonia no jogo.

Os principais resultados alcançados foram:

- Funcionalidade completa do jogo:

O jogo foi finalizado com um loop principal estável, que permite iniciar, jogar e reiniciar a partida após o término ("Game Over"), mantendo o controle de fluxo sem travamentos ou falhas.

- Implementação correta de POO:

Os conceitos de Programação Orientada a Objetos foram aplicados com clareza e eficiência. As classes Player, Projectile e Obstacle herdaram corretamente da superclasse GameObject, promovendo reutilização e organização do código.

- Interatividade com o usuário:

O jogador pode controlar o personagem usando as teclas direcionais e disparar projéteis com a barra de espaço. Após perder toda a vida, o jogo exibe uma mensagem de fim e permite reinício com a tecla R.

- Sprites personalizados:

O personagem principal e os inimigos foram representados por imagens .png personalizadas. Os inimigos aparecem aleatoriamente com diferentes visuais, aumentando a diversidade visual do jogo.

- Recursos sonoros integrados:

O jogo conta com trilha sonora de fundo e efeitos sonoros distintos para disparo, colisão de tiro com inimigo e colisão do inimigo com o jogador. Isso torna a experiência mais imersiva.

- Gerenciamento eficiente de memória:

Todos os recursos gráficos e sonoros utilizados são devidamente carregados e descarregados, evitando vazamentos e mantendo o desempenho.

- Código modularizado e bem estruturado:

A separação do projeto em arquivos main.cpp, game.cpp e game.h facilitou a manutenção, a leitura e futuras expansões do código.

4. Conclusão

O resultado final atendeu às expectativas iniciais: foi desenvolvido um jogo 2D funcional, com controle por teclado, sprites animados, efeitos sonoros e trilha musical, além de mecânicas de disparo, colisão, pontuação por sobrevivência e reinício automático. O jogo também foi modularizado em diferentes arquivos para facilitar a organização e a manutenção do código, seguindo boas práticas de engenharia de software. A imersão no ambiente acadêmico do Instituto Federal de Campos do Jordão causou boas impressões, gerando clima de fantasia e identificação.

No entanto, o projeto ainda possui diversas possibilidades de expansão e aprimoramento. Entre as sugestões de melhorias para versões futuras, destacam-se:

- Implementação de um sistema de pontuação com ranking.
- Aumento progressivo da dificuldade com o passar do tempo (ex: inimigos mais rápidos).
- Inclusão de novos tipos de obstáculos com comportamentos distintos.
- Criação de uma tela inicial com menu interativo.
- Adição de efeitos visuais (explosões, animações de impacto).
- Suporte a múltiplos níveis ou fases.

Concluindo, o projeto foi uma experiência enriquecedora, não apenas do ponto de vista técnico, mas também em termos de planejamento, organização e execução de um sistema interativo completo. Ele demonstrou como os fundamentos da POO podem ser usados para estruturar de forma eficiente aplicações mais complexas e orientadas a eventos.

Referências Bibliográficas

RAYLIB. raylib: A simple and easy-to-use library to enjoy videogame programming. Disponível em: https://www.raylib.com. Acesso em: junho de 2025.

YOUTUBE. Tutoriais sobre Raylib e desenvolvimento de jogos com C++. Diversos autores. Disponível em: https://www.youtube.com. Acesso em: junho de 2025.