UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

INSTITUTO DE INFORMÁTICA

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

ARTHUR VALENTINO RIGO RAVANELLO

LUCAS VIEIRA BAGOLIN

TOBIAS CADONA MARION

RELATÓRIO DO TRABALHO FINAL DE ALGORITMOS E PROGRAMAÇÃO

Porto Alegre

2025

Introdução ao Projeto

O projeto é uma recriação do jogo Jetpack Joyride, desenvolvido com a biblioteca Raylib. Ele utiliza elementos interativos como jogador, mapa dinâmico, obstáculos e coleta de moedas para criar uma experiência imersiva. Além disso, o jogo conta com diferentes estados (início, gameplay e game over) e funcionalidades adicionais, como placar de líderes e cheats, que enriquecem a jogabilidade.

Representação dos Elementos do Jogo

O jogador é representado pela estrutura Player (definida no arquivo player.c), que armazena dados fundamentais para a dinâmica do jogo, como sua posição no eixo Y (positionY), velocidade (speedY), estado de invulnerabilidade (isInvulnerable) e recursos coletados, como moedas (coins). O mapa, por sua vez, é gerado dinamicamente e composto por células definidas na estrutura MapSection (localizada no arquivo map.c). Essas células podem conter espaços vazios (' '), moedas ('C') ou obstáculos ('X' e 'Z'). Esses elementos estruturam o cenário e criam desafios progressivos para o jogador. A pontuação é calculada com base na distância percorrida e nos itens coletados, sendo então registrada em uma leaderboard utilizando a estrutura Save, definida no arquivo leaderboard.c.

Interação dos Componentes

A interação entre os componentes do jogo acontece de forma intuitiva e fluida. O jogador controla os movimentos através da teclas de espaço ou da seta para cima, utilizando a função movePlayer (localizada em player.c). Cheats, como “GHOST” para invulnerabilidade e “SLOW” para câmera lenta, são gerenciados pela função checkCheatWords (também em player.c). A detecção de colisões, realizada pela função checksCollision (em player.c), verifica a interação do jogador com o mapa e ajusta seu estado conforme necessário. Além disso, a movimentação do mapa, implementada pela função moveMap (em map.c), cria um efeito de avanço constante, intensificando o desafio conforme o jogador progride.

Funcionalidades e Estruturas Utilizadas

O programa é composto por diversas funções que permitem inicializar e desenhar elementos do jogo, carregar mapas de forma aleatória e gerenciar a leaderboard. As principais estruturas incluem:

- Player, definida em player.c, para armazenar informações do jogador.
- Level, definida em map.c, que controla dados do nível atual, como gravidade (gravity) e velocidade do mapa (speed).
- Save, definida em leaderboard.c, responsável por registrar pontuações e informações de data através da função saveGame. Essas estruturas trabalham de maneira integrada, garantindo a funcionalidade do jogo.

Como Utilizar o Programa

O uso do programa é simples e acessível. Ao iniciar o jogo, o jogador pode navegar por meio das setas ou da movimentação do mouse pelo menu principal, renderizado com a função drawHomeScreen (em ui.c), para iniciar uma partida (Play), consultar as pontuações (Highscores) ou sair (Exit Game). Durante a gameplay, o jogador deve coletar moedas, evitar obstáculos e, opcionalmente, usar cheats para melhorar sua performance. Ao final do jogo, é possível salvar a pontuação na leaderboard através da função saveGame, criando um registro para comparação futura.

Conclusão

Durante o desenvolvimento deste projeto, tivemos a oportunidade de aprender muito sobre a criação de jogos interativos e o uso eficiente de bibliotecas gráficas, como a Raylib. Cada um de nós contribuiu com diferentes aspectos do desenvolvimento, desde a implementação da lógica do jogo até a pesquisa e coleta dos elementos visuais e a integração das funcionalidades, como a leaderboard e os cheats. Este trabalho nos ensinou a importância de uma abordagem modular e colaborativa, permitindo que entendêssemos melhor como diferentes partes de um programa se conectam para criar uma experiência coesa.

Além disso, a realização do projeto nos desafiou a resolver problemas práticos, como o gerenciamento de estados do jogo e a otimização de recursos, o que nos proporcionou um aprendizado valioso em programação. Estamos confiantes de que esses aprendizados serão úteis em futuros desafios acadêmicos e profissionais.

Resumo da Composição dos Arquivos do Jogo

1. index.c:

Descrição: Controla o fluxo principal do jogo, gerenciando a inicialização, os estados de tela (HOME, GAMEPLAY, PAUSE, etc.) e a lógica central da gameplay.
Principais Funções e Controle:

- Inicializa o jogo com InitWindow, configurações gráficas e FPS (SetTargetFPS).
- Gerencia os estados de tela no loop principal usando currentScreen (HOME, GAMEPLAY, PAUSE, etc.).
- Controla transições entre telas com funções específicas como drawHomeScreen e drawGameOverScreen.
- Lida com a movimentação do mapa e lógica de progresso (moveMap, player.distance).
- Trata cheats com a função checkCheatWords e condições para pausa e gameplay.

---

2. player.c

Descrição: Gerencia o comportamento e as propriedades do jogador, como movimentação, colisão, invulnerabilidade e coleta de moedas.
Principais Funções e Controle:

- Estrutura Player: Armazena informações como posição (positionY), velocidade (speedY), moedas coletadas (coins), vidas, e estado de invulnerabilidade (isInvulnerable).
- initializePlayer: Configura os atributos iniciais do jogador, como posição e textura.
- movePlayer: Atualiza a posição e a velocidade do jogador com base na gravidade e no salto.
- checksCollision: Detecta colisões com o mapa (obstáculos e moedas) e ajusta o estado do jogador.
- checkCheatWords: Processa a entrada do jogador para ativar cheats, como invulnerabilidade ou câmera lenta.
- drawPlayer: Renderiza o jogador na tela.

---

3. map.c

Descrição: Gerencia a lógica e os dados do mapa, incluindo geração de seções, movimentação e carregamento de texturas.
Principais Funções e Controle:

- Estrutura MapSection: Representa as células do mapa, que podem ser espaços vazios, moedas ou obstáculos.
- Estrutura Level: Armazena informações do nível atual, como velocidade do mapa (speed), gravidade (gravity) e texturas.
- loadMapRandomly: Seleciona uma seção aleatória do mapa para gerar novos desafios.
- moveMap: Desloca o mapa em direção ao jogador, criando a sensação de movimento.
- drawMap: Renderiza as células do mapa na tela.
- readMapFile: Lê arquivos de mapas salvos e valida caracteres.
- unloadMapTextures: Libera as texturas carregadas para economizar recursos.
- float offsetX: Variável global que ajusta a fluidez do mapa.

---

4. leaderboard.c

Descrição: Gerencia a leaderboard, incluindo salvamento e ordenação de pontuações.
Principais Funções e Controle:

- Estrutura Save: Representa o registro de pontuações, contendo o nome do jogador (name), pontos (points) e data (currentDate).
- initializeSave: Configura um registro de pontuação vazio.
- giveDate: Captura a data atual e a insere no registro de pontuação.
- score: Calcula a pontuação com base na distância percorrida e moedas coletadas.
- saveGame: Salva uma pontuação na leaderboard, verificando se é alta o suficiente para ser registrada.
- sortSaves: Ordena as pontuações do vetor Save em ordem decrescente.
- openSavesFile e saveSavesFile: Gerenciam a leitura e escrita dos registros da leaderboard no arquivo binário.

---

5. ui.c

Descrição: Controla a interface gráfica do jogo, gerenciando botões, telas de menu, e elementos visuais de HUD.
Principais Funções e Controle:

- createButton: Desenha botões interativos na tela e detecta cliques ou atalhos do teclado.
- createInputText: Cria campos de entrada de texto para o jogador inserir dados, como nome para salvar a pontuação.
- drawHomeScreen: Renderiza a tela inicial com opções como “Play”, “Highscores” e “Exit”.
- drawGameOverScreen: Renderiza a tela de game over, permitindo reiniciar o jogo ou salvar a pontuação.
- drawSaveGameScreen: Gerencia a tela de salvamento, permitindo registrar pontuações.
- drawHighScoresScreen: Exibe a leaderboard, mostrando os melhores desempenhos.
- navigateWithArrowKeys: Gerencia a navegação no menu com teclas direcionais.
- Elementos visuais incluem HUD de distância, pontuação e mensagens de erro.
- int amountOfOptionsOnScreen e int currentSelectedOption: Variáveis globais que controlam a mecânica dos menus.

---

6. constants.h

Descrição: Define constantes globais para uso em todo o projeto, configurando valores fixos para elementos como tamanho do mapa, dimensões da tela e propriedades do jogador.
Principais Constantes e Controle:

- Jogador: INITIAL_X_POSITION, INITIAL_JUMP_POWER, INVULNERABLE_AFTER_HIT_DURATION.
- Mapa: MAP_HEIGHT, MAP_WIDTH, SECTION_WIDTH.
- Níveis: AMOUNT_OF_LEVELS, LEVEL_SPEED_MULTIPLIER.
- Interface: SCREEN_WIDTH, SCREEN_HEIGHT, BUTTON_WIDTH, BUTTON_HEIGHT.
- Leaderboard: MAX_SAVES, LEADERBOARD_FILE_PATH.
- Cheats: INVULNERABLE_CHEAT_CODE e SLOW_CHEAT_CODE.