Universidade Federal de São João del Rei
Campus Tancredo Neves - Ciência da Computação

Oscar Alves Jonson Neto

Documentação do Trabalho Prático de Programação Modular

São João del Rei
2025

Oscar Alves Jonson Neto

Documentação do Trabalho Prático de Programação Modular

São João del Rei
2025

Sumário

1. Introdução

O objetivo do TP era criar um jogo aplicando a matéria vista em sala de aula, com isso, decidi por fazer o jogo da cobrinha (Snake Game), um clássico dos jogos eletrônicos, para auxiliar na criação do jogo utilizei a biblioteca Pygame. O jogo conta com um menu interativo, onde o jogador pode iniciar o jogo, acessar a tela de desbloqueios de skins ou sair do jogo. Durante a partida, o jogador controla uma cobrinha que deve coletar frutas para crescer e aumentar sua pontuação, evitando colidir com as paredes ou com o próprio corpo. O jogo também conta com várias skins que podem ser desbloqueadas atingindo certas metas de pontuação.

2. Módulos e métodos

2.1 Main.py

Funcionalidade

- O código gerencia o fluxo principal do jogo, incluindo o menu, a seleção de skins e a execução do jogo.
- Utiliza arquivos externos (settings.txt) para armazenar configurações e o highscore.
- Inclui sons para melhorar a experiência do jogador.
- Permite a personalização da cobrinha através de skins desbloqueadas com base no highscore.

Métodos

- **carregar_configuracoes** lê as configurações do jogo a partir do arquivo settings.txt, extraindo os valores de largura, altura e highscore.

- **atualizar_highscore** atualiza o valor do highscore no arquivo settings.txt quando um novo recorde é alcançado.

- **main** gerencia o fluxo principal do jogo, inicializando os recursos, exibindo o menu principal e permitindo ao jogador escolher entre jogar, acessar as skins ou sair. Durante a execução, o highscore é atualizado, se necessário, e os sons do menu e de escolha são reproduzidos para melhorar a experiência do jogador.

2.2 Game.py

Funcionalidade

- Loop principal do jogo
- Processa eventos de input
- Atualiza posições
- Verifica colisões
- Renderiza objetos
- Controla FPS

Métodos

- **__init__** inicializa o jogo, configurando a tela, a cobrinha, a fruta, as texturas das skins e os sons.

- **verificar_colisao** verifica se a cobrinha colidiu com as bordas da tela ou com o próprio corpo, encerrando o jogo caso ocorra colisão.

- **verificar_comida** verifica se a cobrinha coletou a fruta, reposicionando a fruta e aumentando o tamanho da cobrinha quando isso acontece.

- **mostrar_informacoes** exibe o score, o highscore e o tempo de jogo na área de informações à direita da tela.

- **desenhar_objetos** desenha a fruta, o corpo da cobrinha e a cabeça da cobrinha na tela, aplicando as texturas das skins selecionadas.

- **gameover** exibe a tela de Game Over, mostrando o score final e uma mensagem de "New Highscore" se o jogador bater o recorde.

- **loop_principal** gerencia o loop do jogo, processando eventos de teclado, atualizando a posição da cobrinha, verificando colisões e comida, e desenhando os objetos na tela. Quando o jogo termina, ele exibe a tela de Game Over e retorna o score final.

2.3 Menu.py

Funcionalidade

- Implementa a interface do menu principal com opções de Jogar, Skins e Sair
- Permite navegação entre as opções usando setas do teclado
- Renderiza o menu com efeitos visuais de seleção

Métodos

- **__init__** configura o menu com as opções e a fonte.
- **desenhar_menu** exibe o menu na tela, destacando a opção selecionada.
- **mover_selecao** permite navegar entre as opções.
- **selecionar** retorna a opção escolhida pelo jogador.

2.4 Skins.py

Funcionalidade

- Gerencia o sistema de skins/texturas do jogo
- Define diferentes skins com requisitos de pontuação para desbloqueio
- Permite visualização e seleção das skins disponíveis
- Carrega e armazena as texturas das skins

Métodos

- **carregar_skins** carrega as texturas das skins a partir dos arquivos de imagem e retorna as texturas da skin padrão.
- **selecionar_skin** permite ao jogador navegar e selecionar uma skin desbloqueada, com base no highscore atual. Ela exibe o menu de desbloqueios e gerencia a interação do jogador com as teclas de navegação e seleção.
- **menu_desbloqueios** exibe o menu de desbloqueios na tela, mostrando as skins disponíveis. As skins desbloqueadas são destacadas em verde, enquanto as bloqueadas são exibidas em cinza com um símbolo "X" e o highscore necessário para desbloqueá-las. A skin selecionada é destacada com uma borda branca.

2.5 Cobra.py

Funcionalidade

- Define as classes Cabeça e Corpo que controlam a cobra
- Implementa movimentação, crescimento e detecção de colisões
- Cabeça: controla direção e movimento
- Corpo: gerencia as partes/segmentos da cobra

Métodos

Classe Cabeça:

- **__init__** inicializa a posição inicial da cabeça e define a direção inicial como parada (dx = 0, dy = 0).
- **mover** atualiza a posição da cabeça com base na direção atual (dx e dy).
- **mudar_direcao** altera a direção da cabeça, evitando que ela reverta a direção atual (impedindo movimentos inválidos, como ir para trás).

Classe Corpo

- **__init__** inicializa a lista de partes do corpo, começando vazia.
- **atualizar** adiciona a nova posição da cabeça ao corpo e remove a parte mais antiga, mantendo o comprimento do corpo constante.
- **crescer** aumenta o comprimento do corpo ao adicionar uma nova parte na última posição.
- **colidiu_com_cabeca** verifica se a cabeça colidiu com alguma parte do corpo, exceto a última (para evitar falsas colisões).

2.6 Fruta.py

Funcionalidade

- Implementa a classe Fruta que representa o item coletável
- Gera posições aleatórias dentro dos limites do jogo
- Permite reposicionar a fruta quando coletada

Métodos

- **__init__** inicializa a fruta com uma posição aleatória dentro dos limites da tela, garantindo que ela fique alinhada com a grade do jogo (baseada no tamanho do bloco).
- **reposicionar** redefine a posição da fruta para uma nova localização aleatória, também alinhada com a grade do jogo, quando a cobrinha a coleta.

3. Desenvolvimento

O desenvolvimento do jogo da cobrinha foi um processo desafiador, que envolveu a aplicação de conceitos da disciplina e da biblioteca PyGame . Abaixo, descrevo as principais dificuldades encontradas durante o processo e como as mesmas foram solucionadas.

- Dificuldade para entender a biblioteca PyGame: por ser uma biblioteca grande, a curva de aprendizado inicial é bem íngreme, mas após estudo da documentação e exemplos que estão no site, esse problema foi solucionado.

- Cobra conseguia se mover na direção oposta imediatamente: no começo do desenvolvimento a cobra podia virar pra dentro de si, para solucionar esse problema foi colocado uma verificação no método **mudar_direção** para evitar que isso acontecesse.

- Posicionamento dos elementos nos menus: quando o menu estava sendo implementado todos os seus elementos eram colocados de maneira estática, por mais que eu não consegui aplicar mais de um tamanho de tela, o objetivo era deixar os elementos dinâmicos, coisa que foi feita no menu principal ao utilizar a altura e largura da tela para posicionar os elementos.

- Troca de Skin dentro do jogo: essa dificuldade veio do fato que o metodo **selecionar_skin** precisava retornar 3 skins diferentes, a cabeça, o corpo e a fruta, isso foi problemático no começo pois não estou acostumado com o processo de return em Python, mas estão compreendi que uma lista com os 3 caminhos poderia ser utilizada.

4. Conclusão

O desenvolvimento deste jogo da cobrinha foi desafiador, mas que permitiu a aplicação de conceitos importantes de programação, como a Programação Orientada a Objetos (POO), o gerenciamento de estados e a integração de recursos multimídia. A adição de funcionalidades como as skins e os efeitos sonoros melhoraram o conceito clássico do jogo.

A modularização do código foi essencial para manter a organização e facilitar a manutenção, permitindo que cada parte do jogo (como a cobrinha, a fruta, o menu e as skins) fosse desenvolvida e testada de forma independente.

5. Referências

Documentação do PyGame: https://www.pygame.org/docs/

Exemplos de outros jogos: https://www.pygame.org/tags/all

Sprites:

- Bowser e Goomba: https://www.deviantart.com/nkelsch/art/Super-Mario-20x20-Sprites-83482505
- Mario Star: https://www.pngwing.com/pt/free-png-zfscm
- Sonic: https://www.pngegg.com/pt/png-ehcuq
- Emerald Chaos: https://www.anyrgb.com/en-clipart-hpol3
- Sonic Golden Ring: https://toppng.com/free-image/vintagesonic1-sonic-ring-gif-PNG-free-PNG-Images_167435
- Plano de Fundo Menu Principal: https://www.behance.net/gallery/65290819/Pixel-Art-Backgrounds-Tutorial-Skip

Sons:

- Coletar Fruta: https://pixabay.com/sound-effects/8-bit-powerup-6768/
- Música do menu: https://pixabay.com/sound-effects/026491-pixel-song-8-72675/
- SFX Botões do menu: https://pixabay.com/sound-effects/coin-collect-retro-8-bit-sound-effect-145251/
- Som do Gameover: https://pixabay.com/sound-effects/videogame-death-sound-43894/