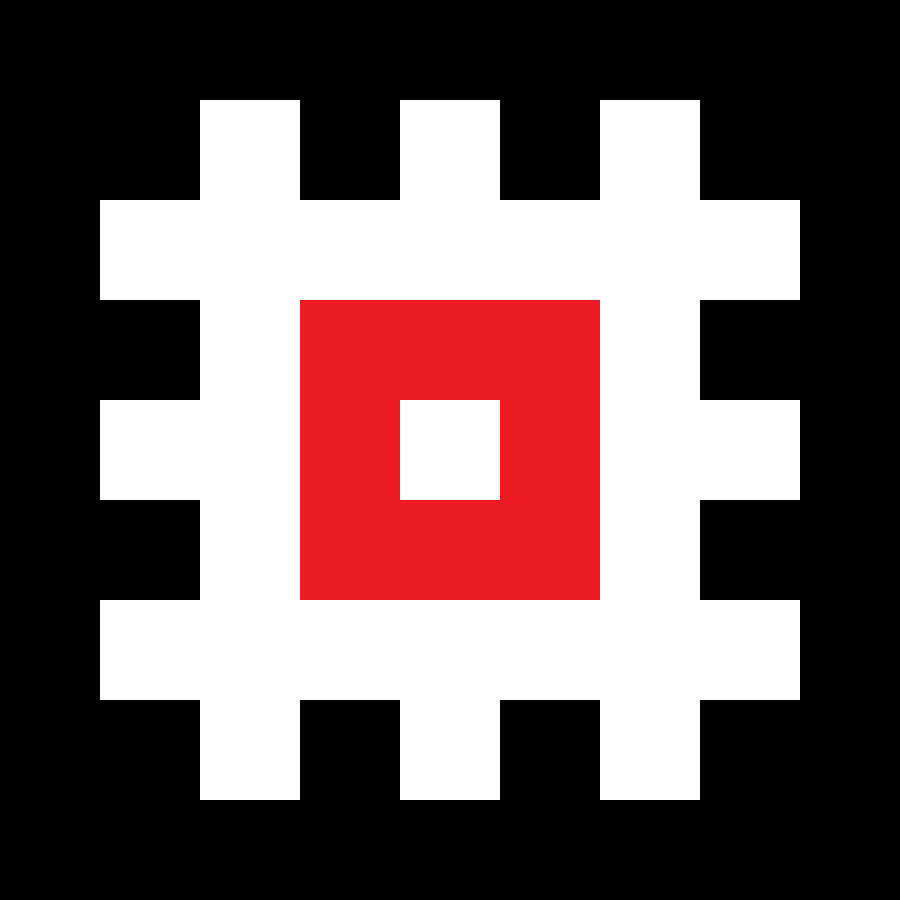
Documentação - Trabalho Prático

**CUBO**



Universidade Federal de Minas Gerais

Programação e desenvolvimento de Software I

Prof: Pedro O.S. Vaz de Melo

Aluno: Marcos Paulo F. de Souza

**Introdução**

O trabalho prático da disciplina de programação e desenvolvimento de software I foi criar um jogo inspirado no jogo Final Fantasy VI. Esse jogo, basicamente, deve ter dois modos de jogo. O modo de exploração, onde o jogador pode andar por um mapa com inimigos escondidos, e o modo de batalha, onde o jogador luta contra um inimigo que o pega de surpresa. Além disso, duas telas finais, sendo uma delas de vitória que mostra ao jogador se os pontos que ele fez foram um novo recorde ou a tela derrota que avisa ao jogador que ele perdeu, e um menu inicial onde o jogador tem a opção de escolher jogar ou sair do jogo, esse menu também pode ser visto a partir do modo exploração.

**Enredo**

O jogo se passa em um mundo de formas geométricas que viviam em harmonia. Até que um quadrado lunático começa a pregar para os outros quadrados que eles deveriam governar esse mundo e se auto denomina como o CUBO que está além das duas dimensões de seu mundo. Poucas outras formas geométricas tiveram coragem de enfrentar o CUBO mas infelizmente elas não tiveram sucesso pois nem conseguiram alcançar o castelo dele. E em algum momento antes do CUBO conseguir chegar ao seu objetivo surge um triângulo que se opôs a ele e treinou para superá-lo antes de enfrentá-lo e se equipou com armaduras que o ajudariam em seu combate.

**Comandos**

No jogo deve-se utilizar as setinhas do teclado (↓ ↑ ← →) para movimentar o herói e as setas para cima e para baixo (↓ ↑) no modo de batalha e no menu para escolher a opção desejada que é selecionada com a tecla Enter. Além disso, enquanto se estiver no modo de exploração é possível pressionar a tecla ESC para abrir o menu inicial.

**Estrutura de dados**

Para desenvolver esse jogo foram necessários 5 tipos de dados, eles são:

Cord: Essa struct armazena dois inteiros chamados x e y, que armazenam uma coordenada do display.

Estrutura: Essa struct armazena as variáveis importantes de qualquer estrutura dentro do jogo, como o tamanho do bitmap da estrutura, o bitmap da estrutura, a coordenada da estrutura utilizando a struct anterior e o tipo da estrutura que guarda informações variáveis.

Retangulo: Essa struct armazena duas coordenadas que são necessárias para desenhar um retângulo através das primitivas do allegro.

Heroi: Essa struct armazena todas as informações importantes para o herói que é controlado pelo jogador. Informações como vida e ataque que são importantes na batalha e os sprites que carregam a aparência do herói, etc.

Monstro: Essa struct armazena todas as informações importantes para os monstros, como vida e ataque importantes na batalha e a aparência do monstro e uma estrutura que é o projétil que será usado contra o herói.

**Código do jogo**

De início é importante dizer que todas as variáveis importantes do jogo, como o herói ou os monstros, foram criadas como variáveis globais, pois assim são de mais fácil acesso e diminui a quantia de parâmetros das funções. Explicarei o código a partir da função main, pois a partir dela todas as funções são chamadas. Dentro da main a primeira coisa que acontece é uma longa rotina de inicialização que cria variáveis fornecidas pelo allegro como “display”, que são usadas durante todo o jogo, e inicializa todas as ferramentas do allegro que serão usadas como exemplo as “primitives\_addon”. No fim dessas rotinas o módulo de inicialização das variáveis globais é chamado e nele, após a inicialização das globais, são chamados dois outros módulos primeiro o módulo “criaMonstros()” que inicializa o vetor de monstros e depois o módulo “carregaBitmap()” que carrega todos os bitmaps que serão usados no jogo para a memória. É importante ressaltar que existem condições para criação dos monstros, uma delas é que as coordenadas deles não podem estar em contato com o objetivo do herói e nem com o herói em sua posição inicial. E assim termina a rotina de inicialização.

Após o fim da inicialização, o jogo entra em um loop que só termina quando a variável global “jogando” receber o define “NAO” que é equivalente a 0, vale ressaltar que ela foi iniciada com o define “SIM” que equivale a 1. Dentro desse loop existe uma sequência de quatro “IFs” que verificam qual o modo de jogo e chamam o módulo correspondente. O jogo tem 5 modos diferentes, o modo de jogo inicial é o “MENU\_INICIAL” que tem dá duas opções ao jogador(“Jogar” ou “Sair”), caso o jogador escolha sair o jogo acaba, caso ele escolha jogar o modo de jogo é alterado para “EXPLORACAO”. No modo de jogo exploração é desenhado no display o castelo, que é o alvo do herói, e o herói, além da pontuação atual do herói. A partir desse modo o jogador pode, através do ESC, voltar ao menu inicial mas sem reiniciar o jogo, entrar em uma batalha caso ele se encontre com um dos monstros escondidos ou ganhar o jogo ao alcançar o seu objetivo que é entrar no castelo. Ao entrar no modo de “BATALHA” será desenhado no display o cenário da batalha o herói e o inimigo com suas respectivas vidas e um pequeno menu que dá opções ao jogador. O jogador nesse modo terá a opção de “Atacar”, “Especial” ou “Fugir”. Caso ele escolha atacar será feito um ataque comum que tem um dano fixo, caso ele escolha o especial será feito um ataque especial que pode variar o dano em três possibilidades dar 0.75 vezes o dano normal, dar o dano normal ou dar 3 vezes o dano normal, esse dano varia com porcentagem sendo 60% de chance da primeira opção, sendo 30% de chance do dano normal e 10% de chance do dano maior. Ao escolher fugir o jogador tem 60% de chance de conseguir, caso não consiga o inimigo ataca e ele terá que escolher de novo. O modo batalha acaba quando o inimigo morrer ou o herói morrer, caso o herói morra o modo de jogo muda para

derrota, caso ele derrote o inimigo o modo de jogo volta para “EXPLORACAO”. E os dois últimos modos de jogo são “VITORIA” e “DERROTA” que são realizados pelo mesmo módulo “desenhaFim()”. Caso o modo de jogo seja um desses dois a variável “jogando” recebe “NAO” depois do “desenhaFim()” ser feito. No desenha fim é verificado qual dos dois modos está sendo feito e caso seja “Derrota” ele desenha a tela de derrota, caso seja “VITORIA” ele desenha a tela de vitória e a pontuação do jogador, que caso seja maior que o último recorde que está salvo em um arquivo ele atualiza o recorde e escreve na tela “Novo Recorde!” caso não seja um novo recorde ele escreve o recorde.

Depois dessa tela final o loop acaba e é realizado as rotinas de finalização e o jogo acaba.