DOCUMENTAÇÃO DO TRABALHO PRÁTICO

Autor: Alexandre Lima Barbosa
Curso: Engenharia de Sistemas – UFMG
Disciplina: Programação e Desenvolvimento de Software 1
Professor: Pedro O. S. Vaz de Melo

1. Instruções do Jogo

O jogo desenvolvido teve como inspiração a engine de FinalFantasy VI. Dividido em três telas: Modo de Navegação, Batalha e tela Final. No modo de navegação a movimentação do personagem se dá pelas teclas W, A, S, D ou pelas teclas direcionais. Já no modo de Batalha a escolha no painel de opções só se dá pelas teclas ↑, ↓ e ao apertar ENTER que de fato escolhe a opção. Em cima do inimigo e do personagem há uma barra de vida, a do inimigo muda de cor em relação ao nível dele. Seu ataque especial tem um tamanho maior, porém seu dano é aleatório podendo ser menor ou maior do que o ataque padrão. Também há a opção de tentar fugir mas talvez você falhe, tome cuidado.
Há também um sistema de pontuação e recorde, sua pontuação sobe conforme você derrota os inimigos, ganhando mais pontos se derrotar inimigos de níveis mais altos.

2. Enredo do Jogo

Você é um viajante(mochileiro) da galáxia que precisa fazer entregas, entre elas o Baby Yoda, no outro lado do seu sistema solar. Porém para isso você precisa passar por uma nuvem de asteroides lotadas de piratas, e se você se chocar com algum asteroide os piratas te abordarão e você precisará se defender. Tome cuidado ao se direcionar para trás enquanto estiver indo pra frente, talvez você seja pego pela corrente de asteroides e enfrentar um grande número de inimigos em seguida. Seu objetivo é chegar ao final da tela no canto superior direito e fazer as entregas.

3. Detalhamento do código

Linha 1 a 9: bibliotecas utilizadas no código;
Linhas 11 a 32: definições de variáveis com valor pré-definido;
Linhas 34 a 58: constantes globais;
Linhas 60 a 72: variáveis globais;

Linhas 75 a 108: estruturas do pirata e do mochileiro, com suas respectivas variáveis;
Linhas 111 a 126: função que inicializa as variáveis do mochileiro;
Linhas 128 a 155: função que inicializa as variáveis do mochileiro;
Linhas 157 a 184: função que inicializa as variáveis globais do código;
Linhas 118 a 132: função que desenha o herói baseado na direção que está navegando;
Linhas 188 a 192: função que escreve o contador da pontuação e o objetivo final;
Linhas 193 a 206: função que processa a movimentação do herói na tela de navegação;
Linhas 208 a 213: função que desenha o cenário da navegação;
Linhas 215 a 219: função que desenha os asteroides;
Linhas 221 a 270: função que processa as teclas no modo navegação;
Linhas 272 a 276: função que calcula quando o mochileiro chega ao objetivo final;
Linhas 278 a 295: funções que calculam e impedem a criação de monstros colidindo com os objetos e consigo mesmo;
Linhas 299 a 316: função que desenha o cenário da batalha;
Linhas 318 a 322: função que detecta se o mochileiro encontrou um inimigo.
Linhas 324 a 338: função que desenha o cenário de batalha, com variações de cor da vida do inimigo, dependendo do seu nível. Também desenha as barras de vida, do inimigo e do herói, e suas variações de acordo com o dano tomado;
Linhas 340 a 374: função que processa as teclas no modo navegação;
Linhas 377 a 383: função que desenha a tela de "game over", caso o jogador morra e perca o jogo, e imprime o recorde e pontuação do jogador;
Linhas 385 a 390: função que desenha a tela de vitória, caso o jogador chegue no objetivo final, e imprime o recorde e pontuação do jogador;
Linha 392: início da função main;
Linhas 394 a 509: rotinas de inicialização do Allegro e inicialização do bitmap;
Linhas 511 a 540: inicialização de variáveis utilizadas na main e chamada de funções essenciais para funcionamento do jogo, antes do while;
Linhas 545 a 709: todo o processo do while, com a chamada de quase todas as funções citadas nas linhas iniciais do código que são necessárias para a

montagem e funcionamento do jogo, e algoritmo para o funcionamento do modo batalha incluso;

Disposição do While:

```
545   while(playing) {
546
547     ALLEGRO_EVENT ev;
548     //espera por um evento e o armazena na variavel de evento ev
549     al_wait_for_event(event_queue, &ev);
550
551     //se o tipo de evento for um evento do temporizador, ou seja, se o tempo passou de t para t+1
552     if(ev.type == ALLEGRO_EVENT_TIMER) {
553
554
555       if(modo_jogo == NAVEGACAO){
556
557         desenhaCenarioNaveg();
558         desenhaPont(&jorj);
559         desenhamochileiroNaveg(jorj);
560
561
562         for(i=0; i < MAX_ASTEROIDES; i++)
563         {
564           //desenhaAsteroide(asts[i]);
565         }
566
567         //desenhaAsteroide(a);
568
569
570         for(i=0; i < MAX_ASTEROIDES; i++)
571         {
572
573           if(detectouAsteroide(jorj, asts[i]) && asts[i].vida>0)
574           {
575             modo_jogo = BATALHA;
576             indicador_asteroide=i;
577           }
578
579
580           /*
581           if(detectouAsteroide(jorj, a))
582             modo_jogo = BATALHA;
583           */
584
585         }
586
587         if(chegouObjetivo(jorj))
588           modo_jogo = FIM;
589
590       }
```

```
590       }
591       else if(modo_jogo == BATALHA){
592         desenhaCenarioBatalha();
593         desenhaHP(asts,indicador_asteroide,jorj,asts[indicador_asteroide].lvl);
594
595           if(jorj.acao == FUGIR && jorj.executar == 1){
596             if(randInt(1,10) >= 5){
597
598               if(jorj.direcao==CIMA)
599                 jorj.y += distM;
600               if(jorj.direcao==BAIXO)
601                 jorj.y -= distM;
602               if(jorj.direcao=DIR)
603                 jorj.x -= distM;
604               if(jorj.direcao==CIMA)
605                 jorj.x += distM;
606
607               modo_jogo = NAVEGACAO;
608               velocidade = 0;
609               jorj.acao = 0;
610             }else{
611               jorj.acao = MONSTRO;
612               jorj.executar = 0;
613             }
614           }
615
616           if(jorj.acao == ATACAR && jorj.executar == 1){
617             al_draw_filled_circle(X_kek1 - velocidade, Y_kek1, 15, al_map_rgb(252,15,192));
618             velocidade += 15;
619
620             if(X_kek1 - velocidade < X_kek2 + 68){
621               velocidade = 0;
622               jorj.acao = MONSTRO;
623               jorj.executar = 0;
624               asts[indicador_asteroide].vida -= DANO_ATAQUE;
625             }
626           }
627
628           if(jorj.acao == ESPECIAL && jorj.executar == 1){
629             al_draw_filled_circle(X_kek1 - velocidade, Y_kek1, 40, al_map_rgb(252,15,192));
630             velocidade += 10;
631
632             if(X_kek1 - velocidade < X_kek2 + 68){
633               velocidade = 0;
634               jorj.acao = MONSTRO;
635               jorj.executar = 0;
636               asts[indicador_asteroide].vida -= DANO_ATAQUE_ESP;
637             }
638           }
```

```
638             }
639
640             if(jorj.acao == MONSTRO && jorj.executar == 0){
641               al_draw_filled_circle(X_kek2 +200+ velocidade, Y_kek2, 20, al_map_rgb(255, 255, 255));
642               velocidade += 15;
643
644               if(X_kek2 +200+ velocidade > X_kek1 +100){
645                 velocidade = 0;
646                 jorj.acao = 0;
647                 jorj.vida -= asts[indicador_asteroide].dano;
648
649                 if(jorj.vida <= 0)
650                   modo_jogo = FIM;
651               }
652               if(asts[indicador_asteroide].vida <= 0){
653                   if(asts[indicador_asteroide].lvl == 1)
654                     jorj.pontuacao += 50;
655                   else if(asts[indicador_asteroide].lvl == 2)
656                     jorj.pontuacao += 100;
657                   else if(asts[indicador_asteroide].lvl == 3)
658                     jorj.pontuacao += 150;
659
660                   velocidade = 0;
661                   jorj.acao = 0;
662                   modo_jogo = NAVEGACAO;
663                 }
664               }
665           }
666           else if(modo_jogo == FIM){
667             al_clear_to_color(al_map_rgb(0,0,0));
668             if(jorj.pontuacao > jorj.recordeCod){
669               recordeArq = fopen("recorde.txt", "w");
670               fprintf(recordeArq, "%d", jorj.pontuacao);
671               jorj.recordeCod = jorj.pontuacao;
672               fclose(recordeArq);
673             }
674               desenhaCenarioGanhou(jorj);
675
676             if (jorj.vida <= 0){
677               desenhaCenarioMorreu(jorj);
678             }
679           }
680
681           //atualiza a tela (quando houver algo para mostrar)
682           al_flip_display();
683
684         }
685         //se o tipo de evento for o fechamento da tela (clique no x da janela)
686         else if(ev.type == ALLEGRO_EVENT_DISPLAY_CLOSE) {
687           playing = 0;
688         }
```

```
688         }
689
690         //se o tipo de evento for um pressionar de uma tecla
691         else if(ev.type == ALLEGRO_EVENT_KEY_DOWN) {
692           movementFlag = 1;
693           movementKey = ev.keyboard.keycode;
694
695         }
696         else if(ev.type == ALLEGRO_EVENT_KEY_UP){
697           movementFlag = 0;
698         }
699
700         if(movementFlag){
701           if(modo_jogo == NAVEGACAO)
702             playing = processaTeclaNaveg(&jorj, movementKey);
703         }
704         else{
705           if(modo_jogo == BATALHA)
706             processaTeclaBatalha(&jorj, ev.keyboard.keycode);
707         }
708
709       } //fim do while
710
711       //procedimentos de fim de jogo (fecha a tela, limpa a memoria, etc)
712       al_destroy_timer(timer);
713       al_destroy_display(display);
714       al_destroy_event_queue(event_queue);
715
716       return 0;
717     }
```

Linhas 711 a 716: procedimentos de fim de jogo.