Documentação TP Allegro

Aluno: André Luiz Abranches Moreira

N° de matrícula: 2019056741

Professor: Pedro O. S Vaz de Melo

Disciplina: Programação e Desenvolvimento de Software I

Como o jogo funciona?

O jogo tem funcionamento parecido com o Guitar Hero original, pois ele desenha círculos de diferentes cores na tela e o jogador deve apertar a tecla correspondente em um momento determinado para conseguir pontos. Esse momento é quando o círculo se encontra na área crítica, antes de chegar ao fim da tela. A criação das notas é feita de forma aleatória, possuindo 20 notas por jogo. As teclas que controlam o jogo são A (pressiona a nota verde), S (pressiona a nota vermelha), D (pressiona a nota amarela), J (pressiona a nota azul), K (pressiona a nota laranja) e ESC (finaliza o jogo a qualquer momento). Além disso a cada acerto, soma-se 1 ponto à pontuação e cada erro diminui-se 2. Neste jogo é possível aumentar a dificuldade, alterando o número de pistas para as notas caírem, podendo variar de 3 a 5 pistas.

Menu

Assim que jogo carrega ele pede que o jogador selecione a dificuldade apertando 1, 2 ou 3 para fácil, médio ou difícil, respectivamente, a partir disto o jogo em si é criado com base na escolha, criando as pistas e as bolinhas de acordo com a dificuldade.

Descrição do código

De início é criada uma nova estrutura de dados chamada Tile, a qual armazena coordenadas do centro x e y de cada nota musical, a posição ocupada no vetor, o valor da pista em que deve descer, um marcador para sinalizar a última nota do vetor e uma variável genérica que represente a cor do círculo e uma variável que indica a existência da nota para ser desenhada. Logo em seguida, inicializa-se um vetor de 20 posições desse novo tipo de dado.

A primeira ação realizada nesse vetor é zerar todos os status que indicam a existência, para que nenhum círculo seja criado, após isso, há um sorteio de posição para a nota gerada na próxima função, variando de acordo com a dificuldade, permitindo que exista 3, 4 ou 5 círculos.

```
void zera_tiles(Tile *tile_novo){
```

```
tile_novo -> status = 0;
}
```

A segunda ação é criar a nota, se o status dela for 0, ela recebe o valor da pista sorteado e a variável que indica o fim do vetor recebe 1 quando está na posição 19, as demais recebem 0. Nessa função a nota recebe seu valor de x e y, este recebe um valor padrão e aquele recebe um valor de acordo com o número de pistas selecionados pelo jogador.

```
void criaTile(Tile *tile_novo, int trilha, int ordem){
    if (tile_novo -> status==0){
        tile_novo ->lane = trilha;
        tile_novo -> id =ordem;
        tile_novo -> ultimaTile = 0;
        if (tile_novo-> id ==19)
            tile_novo -> ultimaTile = 1;
        tile_novo -> y =  SKY_H/2;
        if(NUM_LANES==5){  \\ dificuldade difícil
            if (tile_novo ->lane == 0)
                tile_novo ->x = 180;
            else if (tile_novo ->lane == 1)
                tile_novo ->x = 330;
            else if (tile_novo ->lane == 2)
                tile_novo ->x = 480;
            else if (tile_novo ->lane == 3)
                tile_novo ->x = 630;
            else if (tile_novo ->lane == 4)
                tile_novo ->x = 780;
        }
        else if(NUM_LANES==4){  \\ dificuldade media
            if (tile_novo ->lane == 0)
                tile_novo ->x = 255;
            else if (tile_novo ->lane == 1)
                tile_novo ->x = 405;
            else if (tile_novo ->lane == 2)
                tile_novo ->x = 555;
            else if (tile_novo ->lane == 3)
```

```
                    tile_novo ->x = 705;
                }
                else if(NUM_LANES==3){ \\dificuldade fácil
                    if (tile_novo ->lane == 0)
                        tile_novo ->x = 330;
                    else if (tile_novo ->lane == 1)
                        tile_novo ->x = 480;
                    else if (tile_novo ->lane == 2)
                        tile_novo ->x = 630;
                }
            }
        }
```

Lógica

A lógica do jogo é baseada em 2 temporizadores, um chamado timer e outro chamado contador, este determina quando o status da nota recebe 1 e aquele determina que a nota seja desenhada se o status dela for 1, logo ela é desenhada nos pontos respectivos e alterando o valor de y a cada passagem pelo evento timer. A próxima etapa é a checagem das notas, ela checa se a nota passou do limite da tela e em caso afirmativo, ela muda o status para 0 e não desenha a nota na tela, caso o valor da ultima bolinha seja 1, a variável playing que mantém o jogo em loop recebe 0 e parte para as rotinas de finalização.

```
if(ev.type == ALLEGRO_EVENT_TIMER) {
            draw_scenario(display);
            if( score <= -20)
                playing = 0;
            if (ev.timer.source == timer ){
                for(n=0;n<20;n++){
                    DrawTile(&Vet_Tile[n]);
                }
                al_flip_display();
                for(n=0;n<20;n++){
                    ChecaTile(&Vet_Tile[n], &ultimabolinha);
                }
                playing = ultimabolinha;
```

```
                al_flip_display();
            } \\fim do if == timer


            lse if(ev.timer.source == contador && w<20){
                vivaTile(&Vet_Tile[w]);
                al_flip_display();
                i++;
            }
}\\ fim do evento alegro timer
```

Caso o evento captado seja referente ao pressionamento de alguma tecla, abre-se uma sequência de if e else if que dependendo da tecla pressionada ele analisa o vetor de notas, que possua a pista determinada pela tecla apertada, por exemplo A é a pista 0, então ele checa se dentro do vetor existe nota desenhada e se ela possui o valor de y entre o fim da tela e a área crítica pré-determinada. Caso exista, uma variável recebe 1 e permite que a pontuação seja acrescida +1 ponto, caso não exista uma nota com esses parâmetros a pontuação decresce 2 pontos.

```
        for(i=0;i<TAM_TILES;i++){
if (Vet_Tile [i].status ==1 && Vet_Tile [i].lane == 0 /*&& Vet_Tile[i].y>= 505 &&
Vet_Tile[i].y <=640*/){
        ChecaPosicao(Vet_Tile);
        ponto=1;
        al_flip_display();
} \\ fim do if
}  \\ fim do for
        if(ponto==1)
            score++;
        else{
            al_rest(0.08);
            score-=2;
        }
```

Rotinas de finalização

Caso seja apertado ESC durante o jogo ou o jogador atinja -20 pontos ou a última nota atinja o pixel 640 no eixo y ou seja destruída, o jogo sai do looping infinito de jogo, abre um documento .txt que armazena o maior recorde já atingido, armazena em uma variável inteira e compara com a pontuação atingida, caso a pontuação atual seja maior que o recorde, o jogo mostra na tela o novo recorde, seu recorde atual e salva no documento .txt o valor atingido nessa partida, caso a pontuação seja menor que o recorde, o programa apresenta sua pontuação na partida e o recorde lido do .txt, em seguida ele escreve o recorde no documento .txt, fecha o arquivo e segue destruindo as variáveis do tipo ALLEGRO criadas pelo programa.

```
FILE *recorde = fopen("recorde.txt", "r");
        fscanf(recorde, "%d", &record);
        FILE *rec = fopen("recorde.txt","w");
        if (score > record){
                record = score;
                al_clear_to_color(al_map_rgb(230,240,250));

                sprintf(my_text, "Your score: %d", score);
                al_draw_text(size_32, al_map_rgb(200, 0, 30), SCREEN_W/3, SCREEN_H/2,
0, my_text);

                sprintf(imp_recorde, "New record: %d", record);
                al_draw_text(size_32, al_map_rgb(200, 0, 30), SCREEN_W/3, SCREEN_H/3,
0, imp_recorde);
        }  \\ fim do if
else {
        al_clear_to_color(al_map_rgb(230,240,250));
        sprintf(my_text, "Your score: %d", score);
        al_draw_text(size_32, al_map_rgb(200, 0, 30), SCREEN_W/3, SCREEN_H/2, 0,
my_text);
        sprintf(imp_recorde, "Record: %d", record);

        al_draw_text(size_32, al_map_rgb(200, 0, 30), SCREEN_W/3, SCREEN_H/3, 0,
imp_recorde);

        } \\ fim do else
```

```
            //reinicializa a tela
            al_flip_display();
    al_rest(2);
            fprintf(rec,"%d", record);
rotinas de finalização do alegro e fechamento dos arquivos
   return 0;
} \\ fim do while(playing)
```