DOCUMENTAÇÃO TP ALLEGRO - CANDY CRUSH

Nome: Caio César Silva

Disciplina: Programação e Desenvolvimento de Software I

Professor: Pedro O.S. Vaz de Melo

1 - Introdução

O trabalho prático da disciplina de Programação e Desenvolvimento de Software I consiste na criação de uma versão semelhante ao Candy Crush, onde seria usado a linguagem C e a biblioteca Allegro. O jogo deveria constar com as funcionalidades básicas do jogo, e caso desejasse, seria acrescido alguns extras (tais funcionalidades básicas serão explicadas mais abaixo). O jogo funciona da seguinte forma: ao iniciar o jogo,o jogador conta com 10 jogadas se inicia com 0 pontos. Para realizar uma jogada, o jogador deve clicar e segurar com o botão esquerdo do mouse em um doce e arrastá-lo para onde quer realizar a troca, que só pode ser feita com doces na mesma linha ou coluna, com distância igual a 1. Após realizar a troca, o número de jogadas é decrescido de 1. Se o movimento gerar uma sequência igual a três, os doces dessa sequência são desativados, e sobem para o topo da tela de jogo. Ao atingir zero jogadas é exibida uma tela com a pontuação e o recorde do jogador. Caso a pontuação seja maior que o atual recorde do jogo, é exibido a mensagem “Novo Recorde!”, caso contrário é exibido o atual recorde do jogo. O jogo que criei possui uma estética nostálgica semelhante ao jogo de SNES: The legend of Zelda - A Link to the past, com fundo pixelado e uma das trilhas sonora do jogo.

2 - Bloco Main

O bloco Main é responsável por inicializar o display, a trilha sonora, a execução do plano de fundo, o timer e a fila de eventos retornando uma mensagem caso haja algum erro. Além disso, é no main que são executadas os procedimentos e as funções de lógica do jogo. É nesse bloco que é finalizado o jogo, quando o número de jogadas for igual a zero ou o usuário clicar no botão para fechar a janela do jogo.

3 - Estrutura de dados, funções e procedimentos

3.1 - Estrutura de dados:

O display do jogo é definido com a resolução de 640x480, tendo uma matriz de 9x6, onde cada célula contém um doce. Os doces são definidos como structs, possuindo um valor inteiro para o tipo (que pode variar de 1 a 4), uma cor RGB e caso o doce se encontre em uma sequência de 3, o tipo dele é definido como 0.

3.1 Funções e Procedimentos:

- ***int registraRecorde(int pontuacao, int *recorde):*** recebe a pontuação e o recorde atual como parâmetro, abre o arquivo recorde.txt para leitura e caso haja algum valor no arquivo atribui o valor no parâmetro recorde, e se a pontuação for maior do que o recorde, atualiza o valor do arquivo para a pontuação atual, retorna 1 caso isso ocorra e 0 caso contrário.
- ***void initCandies():*** preenche uma matriz M 9x6 com um valor randômico de tipo (1 a 4), ativo como 1 e uma cor de RGB randômico de 1 a 255.
- ***int getXCoord(int col):*** retorna a coordenada no display ao multiplicar o número da coluna recebida como parâmetro com a largura padrão das células.
- ***int getYCoord(int lin):*** retorna a coordenada no display ao multiplicar o número da linha recebida como parâmetro com a altura padrão das células, e somar com a altura do placar, que é de 64px.
- ***void desenhaCandy(Candy c, int lin, int col):*** recebe um objeto da struct doce, e os número de linha e coluna. De acordo com o tipo gerado, desenha um doce com um formato determinado para aquele tipo (1 = retângulo, 2 = retângulo com bordas redondas, 3 = elipse e 4 = triângulo).
- ***void draw_scenario(ALLEGRO_DISPLAY *display):*** desenha o cenário do jogo, incluindo o placar, os doces e a tela de fundo.

- ***void getCell(int x, int y, int *lin, int *col):*** Define o número da linha e da coluna com base no valor do eixo y e x recebidos como parâmetro, e os divindo com a altura e largura da célula respectivamente (o y é subtraído com a altura do placar, para retornar a coordenada exata).
- ***void swap(int lin_src, int col_src, int lin_dst, int col_dst):*** Recebe o número da linha e coluna de origem e destino, trocando o item da origem com o destino na matriz M se a distância for de uma “casa”, e reduz uma jogada ao realizar a função.
- ***int distancia(int lin1, int col1, int lin2, int col2):*** executa a fórmula de distância euclidiana para descobrir a distância entre dois pontos, fazendo a raiz quadrada da soma das potências de 2 das diferenças entre a linha 1 e a linha 2, e a coluna 1 e a coluna 2.
- ***void criaMatrizAuxiliar(int matriz[N_LINHAS][N_COLS]):*** Cria uma matriz 9x6 preenchidas com 0.
- ***int identificaSequencia():*** Percorre a matriz pelas linhas e colunas e se a houver uma sequência de 3 doces do mesmo tipo, na mesma coluna ou linha, o número 1 é atribuído a sequência e esta é retornada, caso não haja sequência, esta recebe 0.
- ***void zeraSequencia():*** Percorre a matriz pelas linhas e colunas e se a houver uma sequência de 3 doces do mesmo tipo, na mesma coluna ou linha, a matriz auxiliar recebe nestes M(aux)[i][j], o número 1 no lugar do 0. Ao final da função onde houver 1 na matriz auxiliar, as mesmas posições na matriz do jogo são definidas como tipo 0, “sumindo” da matriz.
- ***void sobeZeros():*** Percorre a matriz pelas colunas, e se o tipo for 0 e houver outro doce com tipo diferente de 0 acima dele, os tipos entre esses doces são trocados, fazendo com que os doces do tipo 0 sempre fiquem nas posições acima dos outros.
- ***void imprimeMatriz()*****:** Percorre a matriz imprimindo o tipo de cada doce da matriz, serve para teste.
- ***int calculaPontuacao():*** Percorre a matriz, e caso seja feita uma sequência, retorna o inteiro sequência de acordo com a quantidade de

doces transformados em 0 (por exemplo, 6 doces na sequência, a função retorna 6).

4 - Planejamento do jogo (parâmetros a serem implantados em futuros pacotes de atualização)

- Não são identificadas, nem zeradas sequências maiores que 3, o objetivo do jogo seria encontrar qualquer sequência maior ou igual a 3.
- Permitir a troca de doces somente caso seja realizado uma sequência.
- Não permitir a troca de doces com doces do tipo 0, e caso o MOUSE_UP ocorra fora do display.
- Implementar sistema de dificuldade.
- Descer doces aleatórios no lugar dos que tem tipo 0 até determinada quantidade, de acordo com a dificuldade do jogo.