Universidade Federal de Minas Gerais

DOCUMENTAÇÂO – TP ALLEGRO

“Final Fantasy VI”

Nome: João Vitor Ferreira

Disciplina: Programação e Desenvolvimento de Software I

Professor: Pedro O.S. Vaz de Melo

1 – Introdução

No meu trabalho eu fiz algumas adaptações nas personagens do jogo, este baseado no sexto jogo da franquia Final Fantasy, o modelo usado como base foi a versão original, lançada para o Super Nintendo. A partir dela foi desenvolvida uma versão simplificada dele. Houve algumas adaptações nas personagens do jogo, e os personagens do mundo do Mario substituíram os originas, mais precisamente do Super Mario Bross 3, lançada originalmente para a mesma plataforma, protagonista (herói) foi substituído pelo Mario, com o Power Up de fogo. Os inimigos (monstros) foram substituídos por 5 tipos diferentes presentes no Mario: o Bowser, Larry Koopa, Lemmy Koopa, Ludwig Von Koopa, Roy Koopa e Wendy Koopa.

A maneira em que o jogo está desenvolvido é dividida em 2 partes, na primeira o herói irá explorar o mapa com o objetivo de chegar a um castelo, nesse caminho haverá desafios, monstros escondidos. Caso o jogador encontre com algum destes monstros o modo de batalha será ativado. Neste modo o herói deve batalhar com o monstro até a morte de um dos dois, ou o herói pode tentar fugir. Batalhando o herói tem 2 tipos de ataques possíveis um ataque que causara dano um dano aleatório e outro um especial que causara outro dano aleatório, sendo a soma de dois números aleatórios. Após o herói atacar será a vez do monstro revidar, seu dano será aleatório como o ataque normal do herói. Caso o herói decida fugir ele terá 75% de chance de a fuga ser sucedida, caso ele não consiga o monstro fara um ataque. No caso de o herói morrer o jogo acabara, com o herói perdendo, se o monstro morrer o herói volta para sua exploração inicial e terá recebido uma pontuação referente a quantidade de dano que causou. Outra maneira de finalizar o jogo, está de forma vencedora é chegar ao destino. Logo o herói deverá chegar ao destino vivo e preferencialmente com a maior quantidade de pontos possível.

2 – Controles

Os controles diferem entre os modos:

Navegação

SETA CIMA – Move o herói para cima

SETA ESQUERDA – Move o herói para esquerda

SETA DIREITA – Move o herói para direita

SETA BAIXO – Move o herói para baixo

Batalha

J – Marca a opção de ataque

K – Marca a opção de especial

L – Marca a opção de fugir

ENTER – Confirma a escolha pelas teclas acima

3 – Implementação

Bibliotecas

As bibliotecas implementadas no programa foram: stdio.h, stdlib.h, math.h, time.h, string.h. Alem das padrões do Allegro: allegro5/allegro.h, allegro5/allegro_font.h, allegro5/allegro_ttf.h, allegro5/allegro_primitives.h, allegro5/allegro_image.h

Estrutura de Dados

Foram definidas variáveis globais, structs e constantes

Constantes

NAVEGACAO – define o número que representa o modo de navegação, neste caso é o 0.

BATALHA – define o número que representa o modo de batalha, neste caso é o 1.

TAM_HEROI – define o tamanho do herói, este que é 20

PASSO – define o tamanho do passo do herói, este que é 30

TAM_OBJETIVO – define o tamanho do objetivo, este que é 40

MAX_DIST – define a distância máxima entre o herói e um inimigo, definida como 20

ATACAR – define o número que representa o ataque na batalha, o 0.

ESPECIAL – define o número que representa o especial na batalha, o 1.

FUGIR – define o número que representa o modo de batalha, neste caso é o 1. TAM

– representa o tamanho de caracteres que um vetor char recebera

variáveis globais

danoHeroi – recebe o resultado de uma função rand, que representa o ataque do herói

danoMonstro – recebe o resultado de uma função rand, que representa o ataque do monstro

X_OPCOES – valor de x do menu de batalha Y_OPCOES

– valor de y do menu de batalha

structs

Heroi – Estrutura que define os parâmetros do herói, tais quais: x, y, vida, pontos, posição, ação, executar, x_old, y_old e ataque

Monstro – Estrutura que define os parâmetros do monstro, tais quais: x, y, vida, ataque, tipo.

Menu – Estrutura que define os parâmetros do menu, tais quais: opção, int confirma.

Funções

São separadas por funcionalidade e finalidade, tais como, inicialização, processamento na navegação e processamento na batalha. Ao todo foram utilizadas 24 funções

inicialização

1. void initHeroi(Heroi *h) – Essa função é responsável por atribuir valores para os parâmetros definidos na struct Heroi, a posição x e y é fixa e representada pelos valores de 1 e 680, respectivamente, atribui um valor para a vida do herói que é o valor fixo de 450 mais um número aleatório entre 1 e 100.

2. void initMonstro(Monstro *m, int a) – Essa função é responsável por atribuir valores para os parâmetros definidos na struct Monstro, a posição x e y é variável e é um número maior que 30 e menor que os limites da tela. A vida é definida por um valor fixo de 10 mais um número aleatório entre 1 e 100.

3. void iniciaGlobais() Essa função tem o trabalho de “desenhar” o herói no cenário de navegação, além de processar a sua mudança de posição ao trocar a representação dele.

processamento na navegação

4. void desenhaHeroiNavegacao(Heroi *h) – Essa função tem o trabalho de “desenhar” o herói no cenário de navegação, além de processar a sua mudança de posição ao trocar a representação dele.

5. void desenhaCenarioNavegacao(Heroi *h) – Nessa função é carregado o cenário de navegação juntamente da representação do objetivo. Posiciona no canto superior esquerdo a pontuação do herói. (Imagem 1)

6. void processaTeclaNavegacao(Heroi *h, int tecla) – Essa função processa a navegação do herói através das teclas $(\uparrow, w, \downarrow, s, \rightarrow, d, \leftarrow, a)$, que representam o movimento para cima, baixo, direita e esquerda. Nela também é processado a posição que ira se alterar a pressionar alguma dessas teclas. E redesenha o herói no lado inverso caso ele ultrapasse os limites da tela

7. int chegouObjetivo(Heroi h) – Processa se o herói chegou ao objetivo, e retorna 1 que finaliza o jogo.

8. void dist(int x1, int y1, int x2, int y2) – retorna o cálculo da distância euclidiana entre o herói e monstro

9. int detectouMonstro(Heroi h, Monstro m) – Processa se o herói encontrou algum monstro através do resultado da distância euclidiana

processamento na batalha

10. void desenhaBatalha1(Menu *menu, Heroi h, Monstro *m) – Desenha a batalha do herói com o monstro 1, além de desenhar o menu, pontos e vida do herói e monstro (Imagem 2)

11. void desenhaBatalha2(Menu *menu, Heroi h, Monstro *m) – Desenha a batalha do herói com o monstro 2, além de desenhar o menu, pontos e vida do herói e monstro (Imagem 3)

12. void desenhaBatalha3(Menu *menu, Heroi h, Monstro *m) – Desenha a batalha do herói com o monstro 3, além de desenhar o menu, pontos e vida do herói e monstro (Imagem 4)

13. void desenhaBatalha4(Menu *menu, Heroi h, Monstro *m) – Desenha a batalha do herói com o monstro 4, além de desenhar o menu, pontos e vida do herói e monstro (Imagem 5)

14. void desenhaBatalha5(Menu *menu, Heroi h, Monstro *m) – Desenha a batalha do herói com o monstro 5, além de desenhar o menu, pontos e vida do herói e monstro (Imagem 6)

15. void initMenu(Menu *menu) – Essa função é responsável por atribuir valores para os parâmetros definidos na struct Menu. (Imagem 2)

16. void processaTeclaBatalha(Heroi *h, int tecla, Menu *menu) – É responsável por processar qual tecla foi pressionada na batalha e mudar o ponteiro que indica, qual opção foi escolhida, as teclas J, K, L representam o ataque, especial e fuga, respectivamente,

17. int ataqueHeroi(Heroi *h) – Esta função processa e retorna a quantidade de dano do herói, referente ao ataque, sendo um número aleatório até 60 mais um bônus fixo de 5.

18. int ataqueMonstro(Monstro *m) – Esta função processa e retorna a quantidade de dano do monstro, sendo um número aleatório até 50 mais um bônus fixo de 3.

19. int especialHeroi(Heroi *h) – Esta função processa e retorna a quantidade de dano do herói, referente ao especial, sendo a soma de dois números aleatórios até 50 mais um bônus fixo de 20.

20. int fuga() – Esta função processa e retorna um número aleatório até 100.

21. void animacaoAtaqueHeroi(Heroi *h, Monstro *m) – A animação de ataque do herói é processada e executada. (Imagem 3)

22. void animacaoAtaqueMonstro(Heroi *h, Monstro *m)) – A animação de ataque do monstro é processada e executada. (Imagem 5 e 6)

23. void animacaoEspecialHeroi(Heroi *h, Monstro *m) – A animação de especial do herói é processada e executada. (Imagem 4)

24. int processaAcaoHeroi(Heroi *h, Monstro *m) – Esta função é responsável por administrar as ações na batalha, ou seja, ela processa a fuga, ataque e especial. Na fuga ela chama a função fuga, caso o resultado dela seja um valor acima de 35 a fuga do herói será feita, e o herói voltara para a navegação numa posição antes de colidir com o monstro, caso contrário ocorrera um ataque do monstro, e ela chama a função ataqueMonstro que calculara a quantidade de dano causada pelo monstro e a função animacaoAtaqueMonstro, depois ela subtrai da vida do herói o danoMonstro. No ataque e ela chama a função ataqueHeroi que calculara a quantidade de dano causada pelo herói e a função animacaoAtaqueHeroi, depois ela subtrai da vida do monstro o danoHeroi, após isso ela chama a função ataqueMonstro que calculara a quantidade de dano causada pelo monstro e a função animacaoAtaqueMonstro, depois ela subtrai da vida do herói o danoMonstro. No especial e ela chama a função especialHeroi que calculara a quantidade de dano causada pelo herói e a função animacaoAtaqueHeroi, depois ela subtrai da vida do monstro o danoHeroi, após isso ela chama a função ataqueMonstro que calculara a quantidade de dano causada pelo monstro e a função animacaoAtaqueMonstro, depois ela subtrai da vida do herói o danoMonstro.

Programa Principal

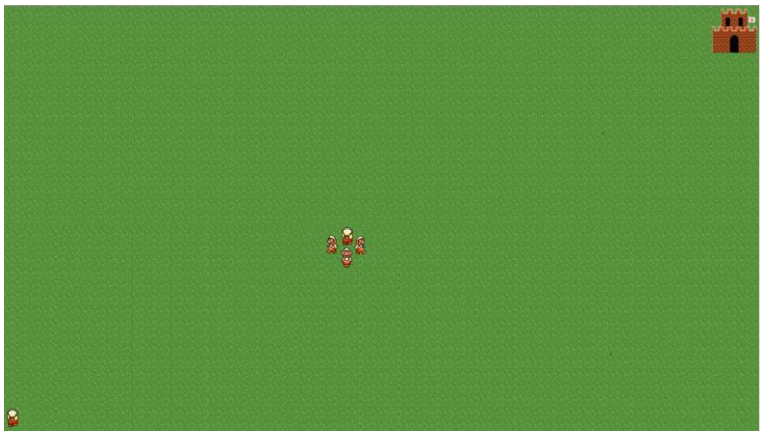
No programa principal somente é usada uma função que é a main. Nela que todas as outras funções são chamadas, os procedimentos de inicialização do alegro são chamados e os processos de fim de jogo. Inicialmente há um bloco de rotinas de

inicialização que é padrão, logo após ele se inicia o bloco do jogo. Começa definindo o modo de jogo, iniciando as variáveis globais, monstros e herói. Após dentro de um loop, são separados o modo de navegação e batalha.

No modo de navegação desenha=se o cenário e o herói no ambiente de navegação, e processa se ocorreu alguma colisão entre o herói e um dos 21 inimigos. Nesta parte também é processado se o herói chegou no seu destino, caso isso ocorra é iniciado a rotina de finalização de vitória.

Passando pelo modo de batalha, processa-se em qual monstro houve o encontro e inicia-se o cenário e altera o modo de jogo, também é processado se o herói morreu, caso ocorra é iniciado a rotina de finalização de Game Over. Após isso iniciam-se as rotinas de finalização.

4 – Imagens e representações



1 – Representação do modo de Navegação

1.1 – Posição em que o Herói inicia o jogo

1.2 – Objetivo do jogo

1.3 – Representação das posições que o herói pode assumir a depender da direção em que ele irá.

2 – Representação do modo de batalha

2.1 – Batalha com o monstro 1

2.2 – Representação do Menu3 – Representação do modo de batalha

3 – Representação do modo de batalha

3.1 – Batalha com o monstro 2

3.2 – Representação do ataque do Herói

4 – Representação do modo de batalha

4.1 – Batalha com o monstro 3

4.2 – Representação do especial do Herói

5 – Representação do modo de batalha

5.1 – Batalha com o monstro 4

5.2 – Representação do ataque do Monstro

6 – Representação do modo de batalha

6.1 – Batalha com o monstro 5

6.2 – Representação do ataque do Monstro