Trabalho Prático - O Homem Bomba

Pedro O.S. Vaz de Melo

May 11, 2016

1 Descrição do Problema

O objetivo deste trabalho é fazer com que o aluno utilize as técnicas de programação aprendidas na disciplina para desenvolver um jogo eletrônico gráfico semelhante ao clássico *Bomberman*. No jogo, o usuário controla um personagem (o *bomberman*) que tem como objetivo eliminar os seus inimigos a partir de bombas. O *bomberman* deposita bombas pelo cenário que, depois de um tempo, explodem e matam todos que estão dentro do seu raio de explosão. Alémd isso, ele deve ter um estoque finito de bombas, que é recuperado uma vez que as bombas explodem, ou seja, se o estoque for de 3 bombas, no máximo 3 bombas devem estar na tela ao mesmo tempo. Se o jogador encostar em um inimigo, ele deve sofrer algum tipo de dano que, no caso mais simples de ser implementado, o leva à morte e o jogo acaba. Pelo menos um dos monstros implementados não deve se mover aleatoriamente, ou seja, ele deve perseguir o jogador. O jogo acaba quando todos os inimigos estiverem mortos.

Além disso, o jogo deverá registrar algum tipo de pontuação. No caso mais simples, esse pontuação pode ser o tempo que o jogador levou para matar todos os inimigos. Ao final do jogo, caso o usuário tenha vencido, deverá ser exibida uma tela informando a pontuação do usuário e o recorde atual. Caso a pontuação do usuário seja melhor que o recorde atual, um texto com essa informação deve ser exibido para o usuário. Este trabalho tem um valor total de 20 pontos. Execute o arquivo bomber.exe para um exemplo de jogo que receberá o total de pontos da implementação mais 1 ponto extra.

2 Critérios de Avaliação

2.1 Solução Apresentada

Os seguintes itens devem ser implementados:

- Movimentação fluida do personagem e dos inimigos, ou seja, sem muitos saltos (teletransportes) na tela;
- Pelo menos três inimigos na tela;
- Implementação do estoque de bombas, com no mínimo três bombas;
- Contagem de pontos;
- Exibição e armazenamento do recorde;

- Colisão do *bomberman* com os seus inimigos;
- Dano (no caso mais simples, morte) aos inimigos e ao *bomberman* a partir das bombas;
- A bomba deve ter um raio de explosão que excede o seu contorno.

2.2 Documentação

Deve conter o Manual de Uso, que descreve como operar o jogo, e detalhes da implementação, que descreve brevemente os trechos de código desenvolvidos por você.

2.3 Conhecimento do Código

Conhecimento do aluno sobre o código apresentado será verificado via entrevista em laboratório. Sua nota total será multiplicada pela sua nota da prova oral, que vale 1. Assim, se você tirar 0.5 na prova oral, sua nota será dividia por 2.

2.4 Pontos Extras

Além dos 20 pontos, o professor pode atribuir até 10 pontos a mais caso o aluno implemente extras, tais como:

- Fazer com que os inimigos atirem no jogador;
- Implementar fases;
- Implementar obstáculos;
- Criar *addons* e *power-ups* que podem, por exemplo, dar ao jogador bombas mais poderosas;
- Criar tipos diferentes de jogadores e monstros;
- Criar um sistema de vidas para o jogador;
- Criar cenários que interajam com o jogador, por exemplo, uma escada rolante.
- Criar animações sofisticadas;
- Permitir modo de dois jogadores ao mesmo tempo;
- Colocar sons e músicas;
- Criar labirintos;
- Permitir um modo de jogo que indica, com uma linha, os pontos usados para calcular a distância entre o inimigo e a bomba quando este for morto pela bomba. Nesse modo, os pontos indicados devem ser aqueles que produzem, necessariamente, a menor distância entre a bomba e o inimigo. No exemplo `bomber.exe` esse modo está implementado. Se você implementar esse extra, será necessariamente questionado sobre ele na prova oral.

- **Qualquer outro extra que você ache interessante!**

IMPORTANTÍSSIMO: Pontos extras só serão dados aos alunos que obtiveram mais de 50% **dos pontos nas provas, ou seja, mais de** 35 **no somatório das três provas.**

3 Como eu faço?

Apesar da descrição fazer o trabalho parecer complicado, ele é bastante simples. Tudo que o aluno precisa saber para desenvolver este jogo são os conhecimentos adquiridos na disciplina e um pequeno entendimento de desenvolvimento de aplicações gráficas. Assim como são necessárias bibliotecas novas para a utilização de funções não nativas da linguagem C, como a math.h, uma biblioteca também é necessária para que se utilize funções gráficas. Para este trabalho, pede-se que se utilize a biblioteca Allegro5, que fornece inúmeras funções que podem ajudar no desenvolvimento deste trabalho.