Trabalho Prático - \a Hero

Pedro O.S. Vaz de Melo

May 10, 2019

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 *Guitar Hero*. Nesse jogo, "notas musicais" se aproximam do jogador, vindo de cima para baixo da tela, e você deve apertar o botão correspondente à nota quando for o momento de tocá-la. Esse momento ocorre quando a nota está prestes a deixar a tela, sob o que chamaremos a partir de agora de *região crítica*. Além disso, as notas são divididas em faixas verticais, com a ideia de que cada faixa representa uma nota diferente. Se o jogador conseguir tocar a nota, um som é emitido e a sua pontuação aumenta. Se o jogador falhar, ou apertar qualquer botão de forma incorreta, a sua pontuação diminui. O jogo termina quando todas as notas passarem pela tela ou quando o jogador atingir uma pontuação menor que o mínimo permitido, ou seja, a plateia quer que o jogador pare de tocar.

Assim, ao implementar o jogo, você deverá registrar e exibir algum tipo de pontuação, que deve variar de acordo com o sucesso do jogador ao apertar os botões corretos no momento correto. Ao final do jogo deverá ser exibida uma tela informando a pontuação do usuário e o recorde atual. Caso a pontuação do usuário seja maior que o recorde atual, um texto com essa informação deve ser exibido para o usuário e um novo recorde deve ser registrado. Este trabalho tem um valor total de 20 pontos. Execute os arquivos `ghero.exe` para um exemplo de jogo que pode ser implementado. A versão clássica do jogo pode ser jogada em `https://guitarflash.com/`.

2 Critérios de Avaliação

2.1 Solução Apresentada

Os seguintes itens **serão** avaliados:

- Movimentação fluida das animações relacionadas ao jogo, como o movimento de aproximação das notas *(1 ponto)*;
- As notas devem aparecer de forma aleatória[1] (ou de acordo com alguma música, se você se aventurar por esse caminho) *(2 pontos)*;

[1] notas que aparecem a cada x segundos ou notas que aparecem de forma previsível nas faixas **não** estão aparecendo de forma aleatória.

- As notas não podem se sobrepor umas às outras *(2 pontos)*;
- O seu jogo deve ter pelo menos três faixas de notas *(2 pontos)*;
- Quando uma nota for tocada corretamente, algum efeito deve ser produzido no jogo (ex: um `printf("\a")`) *1 ponto*;
- Quando o jogador errar ao tentar acertar uma nota, a animação do jogo deve se alterar de alguma forma à sua escolha *(3 pontos)*;
- O seu jogo deve controlar pelo menos 10 notas, que não precisam estar necessariamente na tela ao mesmo tempo *(3 pontos)*;
- O cenário deve exibir alguma informação do jogo em cima das faixas (ex: o nome do jogo) *(1 ponto)*;
- Contagem e exibição dos pontos *(2 pontos)*;
- Exibição e armazenamento do recorde *(3 pontos)*;
- Documentação *(2 pontos)*.

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 e as estruturas de dados desenvolvidas 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:

- Gerar diferentes tipos de cenários;
- Permitir diferentes quantidades de faixas;
- Permitir diferentes velocidades de notas;
- Permitir diferentes quantidades de notas ao mesmo tempo na tela;
- Colocar sons e músicas;
- Fazer com que as notas apareçam de forma sincronizada com uma (ou mais) músicas;
- Implementar animações sofisticadas para a plateia, para o jogador etc;
- Implementar diferentes tipos de notas (ex: uma nota de longa duração);

- Implementar diferentes tipos de instrumentos;
- Criar *addons* e *power-ups* que podem, por exemplo, dar ao jogador invencibilidade por um tempo;
- Implementar modo de dois jogadores;
- Implementar dinheiro e permitir que o usuário compre músicas, *power-ups* etc;
- **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. Os vídeos abaixo ensinam como instalar a biblioteca Allegro5 em um ambiente Windows com o MingW instalado:

`https://www.youtube.com/watch?v=AezxBP687n8`
`https://www.youtube.com/watch?v=cgqjzJzm00w`

4 Roteiro de Desenvolvimento Sugerido

Como o jogo é complexo, identificar a sequência de funcionalidades que devem ser desenvolvidas pode ser um problema. Assim, a seguir estão descritas etapas de desenvolvimento sugeridas, colocadas em ordem cronológica.

1. Desenhar o cenário do jogo (fácil);
2. Definir e desenhar a *região crítica* (fácil);
3. Implementar a animação de uma nota (círculo colorido) ao longo de uma faixa aleatória (médio);
4. Implementar a animação de várias notas (médio);
5. Criar uma função que identifica quando duas notas estão (ou estarão) sobrepostas (difícil);
6. Fazer com que as notas sejam criadas sem sobreposição (médio);

7. Implementar a verificação da tecla apertada com a posição das notas, aumentando ou diminuindo a pontuação do usuário se ele acertar ou errar, respectivamente (médio);
8. Implementar o sistema de pontuação e o exibir na tela (fácil);
9. Implementar o sistema de armazenamento do recorde (fácil);
10. Divirta-se implementando funcionalidades extras (super divertido);
11. Escrever a documentação (fácil).