Definição do trabalho da M2

Flappy Bird

Data de entrega: 11/05/2022. (até 08:00)
Modalidade: Quatro Integrantes.

DESCRIÇÃO

Flappy Bird é um jogo eletrônico para dispositivos móveis de 2013 desenvolvido em Hanói pelo programador vietnamita Nguyễn Hà Đông e publicado pela GEARS studios. O jogo foi publicado em maio de 2013 para o iPhone 5, e então atualizado para o iOS 7 em setembro de 2013. Em janeiro de 2014, ele ficou no topo da categoria de jogos gratuitos da iTunes App Store chinesa e americana, e mais tarde naquele mês da loja do Reino Unido, onde foi chamado de "o novo Angry Birds". Terminou o mês de janeiro como o aplicativo mais baixado da App Store. Existe uma versão para Windows 8 e Windows 8.1.

Figura 1

FUNCIONAMENTO

O objetivo no jogo é ganhar o maior número possível de pontos, controlando um pássaro (tocando na tela) sem deixá-lo colidir nos canos. Se o pássaro tocar em algum obstáculo - ou se deixar o pássaro cair -, o jogo termina. Sempre que o personagem passa por um conjunto de canos, o jogador ganha um ponto.

Link para o jogo: https://flappybird.io

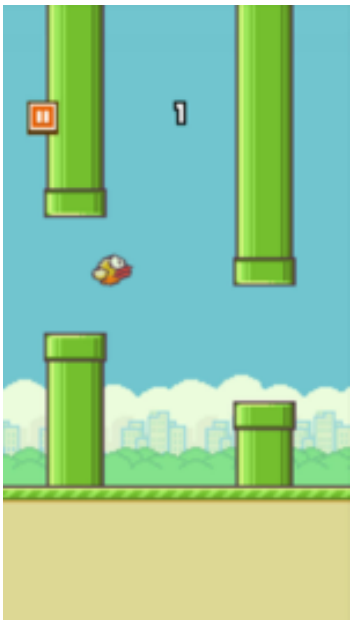

Figura 2

DESENVOLVIMENTO

Agora que você tem uma ideia de como será o jogo, vamos as regras:

- Você recebeu um programa com alguns defeitos que precisam ser resolvidos, ou seja, você deve usar o código fornecido como referência;
- Problemas a serem solucionados
    - O pássaro "deixa rastros": Quando o pássaro é desenhado na sua nova posição, o jogo não apaga o desenho do pássaro da posição anterior;
    - O obstáculo "deixa rastros": O obstáculo é desenhado em sua nova posição, mas o jogo não apaga o desenho da posição anterior do obstáculo;
    - Não é apresentado o placar em lugar nenhum da tela, mas o jogo deveria apresentar uma pontuação que aumenta a cada obstáculo superado; e
    - Não existe tratamento de colisão, o pássaro não colide com o teto, chão ou com os obstáculos.
- Novas funcionalidades a serem implementadas
    - Os obstáculos deverão ser escolhidos aleatoriamente com passagens: no meio, em cima e embaixo; (explicação sobre números aleatórios no final do documento)
    - O jogo deverá apresentar 2 obstáculos simultâneos com uma certa distância entre eles; e
    - A velocidade do jogo deverá aumentar a cada 5 obstáculos superados.
- Sintam-se, a vontade, para ir além dos requisitos do sistema, mas lembrem-se que todas as funcionalidades implementadas deverão ser defendidas na apresentação e que elas também devem respeitar os requisitos e restrições impostas ao trabalho.

Dicas de desenvolvimento:

O código, a seguir, exemplifica o uso das funções `rand()` e `srand()`;

- `rand()` gera um número pseudo-aleatório entre `0` e `RAND_MAX`, mas essa faixa pode ser facilmente alterada com o operador de resto da divisão inteira.
- `srand()` gera uma nova semente aleatória baseada no parâmetro passado entre os parênteses da função. É comum utilizar a função `time()`, pois ela pega o horário do sistema que muda a cada milésimo de segundo. Note que se a função `srand()` não for utilizada a sequência de números pseudo-aleatórios gerados pela função `rand()` será sempre a mesma.

```
1   #include <iostream>
2   #include <time.h> //para habilitar a função time
3   using namespace std;
4
5   int main()
6   {
7       srand(time(NULL)); //semente randomica gerada a partir da hora do sistema
8
9       int numeroAleatorio;
10
11      numeroAleatorio = rand()%10; //O %(mod) coloca os números gerados entre 0 e o resto da divisão-1
12
13      cout << numeroAleatorio << endl;
14
15  }
```

Outros dois comandos bastante úteis no desenvolvimento de programas no console, são os comandos `system("cls")` e `system("pause")`.

- `system("cls")` é um comando que limpa a tela do console (**cl**ear **s**creen). Esse comando é bastante útil, pois em uma tela limpa é mais fácil dar destaque aquilo que se está mostrando no momento.
- `system("pause")` é um comando útil, principalmente quando usado em conjunto com o `system("cls")`, pois ele pausa a execução da aplicação até que o usuário aperte qualquer tecla, bastante útil quando se quer exibir algo antes de limpar a tela para iniciar uma nova execução.

*Os comandos equivalentes ao `system("cls")` e `system("pause")` no linux/MacOS são respectivamente o `system("clear")` e `system("read 0 -p")`.

Obs.: Para o desenvolvimento do código não poderão ser utilizadas variáveis compostas (*arrays, structs*) e funções.

Defesa (Obrigatória)

Durante a defesa serão realizados questionamento sobre o trabalho realizado pelo grupo. A defesa é obrigatória e deverá ser feita pelos integrantes do grupo na aula. Se algum integrante não estiver presente durante a aula de defesa, deverá justificar a falta, o mesmo defenderá posteriormente em data a ser agendada com o professor.

Entregas:

- Postar no repositório criado especialmente para o trabalho no material didático: **Trabalho T2**
- Código fonte desenvolvido: é de responsabilidade do grupo verificar se o arquivo postado é o correto.

Critérios de Avaliação:

---

1. Organização e clareza do código = 5% da nota.
2. Identificação dos autores e Comentários pertinentes e oportunos no código = 10% da nota.
3. Funcionamento correto conforme a especificação = 40% da nota.
4. Recursos da linguagem utilizados = 20% da nota.
5. Apresentação do código = 25% da nota.

Obs.: Todas as notas relativas ao código dependem do desempenho na defesa. Sem a defesa o trabalho terá nota ZERO.