Projeto Space Invaders – parte 1

Programação Orientada a Objetos

Prof. Robson L. F. Cordeiro

5 de outubro de 2020

1 Descrição

O projeto a ser desenvolvido nessa disciplina será a implementação do jogo *Space Invaders* que foi desenhado e programado por Toshihiro Nishikado da empresa Game Taito do Japão em 1978 e continua sendo um dos jogos arcade mais populares de todos os tempos. A Figura 1 a seguir apresenta um exemplo de tela desse jogo.

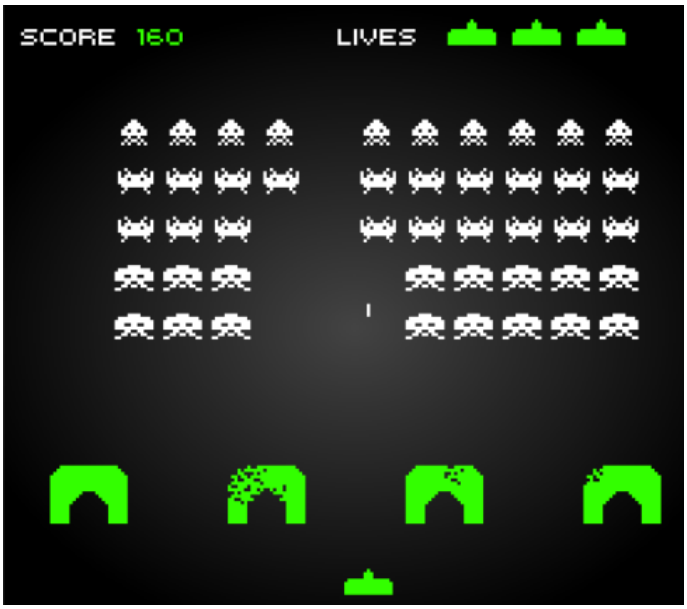

Figura 1: Tela do jogo *Space Invaders*.

As regras de funcionamento desse jogo são descritas a seguir:

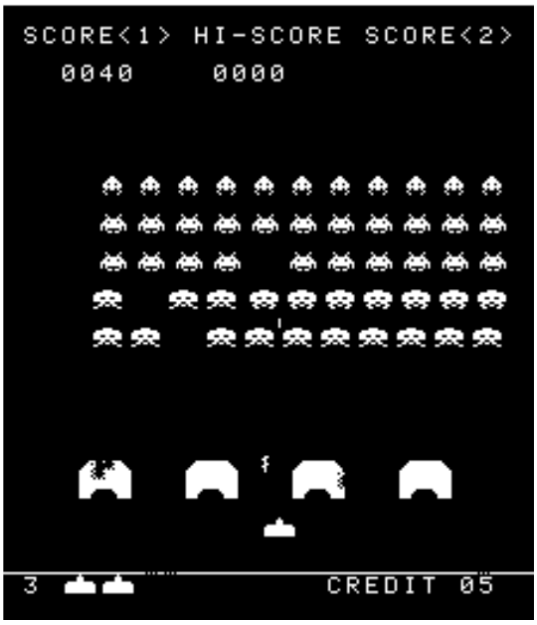

Figura 2: Canhão Protetor.

Em Space Invaders você controla um canhão que defende a terra de invasores do espaço. O canhão fica na parte inferior da tela, e ele pode se mover para a esquerda e para a direita. Ele se defende atirando contra os invasores.

Os invasores se movimentam para a direita ou para a esquerda. Quando um dos flancos toca um lado da tela, eles descem uma linha e se movem para o lado oposto:

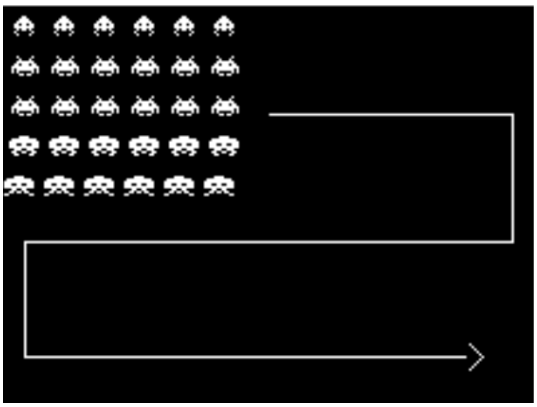

Figura 3: Movimento dos Invasores.

Os invasores também se defendem atirando contra o canhão. Eles se movem mais rápido à medida que vão sendo destruídos; mais adiante vai ser

explicado como eles são destruidos. Existem 3 tipos de invasores e eles ficam organizados como uma matriz. Eventualmente aparece uma nave em cima que se move para o lado oposto da tela e dá pontos especiais para o jogador que a acertar.

Os tiros do jogador, se acertarem a base, a destroem parcialmente, se acertarem uma nave inimiga a destrói imediatamente. Os tiros das naves não acertam naves, mas acertam a base, destruindo ela parcialmente, e quando acertam o canhão o jogador é destruido e ele perde uma vida.

O jogador passa de fase quando todos os invasores da matriz morrem; a cada fase que ele passa os invasores começam uma linha abaixo. O jogador perde o jogo quando ele perde todas as três vidas. As fases são infinitas.

Canhão: Qual a posição inicial do canhão? O que acontece quando o jogador leva um tiro? Ele perde uma vida e fica quanto tempo sem se mexer? Ele fica invencível por um tempo? Por quanto tempo? Quando ele leva um tiro ele volta para a posição inicial? O que acontece quando ele toca um extremo da tela? Ele aparece no outro extremo? Ele para de se mexer? Ele tem tiros infinitos? Se tem, ele pode atirar uma vez e deve esperar até o tiro chegar ao alvo ou pode atirar quantas vezes ele quiser em sequência?

O canhão começa na esquerda, mais ou menos em uns 20% da tela. Quando ele leva um tiro, o jogo para e espera-se até o jogador apertar o botão de atirar e ele volta à posição inicial dele, mas os invasores continuam na posição onde eles estavam. Quando o canhão se move até um extremo da tela ele para e não se move mais. Ele tem tiros infinitos e só pode dar um tiro por vez.

Invasores: Como eles ficam organizados? Qual é o tamanho da matriz? Qual a velocidade deles? Como é a progressão da velocidade à medida que eles vão morrendo? Qual é o critério para eles atirarem?

Os invasores se organizam em uma matriz com 11 colunas e 5 linhas. A velocidade inicial e a progressão é algo que nós vamos ter que experimentar. Mas no jogo, quando sobra o último invasor aparentemente a velocidade dele dobra quando ele se move para a direita; quando ele se move para a esquerda ela diminui um pouco (acho que é para o jogador ter mais chance, fica realmente muito rápido). O critério para eles atirarem me parece ser algo assim: nas primeiras fases um deles atira aleatoriamente, enquanto o outro tiro é sempre da nave que está na coluna em que o jogador está, a medida que as fases vão passando vai aumentando o número de tiros aleatórios.

Barreira: Como a barreira vai sendo destruída? Ela tem seções que aguentam por exemplo 3 tiros? Ou ela é destruída por terreno (tipo *worms*)?

A barreira é destruída por terreno, de acordo com a animação. No jogo original, a barreira é "deletada" de acordo com a animação da explosão, por exemplo: um tiro acerta a barreira, vira uma explosão e o desenho dessa explosão simplesmente sobrepõe o que tinha sido desenhado na barreira.

2 Conteúdo de Entrega

Nessa primeira etapa, o que precisa ser entregue envolve:

- A estrutura de classes que represente os elementos do jogo: canhão, invasores, etc. Lembre-se de criar estruturas hierárquicas de classes abstratas (ou interfaces) e classes concretas;
- O sistema deve ser modularizado em três partes: (1) elementos do sistema (canhão, invasores, etc.); (2) interface gráfica, e; (3) *engine* de funcionamento (movimentação dos elementos, contagem de pontos, etc.). Nessa etapa, o que precisa ser entregue são os itens (1) e (3). A *engine* ainda não precisa estar completa, nessa fase o que precisa ser implementado é o método que faz os invasores se movimentarem;
- Além do código, um diagrama de classes UML, indicando quais as classes desenvolvidas, seus métodos, atributos e relacionamentos (herança, agregação, composição, etc.) precisa ser entregue. A ferramenta de código aberto utilizada para desenhar o diagrama é:
  - StarUML: `http://staruml.io/download`
- Como teste dessa primeira etapa, deve-se ao executar o projeto, desenhar os invasores na tela (usando *System.out.println(...)*) e em seguida deve-se colocar o canhão, mostrando o movimento dos invasores para encontrar o canhão;

Observações importantes:

- Apesar da *engine* não precisar estar completa nessa fase do projeto, ela já deve ser pensada e projetada considerando que outras funcionalidades serão acrescentadas;
- Dica para implementação do jogo: podemos ter quatro classes básicas, sendo elas canhão, naves, base e tiro. Mais detalhes podem ser vistos através do link: `http://www.cokeandcode.com/main/tutorials/space-invaders-101/`. Essa é uma forma de implementar, mas estratégias mais elegantes também podem ser usadas;

- Faça um programa bem feito, pois você ficará com ele até o final do semestre! Vou pedir que você estenda este programa;
- Um programa que faça tudo que pedi acima vale $8,0$. Se você me surpreender positivamente, seu programa vale 10.

3 Formato e Data de Entrega

O trabalho é **individual**, e a data de entrega é **05/11/2020** (até as 8:00hs do dia posterior). Trabalhos atrasados não serão aceitos, recebendo nota **zero**. Quaisquer programas similares terão nota zero independente de qual for o original e qual for a cópia. Será utilizada **ferramenta automatizada** para a detecção de **plágio**, com conferência manual de casos suspeitos. Os projetos devem ser entregues via Atividades do Tidia-ae (`https://ae4.tidia-ae.usp.br/portal`).

O formato da entrega deve ser um arquivo *.ZIP contendo:

- A pasta a ser zipada deve ter por nome o número USP do aluno. Trabalhos sem esse padrão de nome de arquivo não serão corrigidos e valerão ZERO;
- Uma imagem (*.JPG ou *.PNG) contendo o diagrama de classes;
- Um projeto NetBeans contendo todo o código desenvolvido.

4 Critério de Avaliação

- Funcionamento correto e estruturação em classes (separação de funcionalidades, relação de hierarquia, classe abstrata/interface, etc.): 50%
- Eficiência e elegância da abordagem sugerida da *engine* do sistema: 40%;
- JavaDoc e documentação interna das classes e métodos: 10%. Pesquise como criar e padronizar a documentação de software escrito em Java.