# Projeto Space Invaders – parte 2

SCC 604 – Programação Orientada a Objetos

Prof. Robson L. F. Cordeiro

7 de novembro de 2022

## 1 Descrição

Para a segunda etapa do projeto do jogo *Space Invaders*, fica a entrega do jogo completo, funcionando, com interface gráfica implementada em Java FX. As regras mínimas do jogo que serão verificadas são descritas a seguir.

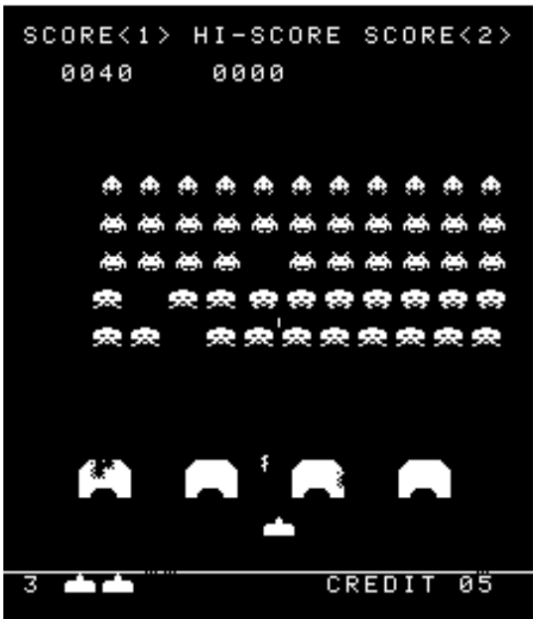


Figura 1: Canhão protetor.

Em *Space Invaders* você controla um canhão que defende a terra de invasores do espaço. Assim como é ilustrado na Figura 1, o canhão fica na parte inferior da tela, e ele pode se mover para a esquerda e para a direita. O canhão 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. Vide Figura 2.

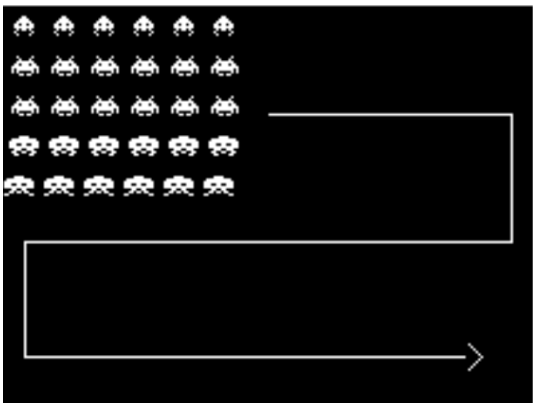

Figura 2: Movimento dos invasores.

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 na parte superior da tela que se move para o lado oposto 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.

Note-se que regras mais avançadas também podem ser empregadas, como por exemplo mudanças de fase, ou mesmo a inserção de outros personagens no jogo.

**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, pois 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; à 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 segunda etapa, o que precisa ser entregue envolve:

- Todo o código, bem estruturado e funcionando (modularizado de acordo com o que foi pedido na etapa anterior).
- Manual de usuário explicando o funcionamento completo do jogo, quais são os comandos principais, as regras do jogo, a descrição de cada fase (se existirem fases diferentes), etc. Tudo sucintamente;
- Um manual do sistema, explicando o funcionamento das partes, principalmente qual foi a estratégia empregada para o funcionamento dos invasores, se existem diferenças entre os invasores, como os desenhos

são criados, etc. É uma descrição pequena de cada parte, só para eu ter uma ideia de como as dificuldades de implementação do sistema foram superadas.

**Observações importantes:**

- A interface gráfica deve ser implementada em Java FX. Programas implementados com outra plataforma gráfica, como AWT ou Swing, serão desconsiderados, ficando com nota **zero**.
- Um programa que faça tudo o que pedi acima vale 8,0. Se você me surpreender positivamente, seu programa vale 10.
- Os sistemas não serão avaliados comparativamente, mas os que conseguirem ir além do que o pedido receberão com certeza maiores notas.
- Não somente o código será avaliado, mas também a qualidade final do software (interface gráfica, opções do jogo, etc). Nesse momento a avaliação também será feita como se eu fosse um usuário do sistema, não somente o professor que julga um código.

# 3 Formato e data de entrega

O trabalho é **individual**, e a data de entrega é **11/12/2022** (serão tolerados atrasos até às 8h do dia posterior). Trabalhos com maior atraso 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 **só o número USP** do aluno. Trabalhos sem esse padrão de nome de arquivo não serão corrigidos e valerão **zero**;
- Os manuais do sistema e do usuário em *.pdf ou *.doc.
- Um projeto NetBeans contendo todo o código desenvolvido.

# 4 Critérios de avaliação

- Funcionamento correto e estruturação em classes (separação de funcionalidades, relação de hierarquia, classe abstrata/interface, etc.): 40%
- Eficiência e elegância das estratégias sugeridas e qualidade geral do software (a partir do ponto de vista de um usuário): 30%
- Manuais do sistema e do usuário: 20%
- JavaDoc e documentação interna das classes e métodos: 10%. Pesquise como criar e padronizar a documentação de software escrito em Java.