Die Hausaufgaben müssen von den Studierenden einzeln bearbeitet und abgegeben werden. Für die Hausaufgabe sind die aktuellen Informationen vom Blog `https://seblog.cs.uni-kassel.de/ws2223/einfuhrung-in-die-informatik/` zu berücksichtigen.

Abgabefrist ist der 08.12.2022 - 23:59 Uhr

Vorbereitung

Zur Abgabe dieser Hausaufgabe muss zunächst ein neues Respository angelegt werden. Mit dem folgenden Link kannst du das Respository für Hausaufgabe 7 anlegen bzw. einsehen.

`https://classroom.github.com/a/fjo7ISqX`

Achte bei jeder Aufgabe auf das verlangte Dateiformat. Andere Formate werden nicht akzeptiert und folglich mit **0** Punkten gewertet. Handschriftliche Abgaben werden nicht akzeptiert.

Es sollen nur die bis zur jeweiligen Übung vermittelten Konzepte verwendet werden.

Nicht oder zu spät abgegebene (Teil-)Aufgaben werden mit 0 Punkten bewertet.

Code-Formatierung

Formatiere deinen Code vor dem Abgeben wie in Vorlesung und Übung gezeigt. Verstöße gegen Formatierungs- und Namenskonventionen führen zu Punktabzug.

Aufgabe 1 - Klassen (23P)

Mit Processing soll ein „Space Invaders"-Spiel programmiert werden.

Spiel-Prinzip

Auf dem Spielfeld gibt es mehrere Reihen von Gegnern. Alle Gegner einer Reihe bewegen sich gemeinsam nach links bzw. rechts, bis der äußerste Gegner am Rand des Fensters ankommt, dann ändern alle Gegner der Reihe gleichzeitig die Richtung.

Der User steuert die Kanone, die sich im unteren Bereich des Spielfelds befindet. Die Kanone kann nach links und rechts bis zum Rand des Fensters bewegt werden.

Die Gegner und die Kanone können schießen. Per Leertaste kann die Kanone einen Schuss nach oben Richtung Gegner schießen. Die Gegner schießen Richtung Spieler. Ob ein Gegner einen Schuss abfeuert, entscheidet der Zufall.

Der User hat zu Beginn jedes Levels drei Leben. Trifft ein Gegner-Geschoss die Kanone, verliert der User ein Leben. Trifft das Player-Geschoss einen Gegner, verschwindet dieser Gegner. Sobald ein Geschoss etwas getroffen hat, verschwindet es. Ein Gegner bzw. der Player können erst dann wieder schießen, wenn es kein eigenes Geschoss mehr auf dem Spielfeld gibt.

Mit jedem getroffenen Gegner erhöht sich die erreichte Punktzahl um einen Punkt. Sind alle Gegner abgeschossen, beginnt ein neuer Level mit mehr Gegnern. Wird die Kanone in einem Level dreimal von Gegnern getroffen, sind die Leben aufgebraucht und das Spiel ist vorbei. Die erreichten Punkte werden angezeigt und das Spiel kann erneut gespielt werden.

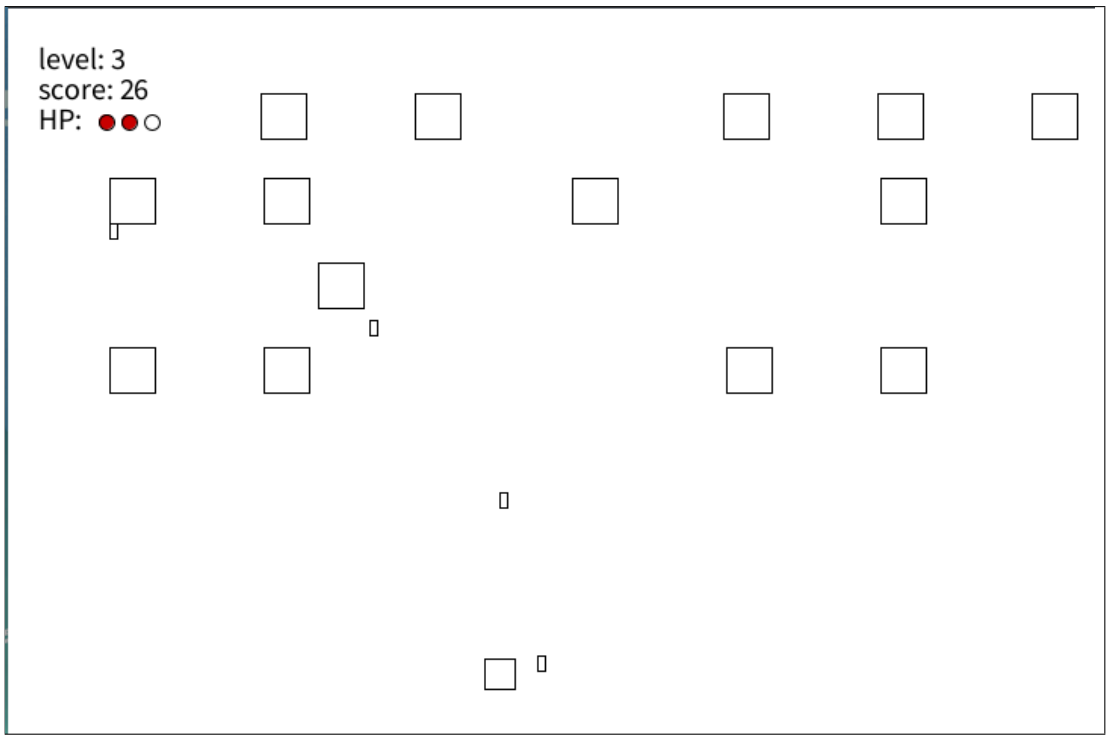

Abbildung 1: „Space Invaders"-Klon

Vorgegebener Code

Die Hauptdatei `spaceInvaders.pde` und die Klasse `Bullet.pde` findest du in deinem Repository. Um das Programm zu vervollständigen, müssen die Klassen `Enemy` und `Cannon` implementiert werden. Außerdem muss die Methode `initializeLevel()` in der Datei `spaceInvaders.pde` vervollständigt werden. Die entsprechende Stelle ist mit einem `TODO`-Kommentar markiert.

Mit der Klasse `Bullet` soll ein Geschoss repräsentiert werden können. In der Klasse befinden sich ebenfalls `TODO`-Kommentare, die es dir erlauben die Größe, Geschwindigkeit und Visualisierung anzupassen.

Spiel vervollständigen

Anhand des vorgegebenen Codes zeigt dir Processing bereits an, welche Klassen fehlen und welche Attribute und Methoden der Klassen noch fehlen. Es empfiehlt sich zunächst die Klassen grundlegend aufzubauen, sodass das Programm ausführbar ist. Währenddessen kannst du in den neu erstellten Methoden selbstständig `TODO`-Kommentare schreiben, um später dort den richtigen Code zu ergänzen.

Tipp: In den `update()`-Methoden von `Enemy` und `Cannon` muss unter anderem geprüft werden, ob das eigene Geschoss noch im Fenster zu sehen ist. Sobald es nicht mehr zu sehen ist, wird die Variable für das Geschoss auf `null` zurückgesetzt.

Hinweise

Es genügt, wenn die Gegner, die Kanone und die Geschosse einfache Rechtecke sind. Größen und Geschwindigkeiten dürfen verändert werden, solange alle Elemente erkennbar bleiben und das Spiel normal gespielt werden kann.

Wenn du möchtest, darfst du das Spiel auch etwas hübscher gestalten. Es können allerdings keine Bonuspunkte vergeben werden. Achte darauf, dass die oben beschriebenen Elemente erkennbar sind!

Die von Processing zur Verfügung gestellte Methode `rectMode()` darf **nicht** verwendet werden, damit die Kollisionserkennung des vorgegebenen Programmcodes weiterhin funktioniert.

Wenn du mit deiner Implementierung fertig bist, soll das Programm ausführbar und nach dem oben erklärten Spiel-Prinzip spielbar sein.

Die vorgegebenen Dateien dürfen nur an den Stellen verändert werden, die mit einem `TODO`-Kommentar markiert sind! Wenn du die Bearbeitung einer solchen Stelle abgeschlossen hast, lösche den `TODO`-Kommentar, sodass dein finales Programm keine `TODO`-Kommentare mehr enthält.

Abgabe

Gib als Lösung deine Dateien mit den folgenden Namen im GitHub-Repository zur aktuellen Hausaufgabe ab:

- spaceInvaders.pde
- Bullet.pde
- Cannon.pde
- Enemy.pde