INF-1400
Mandatory assignment 1 - Breakout clone

January 22, 2021

1 Introduction

In this assignment you will create a simple game, more specifically a clone of the classic game Breakout. The game will be written in Python, using the pygame library, with a goal of using the principles of object-oriented programming as best possible. That means you will implement the game using classes and methods.

2 Implementation

Your game must support the basic mechanics of the original game. In Breakout, the player controls a platform at the bottom of the screen, and attempts to hit a ball. On the top half of the screen there is a set of bricks. When the ball hits the bricks they disappear. The game is won by removing all the bricks on the screen. If the player misses the ball so it disappears below the screen, the game is lost.

It is important that the ball bounces off the platform with an angle dependant on where it hits the platform, so the player can control the ball. To implement this it may help to think of the platform as a semicircle, you may also draw the platform as a semicircle.

The look of the game is up to you, you can use the built-in functions in pygame for drawing rectangles and circles of different colors, or import images for a more pleasing aesthetic.

2.1 Requirements

Required elements in the implementation

1. Implement the game in accordance with object-oriented design, use objects and classes.
2. The platform should be controlled by the mouse or keyboard.
3. The ball should bounce in a different direction based on where on the platform it hits.
4. A brick disappears when the ball hits it.
5. The ball bounces off the wall and ceiling, angle in = angle out.
5. The game is won when all bricks are removed, the game is lost when the ball hits the bottom of the screen.
6. Well structured and commented code.

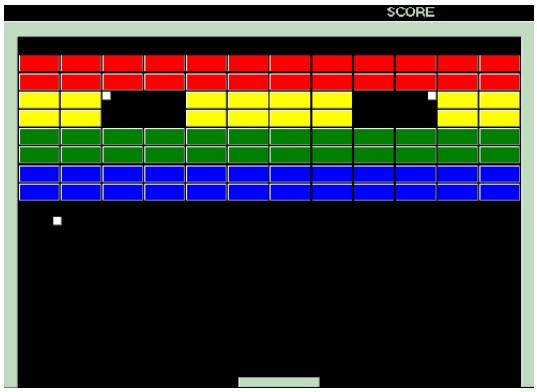

Figure 1: Breakout (1976)

2.2 Extras

If you feel like it, you can of course expand the game with extra features, examples include:

1. More levels.
2. A health system (so the game isn't immidiately over when the ball hits the bottom of the screen).
3. A high score list.
4. Sound.
5. Power-ups (extra life, shooting, larger platform, etc.).
6. Bricks can withstand more than one hit.

2.3 Precode

To help you get started, you have received some precode. The precode includes the pygame Vector2-class to easier handle position and speed, as well as methods to handle collision between two objects (circles and rectangles). The code also contains code to draw circles and rectangles, more help for the graphical things can be found in the pygame code from the lectures.

It is not required that you use this code, feel free to roll your own.

3 Report

Describe your implementation. Show that you have understood what you have done. Be precise. Explain what was difficult, and how you solved those problems. The report should contain figures that explain the design of your program. Include class diagrams (as shown in lectures) which describes relations between the different classes. Remember that the code should be well enough commented and written so that it, in addition to the report, makes it easy for another person to understand how your program is put together.

The report *must* be delivered in PDF format.

Templates will be handed out in Word, ODF and LaTeX formats.

4 Hand-in

A student with student-id *qwe123* puts the files in the following directory structure:

```
inf1400-qwe123-1/
       src/
        |--all the source files here
        |--README
report.pdf
```

The directory `inf1400-qwe123-1` is compressed to a zip or tar.gz archive and is handed in through Canvas before the deadline. The README should contain information on how to run the program. The report should be uploaded to Canvas separately from your zipped code-folder.

5 Peer-Review

After the deadline there will be a review where you and another student will look at each others code and solution in order to come up with constructive feedback on e.g. structure, flow, implementation, etc. At the next colloquium session you will exchange feedback, with the TA present as a moderator.

What to look for in a code review:
`https://google.github.io/eng-practices/review/reviewer/looking-for.html`

6 Alternative assignment

If games are not your thing, and you want to make something else, this is also possible, with some requirements:

- You must use object-oriented principles and design.
- The project must be sufficiently complex.
- You must present the project to the TAs.
- The design must be approved by the TAs.

You must get approval for your project before starting!

7 Cheating

Remember that cheating, or attempted cheating during mandatory assignments, is considered the same as cheating during an exam. Here are some guidelines.

- Copying code is not allowed.
- Copying the design off someone or off the internet is not allowed.
- Wrong use of/missing references are not allowed.
- Getting *help* from another student to solve a problem is allowed.
- Discussing design and code problems with other students is allowed.
- Getting the *solution* (code, design, or report elements) is not allowed.

8 Deadline

23:59, February 18th, 2020