CMPE 160 Assignment 1
Angry Bullet Game

Contacts:
Berk Gökberk (berk.gokberk@bogazici.edu.tr)
İrem Beyza Urhan (beyza.urhan@bogazici.edu.tr)

Due: March 21st, 6 a.m. in the morning

Before reading the assignment description, please first watch the gameplay video:
https://www.youtube.com/watch?v=yFUMJtHRrAs

In this assignment, a simple Angry Birds-like game will be implemented using Java. You should use the StdDraw graphics library. Aim of the game is to hit the targets by shooting a bullet. Bullet, as a projectile, should follow simple projectile motion physics rules. In projectile motion, (x_t, y_t) coordinates of a particle at time t are determined by the following formulas: $x_t = x_0 + v_0 \cos(\theta)t$ and $y_t = y_0 + v_0 \sin(\theta)t - \frac{1}{2}gt^2$ where (x_0, y_0) is the initial position coordinate pair, v_0 is the initial velocity, θ is the launch angle, and g is the gravity constant for earth. When you start the game, initial velocity will be 180 and the initial angle will be 45 degrees (See the black box in Figure 1 lower left corner). User will be able to change these values interactively by pressing up/down/left/right arrows. Your program should display the bullet trajectory coordinates at each time interval.

Gameplay

- Aim is to hit the targets by avoiding obstacles.
- Display the outcome of each shooting on the upper left corner. See Figure 2 to 6 for examples. Possible outcomes per shooting is: 1) hit the target, 2) hit the obstacle, 3) Touched the ground, 4) Max x interval is exceeded (bullet exits from the right side of the game window). It is allowed for a bullet to exceed the maximum y height of the game window and return to the game (See Figure 6).
- Press space bar to start shooting animation.
- Press Up/Down keys to adjust the bullet shooting angle before shooting starts.
- Press Right/Left keys to increase/decrease bullet speed before shooting starts.
- User should be able to shoot again by pressing the 'r' key after a shooting simulation is finished.

Game Environment

Figure 1 shows a sample game environment. It consists of the following elements:

- Targets: Orange-colored rectangles. These are the targets we aim to shoot with the bullet.
- Obstacles: Dark grey rectangles. If you hit an obstacle, shooting fails.
- Shooting platform: Black box in the lower left corner. Angle (a) and velocity (v) of the bullet is displayed inside the shooting platform box.
- Shooting Line: Illustrates the angle and velocity of the bullet. Line length and angle denote the velocity and the bullet shooting angle, respectively. Initially, angle is 45 degrees and velocity is 180. By pressing up/down arrows, you can rotate the shooting line. By pressing left/right arrow keys, you decrease/increase the line length.
- Bullet Trajectory: Black dots with lines to illustrate bullet trajectory. See Figures 2 to 6 for examples.

Figure 1. Game environment.

Game Parameters

Use the following game parameters in your code:

```
// Game Parameters
int width = 1600; //screen width
int height = 800; // screen height
double gravity = 9.80665; // gravity
double x0 = 120; // x and y coordinates of the bullet's starting position on the platform
double y0 = 120;
double bulletVelocity = 180; // initial velocity
double bulletAngle = 45.0; // initial angle

// Box coordinates for obstacles and targets
// Each row stores a box containing the following information:
// x and y coordinates of the lower left rectangle corner, width, and height
double[][] obstacleArray = {
  {1200, 0, 60, 220},
  {1000, 0, 60, 160},
  {600, 0, 60, 80},
  {600, 180, 60, 160},
  {220, 0, 120, 180}
};

double[][] targetArray = {
  {1160, 0, 30, 30},
  {730, 0, 30, 30},
  {150, 0, 20, 20},
  {1480, 0, 60, 60},
  {340, 80, 60, 30},
  {1500, 600, 60, 60}
};
```

Sample Gameplay Screens

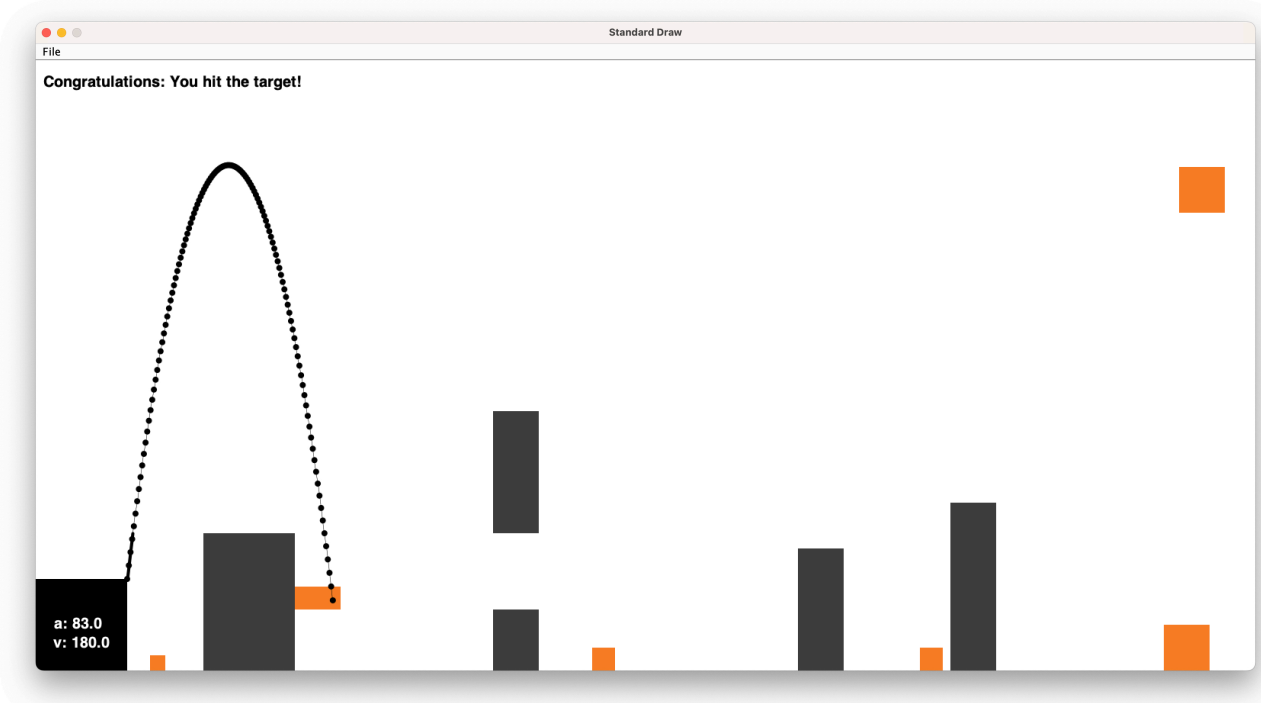

Figure 2. Case for hitting the target (orange box). Successful shot hits one of the orange targets.

Figure 3. Case for hitting an obstacle.

Figure 4. Case for ground contact. Shooting simulation should finish in this case.

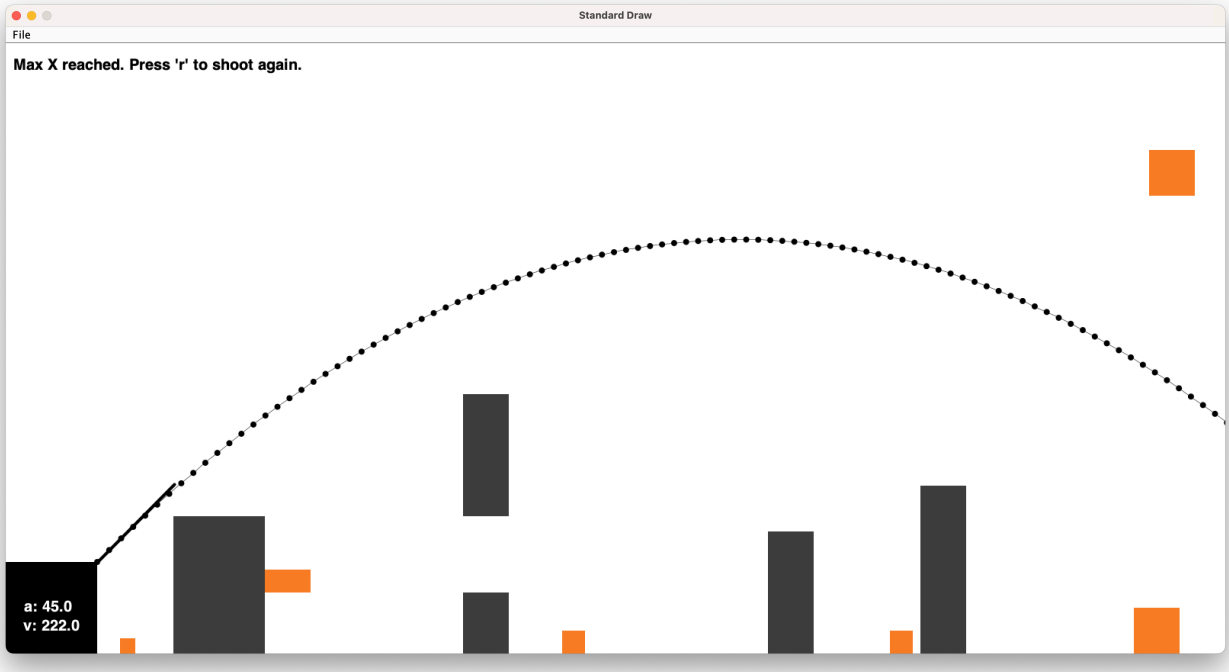

Figure 5. Case for Max X coordinate reach. Shooting simulation should finish in this case.

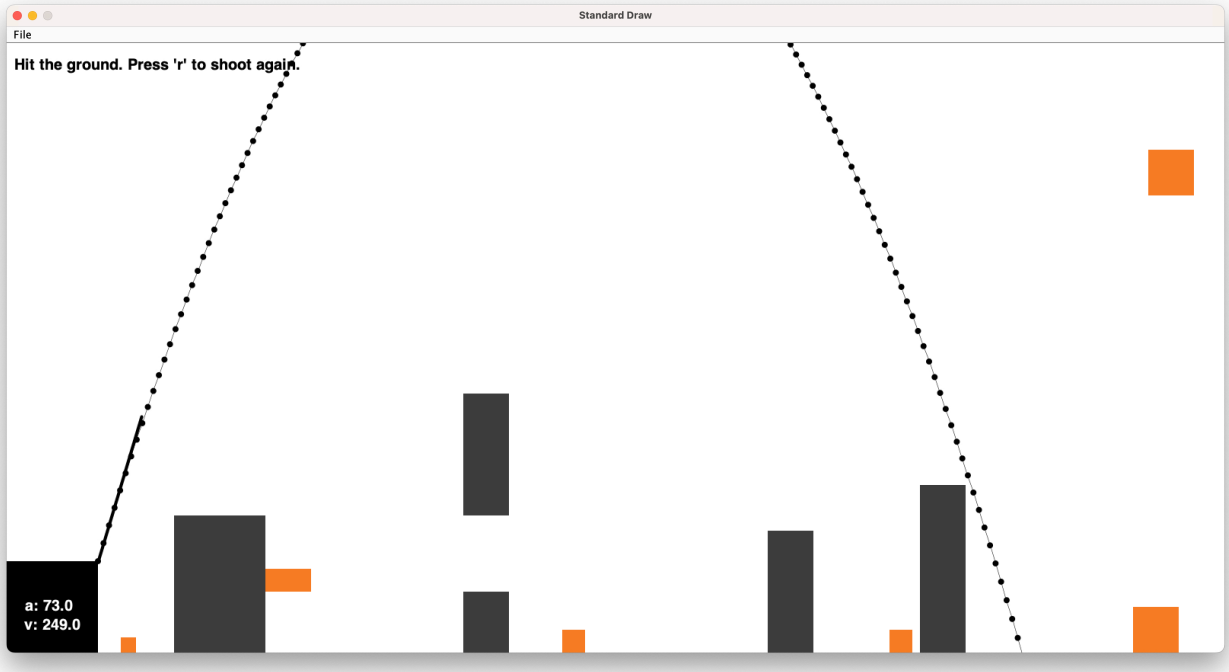

Figure 6. Bullet is allowed to travel through the upper part of the sky.

Submission Files

1. Java source code.
2. Report (in .pdf format) containing screen shots as seen in Figures 2 to 6 for the obstacle/target arrays provided to you above. Additionally, create your own game scene with difficult/creative target & obstacle positionings. Provide screenshots for few cases where you can be able to shoot a target.
3. A short gameplay video. Size of the video file should be small. If the maximum file size quota is reached on Moodle (2MB), you can provide a link for your video in your report. In this case, gameplay video link should be provided at the beginning of your report.

See the Submission Guide section (shown below) for more details on how to submit your files.

Notes

- Use `double[][] obstacleArray and double[][] targetArray` 2D array variable names in your code. Do not change these variable names. We may test your game with different target/obstacle configurations.
- You should use the StdDraw graphics library.
- Do not use object-oriented programming concepts in this assignment.
- In all assignments, do not use topics that we have not covered in the lectures.

Submission Guide

Submission Files

Submit a single compressed (.zip) file, named as NameSurname.zip, to Moodle. It should contain all source code files (under the \code directory), report (in PDF format, under the \report directory) and all other files if needed (under \misc directory). Name the main Java code as "NameSurname.java". Name your report as "NameSurname.pdf". Do not use Turkish characters in filenames, codes, and comments etc.

Late Submission Policy

Maximum late submission duration is two days. Late submissions will be graded on a scale of 50% of the original grade.

Mandatory Submission

Submission of assignments is mandatory. If you do not submit an assignment, you fail the course.

Plagiarism

Leads to grade F and YÖK regulations are applied.