У мене є код: import acm.graphics.GObject; import acm.graphics.GOval; import acm.graphics.GRect; import com.shpp.cs.a.graphics.WindowProgram; import java.awt.\*; import java.awt.event.MouseEvent; public class Breakout extends WindowProgram { //Width and height of application window in pixels public static final double APPLICATION\_WIDTH = 400; public static final double APPLICATION\_HEIGHT = 600; //Dimensions of the paddle private static final double PADDLE\_WIDTH = 60; private static final double PADDLE\_HEIGHT = 10; //Offset of the paddle up from the bottom private static final double PADDLE\_Y\_OFFSET = 30; //Number of bricks per row private static final double N\_BRICKS\_PER\_ROW = 10; //Number of rows of bricks private static final double N\_BRICK\_ROWS = 10; //Separation between bricks private static final double BRICK\_SEP = 4; //Height of a brick private static final double BRICK\_HEIGHT = 8; //Radius of the ball in pixels private static final double BALL\_RADIUS = 10; //Offset of the top brick row from the top private static final double BRICK\_Y\_OFFSET = BRICK\_SEP + BRICK\_HEIGHT; //Amount of games private static final double AMOUNT\_OF\_GAMES = 3; //Width right and left walls private static final double WALL\_WIDTH = 4; //The amount of time between frames (20fps) private static final double PAUSE\_TIME = 1000.0 / 20; //The initial horizontal velocity of the ball private static final double HORIZONTAL\_VELOCITY = 1.0; //Gravitational acceleration private static final double GRAVITY = 0.425; //Elasticity private static final double ELASTICITY = 0.75; public void run() { setSize((int) APPLICATION\_WIDTH, (int) APPLICATION\_HEIGHT); matrix(); rocket(); threeWall(); GObject o = drawOval(); addMouseListeners(); } private GObject selectedObject; private boolean isObjectSelected = false; @Override public void mousePressed(MouseEvent e) { selectedObject = getElementAt(e.getX(), e.getY()); if (selectedObject != null) { isObjectSelected = true; } } @Override public void mouseMoved(MouseEvent e) { if (selectedObject != null) { double newX = e.getX() - selectedObject.getWidth() / 2.0; double newY = selectedObject.getY(); // Используем текущее значение Y, чтобы ракетка не меняла свое положение по вертикали // Проверяем выход за границы экрана слева if (newX < WALL\_WIDTH) { newX = WALL\_WIDTH; } // Проверяем выход за границы экрана справа if (newX > getWidth() - selectedObject.getWidth() - WALL\_WIDTH) { newX = getWidth() - selectedObject.getWidth() - WALL\_WIDTH; } selectedObject.setLocation(newX, newY); } } private void brick(int x, double y, double iter) { GRect realBrick = new GRect(x, y, getWidth() / (N\_BRICKS\_PER\_ROW + BRICK\_SEP), BRICK\_HEIGHT); Color[] colors = {Color.RED, Color.ORANGE, Color.YELLOW, Color.GREEN, Color.CYAN, Color.PINK}; double[] thresholds = {1, 3, 5, 7, 9}; for (int i = 0; i < thresholds.length; i++) { if (iter <= thresholds[i]) { realBrick.setFilled(true); realBrick.setColor(colors[i]); break; } } add(realBrick); } //cycle private void matrix() { double startX = WALL\_WIDTH + BRICK\_SEP; double startY = getHeight() \* 0.1; for (int i = 0; i < N\_BRICK\_ROWS; i++) { for (int j = 0; j < N\_BRICKS\_PER\_ROW; j++) { if (j == 0) { startX = WALL\_WIDTH + BRICK\_SEP / 2; } brick((int) startX, startY, i); startX += getWidth() / N\_BRICKS\_PER\_ROW; } startY += BRICK\_Y\_OFFSET; startX = WALL\_WIDTH + (BRICK\_SEP / 2); } } private GOval drawOval() { GOval oval = new GOval ( (double) (getWidth() / 2) - BALL\_RADIUS, (double) getHeight() / 2, BALL\_RADIUS \* 2, BALL\_RADIUS \* 2 ); oval.setFilled(true); oval.setFillColor(Color.BLACK); oval.setColor(Color.BLACK); add(oval); return oval; } private void rocket() { GRect realRocket = new GRect ( (double) (getWidth() / 2) - (PADDLE\_WIDTH / 2), getHeight() - PADDLE\_Y\_OFFSET, PADDLE\_WIDTH, PADDLE\_HEIGHT); realRocket.setFilled(true); realRocket.setFillColor(Color.BLACK); realRocket.setColor(Color.BLACK); add(realRocket); } private void oneWall(double x, double y, double width, double height) { GRect wall = new GRect(x, y, width, height); wall.setFilled(true); wall.setFillColor(Color.BLACK); wall.setColor(Color.BLACK); add(wall); } private void threeWall() { //top oneWall(0, 0, getWidth(), BRICK\_SEP); //left wall oneWall(0, 0, BRICK\_SEP, getHeight()); //right wall oneWall(getWidth() - BRICK\_SEP, 0, BRICK\_SEP, getHeight()); } } Мені потрібно : 1) визначити кулю і описати її відскік від стін 2) визначити кулю і описати її відскік від стін 3) описати логіку його пересувань. 4) Програма повинна стежити за швидкістю м’яча, який складається з двох окремих компонентів, які ви, ймовірно, оголосили в змінних класу: private double vx, vy; Компоненти швидкості відбивають зміни в положенні, яке відбувається на кожному кроці. 5) Спочатку куля має бути спрямована вниз, і ви можете розпочати з швидкості +3.0 для vy (пам’ятайте, що значення y в Java збільшується при русі вниз, початок координат — в лівому верхньому кутку). 6) Гра буде нудна, якщо кожна куля рухатиметься по одній і тій же траєкторії, тому вам слід вибрати компонент vx випадковим чином. 7) У продовженні нашої розмови про генерацію випадкових чисел минулого тижня, ви повинні ініціалізувати vx наступним чином: 8) RandomGenerator rgen = RandomGenerator.getInstance(); vx = rgen.nextDouble(1.0, 3.0); if (rgen.nextBoolean(0.5)) vx = -vx; 9) Цей код привласнює випадкове значення типу double в діапазоні від 1.0 до 3.0 і після чого, з ймовірністю 50%, змінює знак отриманого числа на мінус. 10) Цей спосіб краще для Breakout, ніж просто виклик функціїRgen.nextDouble(1.0, 3.0), яка може генерувати рух кулі більш-менш прямо вниз. Це сильно спростить життя гравця і зробить його нудним. 11) Після того, як ви зробили це, ваша наступна задача: навчити кулю відскакувати від всіх об’єктів у світі, повністю ігноруючи ракетку (зараз куля повинна пройти прямо через неї). 12) Щоб зробити це, вам треба перевірити, чи пішли координати м’яча за кордон, зважаючи, що куля має ненульовий розмір. 13) Так, щоб побачити, що м’яч відскочив від правої стіни, необхідно перевірити — чи не стала координата правого краю кулі більше ширини вікна; аналогічно і 3 інших напрями. 14) Відскік від нижньої стінки не треба, оскільки це означає фейл гравця, але спершу можете зробити і його.

Обчислення, які відбуваються далі — украй прості. Якщо куля досягає верху або низу, все, що вам треба — це поміняти знак vy. Аналогічно, стикання з бічними стінами змінює знак vx. Замисліться, наскільки це просто і красиво. Коли ми робили анімацію стрибаючої кулі, нам треба було хвилюватися про той випадок, коли куля застрявала в стінах або підлозі, зараз же нам не треба про це турбуватися. Якщо м’яч входив в підлогу — він міг не мати достатньо енергії, щоб відстрибнути, тому що м’яч сповільнюється при зіткненні (там був код уповільнення, згасання його вектора руху). У Breakout зіткнення не викликають у кулі втрати енергії, тому немає потреби в таких перевірках.