*/\*  
 \* Copyright (c) 2017. Phasmid Software  
 \*/*package edu.neu.coe.info6205.randomwalk;  
  
import java.util.Random;  
  
public class RandomWalk {  
  
 private int x = 0;  
 private int y = 0;  
  
 private final Random random = new Random();  
  
 */\*\*  
 \* Private method to move the current position, that's to say the drunkard moves  
 \*  
 \* @param dx the distance he moves in the x direction  
 \* @param dy the distance he moves in the y direction  
 \*/* private void move(int dx, int dy) {  
 *// FIXME do move by replacing the following code* x += dx;  
 y += dy;  
 *// END* }  
  
 */\*\*  
 \* Perform a random walk of m steps  
 \*  
 \* @param m the number of steps the drunkard takes  
 \*/* private void randomWalk(int m) {  
 *// FIXME* for (int i = m; i > 0; i--) {  
 randomMove();  
 }  
 *// END* }  
  
 */\*\*  
 \* Private method to generate a random move according to the rules of the situation.  
 \* That's to say, moves can be (+-1, 0) or (0, +-1).  
 \*/  
  
 // ns == true --> move on X; ns == false --> move on Y.  
 // step(1 or -1) --> forward/backward  
 // result --> (1, 0) / (-1, 0) / (0, 1) / (0, -1)* private void randomMove() {  
 boolean ns = random.nextBoolean();  
 int step = random.nextBoolean() ? 1 : -1;  
 move(ns ? step : 0, ns ? 0 : step);  
 }  
  
 */\*\*  
 \* Method to compute the distance from the origin (the lamp-post where the drunkard starts) to his current position.  
 \*  
 \* @return the (Euclidean) distance from the origin to the current position.  
 \*/* public double distance() {  
 *// FIXME by replacing the following code* double eDistance = Math.*sqrt*((x\*x) + (y\*y));  
 return eDistance;  
 *// END* }  
  
 */\*\*  
 \* Perform multiple random walk experiments, returning the mean distance.  
 \*  
 \* @param m the number of steps for each experiment  
 \* @param n the number of experiments to run  
 \* @return the mean distance  
 \*/* public static double randomWalkMulti(int m, int n) {  
 double totalDistance = 0;  
 for (int i = 0; i < n; i++) {  
 RandomWalk walk = new RandomWalk();  
 walk.randomWalk(m);  
 totalDistance = totalDistance + walk.distance();  
 }  
 return totalDistance / n;  
 }  
  
 public static void main(String[] args) {  
  
 Random r = new Random();  
  
 for (int i = 0; i < 10; i++) {  
  
 int m = r.nextInt(91) + 10;  
 int n = 30;  
 double meanDistance = *randomWalkMulti*(m, n);  
 System.*out*.println(m + " steps: " + meanDistance + " over " + n + " experiments");  
  
 }  
  
 }  
  
}