1 /\*\*  
 2 \* The Catapult object takes a series of data (speeds and angles) and determines if they can hit  
 3 \* a target distance (between min and max).  
 4 \*  
 5 \* The constructor accepts an array of speeds, and array of angles, a minimum, and a maximum trajectory.  
 6 \* The method calculate() uses the trajectory of a projectile formula to calculate each speed and  
 7 \* angle combination, putting it all into a multidimensional array.  
 8 \* The method getBestTraj() determines if any of the possible combinations are within the min and max  
 9 \* range. If so, it adds the speed, angle, and distance into an arraylist.  
 10 \* The method printTable() takes all of the information from the constructor and previously mentioned  
 11 \* methods and prints them out in a readable table.  
 12 \*   
 13 \* @author Stephanie Gremillion  
 14 \* @version 17.0.2  
 15 \* @since 2022/06/26  
 16 \*/  
 17   
 18 import java.lang.Math;  
 19 import java.util.ArrayList;  
 20 import java.util.\*;  
 21   
 22 public class Catapult {  
 23   
 24 // set variables  
 25 private int[] speeds, angles; // arrays that store the accepted speeds and angles  
 26 private double[][] calculations; // 2D array for the calculated combinations of speed and angles  
 27 private int min, max; // the min and max distances  
 28 private ArrayList<Integer> bestTraj = new ArrayList<Integer>(); // list of best trajectories after calculations  
 29 private boolean noBestTraj = false; // determines if there is a best trajectory or not  
 30   
 31   
 32   
 33 // constructor  
 34 public Catapult(int[] speedList, int[] anglesList, int mn, int mx) {  
 35 min = mn;  
 36 max = mx;  
 37   
 38 speeds = new int[speedList.length];  
 39 for(int i = 0; i < speeds.length; i++) { // filling speeds array  
 40 speeds[i] = speedList[i];  
 41 }  
 42   
 43 angles = new int[anglesList.length];  
 44 for(int i = 0; i < angles.length; i++) { // filling angles array  
 45 angles[i] = anglesList[i];  
 46 }  
 47 }  
 48   
 49   
 50   
 51 // method to calculate each speed and angle combination  
 52 public void calculate() {  
 53 calculations = new double[speeds.length][angles.length];  
 54 for(int i = 0; i < speeds.length; i++) {  
 55 for(int ii = 0; ii < angles.length; ii++) {  
 56 calculations[i][ii] = (Math.pow(speeds[i], 2) \* Math.sin(Math.toRadians(angles[ii]) \* 2.0) / Math.pow(9.8, 2));  
 57 }  
 58 }  
 59 }  
 60   
 61   
 62   
 63 // method to determine if there are any combinations that fit the distination  
 64 public void getBestTraj() {  
 65 // set variables  
 66 int x = 0; // for counting if there are any data within range  
 67 for(int i = 0; i < speeds.length; i++) {  
 68 for(int ii = 0; ii < angles.length; ii++) {  
 69 if((calculations[i][ii] > min) && (calculations[i][ii] < max)) {  
 70 bestTraj.add(speeds[i]); // adding speeds coordinate to correspond with correct trajectory  
 71 bestTraj.add(angles[ii]); // adding angles coordinate to correspond with correct trajectory  
 72 bestTraj.add((int)calculations[i][ii]); // adding landing point for correct trajectory  
 73 x++;  
 74 }  
 75 }  
 76 }  
 77 if(x == 0) { // if there are no trajectories stored, will output such at printTable  
 78 noBestTraj = true;  
 79 }  
 80 }  
 81   
 82   
 83   
 84 // method to print the projectile table  
 85 public void printTable() {  
 86 //set variables  
 87 int i, ii;  
 88   
 89 // printing header and degrees  
 90 System.out.println("\t\tProjectile Table");  
 91 System.out.println("---------------------------");  
 92 System.out.printf("%-10s", "Speed");  
 93 for(i = 0; i < angles.length; i++) {  
 94 System.out.printf("%10s", (String.valueOf(angles[i]) + " deg"));  
 95 }  
 96 System.out.println(); // next line  
 97   
 98 // printing subsequent lines of table  
 99 for(i = 0; i < speeds.length; i++) {  
100 System.out.printf("%10d", speeds[i]);  
101 for(ii = 0; ii < angles.length; ii++) {  
102 System.out.printf("%10.3f", calculations[i][ii]);  
103 }  
104 System.out.println(); // next line  
105 }  
106   
107 // printing footer  
108 System.out.println("---------------------------");  
109 if(noBestTraj == false) { // if there are viable combinations...  
110 int y = 0; // for counting how many bestTraj coordinates have been printed  
111 System.out.println("Best Trajectory Values:");  
112 for(i = 0; i < bestTraj.size(); i+=3) {  
113 System.out.print("(Speed: " + bestTraj.get(i));  
114 System.out.print(", " + bestTraj.get(i+1) + " deg) = ");  
115 System.out.print(bestTraj.get(i+2) + "m ");  
116 y++;  
117 if((y%5 == 0) && (i+3 < bestTraj.size())) { // making a line break after each 5th coordinates printed  
118 System.out.println(); // next line  
119 }  
120 }  
121 System.out.println(); // next line  
122 }  
123 else { // if there are no viable combinations...  
124 System.out.println("No viable launch available for these combinations.");  
125 }  
126 System.out.println("---------------------------");  
127 System.out.println(); // empty line  
128 }  
129 }

1 /\*\*  
 2 \* The class test the Catapult object.  
 3 \*  
 4 \* The main() method reads the file with the data, creates a Catapult object, calls calculate() and getBestTraj()  
 5 \* to manipulate the object, then calls printTable() to print the data.  
 6 \*   
 7 \* @author Stephanie Gremillion  
 8 \* @version 17.0.2  
 9 \* @since 2022/06/26  
10 \*/  
11   
12 import java.util.Scanner;  
13 import java.io.\*;  
14 import java.util.ArrayList;  
15   
16 public class CatapultTester {  
17   
18 public static void main(String[] args) throws IOException {  
19 // set variables  
20 int sets = 0;  
21 int temp = 0;  
22 int min = 0;  
23 int max = 0;  
24 int i, ii;  
25 int[] speedList, anglesList;  
26   
27 // open file  
28 File file = new File("project\_2\_data.txt");  
29 if (!file.exists()) {   
30 System.out.println("File does not exist.");// error if cannot find file  
31 System.exit(0);  
32 }  
33   
34 // setup file scanner  
35 Scanner input = new Scanner(file);  
36   
37 sets = input.nextInt(); // setting the amount of set to read and the number of times for the loop to run  
38   
39 // loop for each set  
40 for(i = 0; i < sets; i++) {  
41 temp = input.nextInt(); // getting the quantity of speeds  
42 speedList = new int[temp]; // set size of speedList  
43   
44 // loop for each speed  
45 for(ii = 0; ii < temp; ii++) {  
46 speedList[ii] = input.nextInt();  
47 }  
48   
49 temp = input.nextInt(); // getting the quantity of angles  
50 anglesList = new int[temp]; // set size of anglesList  
51 for(ii = 0; ii < temp; ii++) {  
52 anglesList[ii] = input.nextInt();  
53 }  
54   
55 min = input.nextInt(); // sets min trajectory  
56 max = input.nextInt(); // sets max trajectory  
57   
58 Catapult projTable = new Catapult(speedList, anglesList, min, max);  
59 projTable.calculate(); // calculates the data for the table  
60 projTable.getBestTraj(); // finds the best trajectories   
61 projTable.printTable(); // prints all of the information  
62 }   
63   
64 input.close(); // closes file  
65 }  
66 }