









package org.mcustiel.codility;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.List;

public class Battleship {

public Battleship() {

}

public String solution(int N, String S, String T) {

List<Ship> ships = parseShips(S, N \* N);

List<Point> hits = parseHits(T, N \* N);

int touched = 0, sunken = 0;

Iterator<Ship> shipIt = ships.iterator();

while (shipIt.hasNext()) {

Ship current = shipIt.next();

int touching = current.getHits(hits);

if (touching > 0) {

if (touching == current.getSize()) {

sunken++;

} else {

touched++;

}

}

}

return "" + sunken + "," + touched;

}

public List<Point> parseHits(String hits, int maxHits) {

List<Point> hitsList = new ArrayList<Point>(maxHits);

String[] coords = hits.split(" ");

for (String coord : coords) {

hitsList.add(new Point(coord));

}

return hitsList;

}

public List<Ship> parseShips(String ships, int maxShips) {

List<Ship> shipsList = new ArrayList<Ship>(maxShips);

String[] shipsCoords = ships.split(",");

for (String shipCoord : shipsCoords) {

String[] coords = shipCoord.split(" ");

shipsList.add(new Ship(new Point(coords[0]), new Point(coords[1])));

}

return shipsList;

}

static public class Point {

int x;

int y;

public Point(String coord) {

x = (coord.toUpperCase().charAt(1)) - ('A');

y = (coord.charAt(0)) - ('1');

}

public int getX() {

return x;

}

public int getY() {

return y;

}

public boolean greaterOrEqual(Point other) {

return x >= other.x && y >= other.y;

}

@Override

public String toString() {

return "(" + x + ", " + y + ")";

}

}

static public class Ship {

private Point topLeft;

private Point bottomRight;

public Ship(Point topLeft, Point bottomRight) {

this.topLeft = topLeft;

this.bottomRight = bottomRight;

}

public Point getTopLeft() {

return topLeft;

}

public Point getBottomRight() {

return bottomRight;

}

public int getSize() {

return (Math.abs(topLeft.getX() - bottomRight.getX()) + 1)

\* (Math.abs(topLeft.getY() - bottomRight.getY()) + 1);

}

@Override

public String toString() {

return "(" + topLeft + ", " + bottomRight + ")";

}

public int getHits(List<Point> shots) {

Iterator<Point> shotIt = shots.iterator();

int hits = 0;

while (shotIt.hasNext()) {

Point shot = shotIt.next();

if (shot.greaterOrEqual(topLeft) && bottomRight.greaterOrEqual(shot)) {

hits++;

}

}

return hits;

}

}

}