Assignment 1 (Random Walk)

1.Conclusion:

The relationship between *d* and *m*.

This program simulates a random walk of 10 steps and calculates the distance from the starting pont using the euclidean distance formula. Since, the person takes 1 meter steps in any of the four cardinal directions, the distance (d) after m steps can be computed using the formula: d = sqrt(m) (m). Because the distance depends on how randomly the steps are done, it may not be exactly the same as this estimation.

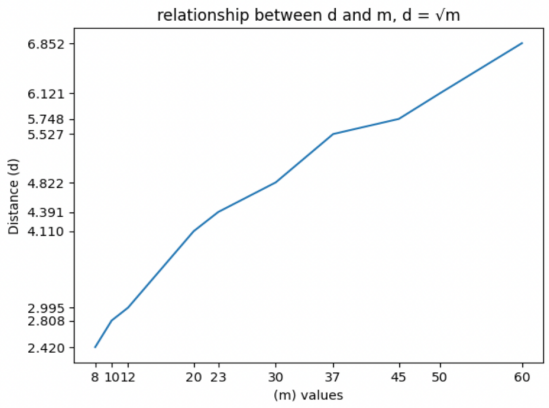
Therefore, **I conclude the relationship of d and m => d = sqrt(m)** implies that, as **the number of steps taken rises**, the **man's distance from the lamp post will also increase** (m).

2. **Evidence:**

The relationship between *d* and *m*, *d* = √m

m = [8, 10, 12, 20, 23, 30, 37, 45, 50, 60]
d = [2.4200034841367373, 2.8076524637580618, 2.9952519471271124,4.110348426613134, 4.390885214146857, 4.82204283856797, 5.5273669206673965, 5.747701721854294, 6.120839125315964, 6.85211527437173]

```
[9]: import matplotlib.pyplot as plt

     # Generate some data
     x = [8, 10, 12, 20, 23, 30, 37, 45, 50, 60]
     y = [2.4200034841367373, 2.8076524637580618, 2.9952519471271124, 4.110348426613134, 4.390885214146857, 4.82204283856797, 5.5273669206673965, 5

     # Create the plot
     plt.plot(x, y)

     plt.xticks(x)
     plt.yticks(y)

     # Add labels and title
     plt.xlabel('(m) values')
     plt.ylabel('Distance (d)')
     plt.title('relationship between d and m, d = √m')

     # Show the plot
     plt.show()
```

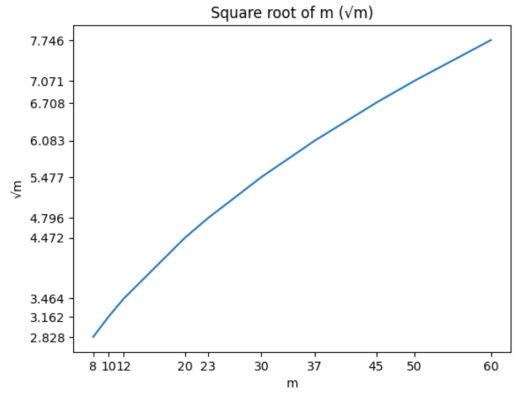


Assignment 1 (Random Walk)

Square root of m

x = [8, 10, 12, 20, 23, 30, 37, 45, 50, 60]
y = [2.8284271247461903, 3.1622776601683795, 3.4641016151377544, 4.47213595499958, 4.795831523312719, 5.477225575051661, 6.082762530298219, 6.708203932499369, 7.0710678118654755, 7.745966692414834]

```
[11]: import matplotlib.pyplot as plt

# Generate some data
x = [8, 10, 12, 20, 23, 30, 37, 45, 50, 60]
y = [2.8284271247461903, 3.1622776601683795, 3.4641016151377544, 4.47213595499958, 4.795831523312719, 5.477225575051661, 6.082762530298219, 6.

# Create the plot
plt.plot(x, y)

plt.xticks(x)
plt.yticks(y)

# Add labels and title
plt.xlabel('m')
plt.ylabel('√m')
plt.title('Square root of m (√m)')

# Show the plot
plt.show()
```



Inferring from a comparison of two graphs that d and m are related by the formula d = sqrt (m)

3. Code:

```
/*
 * Copyright (c) 2017. Phasmid Software
 */

package edu.neu.coe.info6205.randomwalk;


import java.util.ArrayList;
import java.util.Random;
import java.lang.Math;
import java.util.*;



public class RandomWalk {
```

Assignment 1 (Random Walk)

```
    private int x = 0;
    private int y = 0;

    private final Random random = new Random();


    RandomWalk walk ;

    /**
     * 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
     */

    /*
      if x=1 => person moves in east direction
      if x=-1 => person moves in west direction
      if y=1 => person moves in north direction
      if y=-1 => person moves in south direction
     */
    private void move(int dx, int dy) {
        // FIXME do move by replacing the following code
        //System.out.println(" move ");
         x = x + dx;
         y = y + dy;
    }

    /**
     * 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 = 0; i < m; 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).
     */
    private void randomMove() {
        boolean ns = random.nextBoolean();
        int step = random.nextBoolean() ? 1 : -1;
      //  System.out.println(ns+"  "+step);
        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.
```

Assignment 1 (Random Walk)

```
     *
     * @return the (Euclidean) distance from the origin to the current
position.
     */

    //provides the distance of the person from the lamp-post
    public double distance() {
        //System.out.println(x + " " +y);
        //FIXME by replacing the following code
         Double dist = Math.sqrt(y*y + x*x);
         //System.out.println(dist);
         return dist;
        // 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();
            System.out.println("Experiment Number: "+n+" Steps Taken: "+m+"
Distance Covered: "+walk.distance());
        }
        //System.out.println(dist_arr);
//        System.out.println(m+ "Experiment and mean is "+totalDistance / n);
        return totalDistance / n;
    }

    public static void main(String[] args) {
        //System.out.println(args[0]+" vcd "+args[1]);
//        if (args.length == 0)
//            throw new RuntimeException("Syntax: RandomWalk steps
[experiments]");
        int[] m = new int[]{ 8, 10, 12, 20, 23, 30, 37, 45, 50, 60 };
        int n = 20;
        List<Double> dist_arr = new ArrayList<Double>();
        List<Double> sq_arr = new ArrayList<Double>();
       // Arrays.sort(m);
        if (args.length > 1) n = Integer.parseInt(args[1]);
        for(int i=0;i<m.length;i++) {
            double meanDistance = randomWalkMulti(m[i], n);
            dist_arr.add(meanDistance);
            System.out.println(m + " steps: " + meanDistance + " over " + n +
" experiments");
        }
        for(int i=0;i<m.length;i++) {

            sq_arr.add(Math.sqrt(m[i]));
```

Assignment 1 (Random Walk)

```
            //System.out.println(m + " steps: " + meanDistance + " over " + n
+ " experiments");
        }


        System.out.println(Arrays.toString(m));
        System.out.println(dist_arr);
        System.out.println(sq_arr);
       // System.out.println(m + " steps: " + meanDistance + " over " + n + "
experiments");
    }

}
```

4. Screenshot of unit test cases: