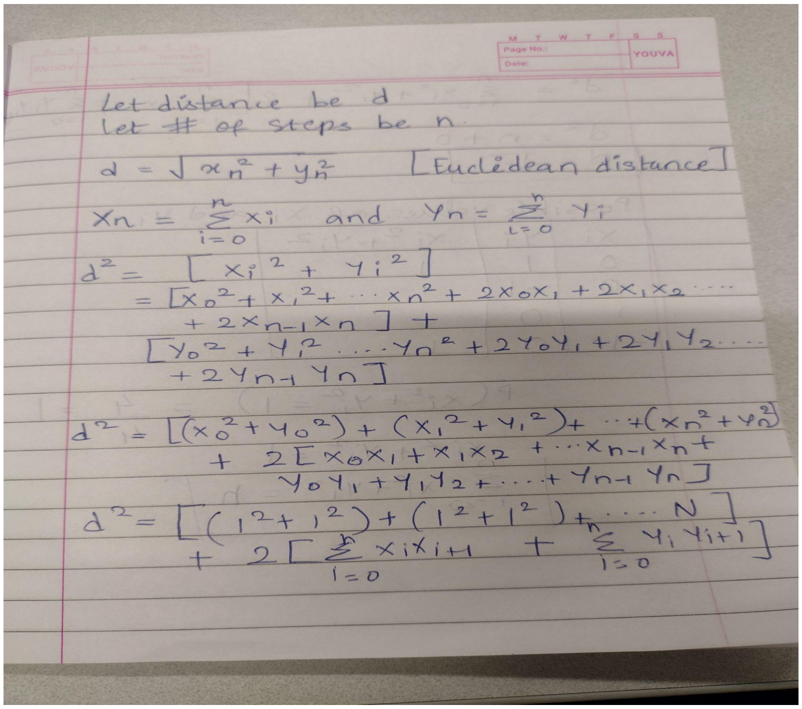
Riddhi Bhatti | NUID: 001502713

Program Structures & Algorithms
Fall 2021
Assignment 1

1. **Tasks Performed in the Assignment :**
   a. **Implemented the code for mentioned methods**
   b. **Ran the experiment for 15 different values of steps (n) and ran each step more than 20 times to increase confidence**
   c. **Plotted the values of d and n on line chart using google sheets**
   d. **Deduce the relationship between the mean distance (d) and number of steps (n)**
   e. **Ensured that all the test cases ran successfully**

2. Relationship conclusion: d = √n



Page No.: Date: YOUVA

Let distance be d
Let # of steps be n

$d = \sqrt{x_n^2 + y_n^2}$ [Euclidean distance]

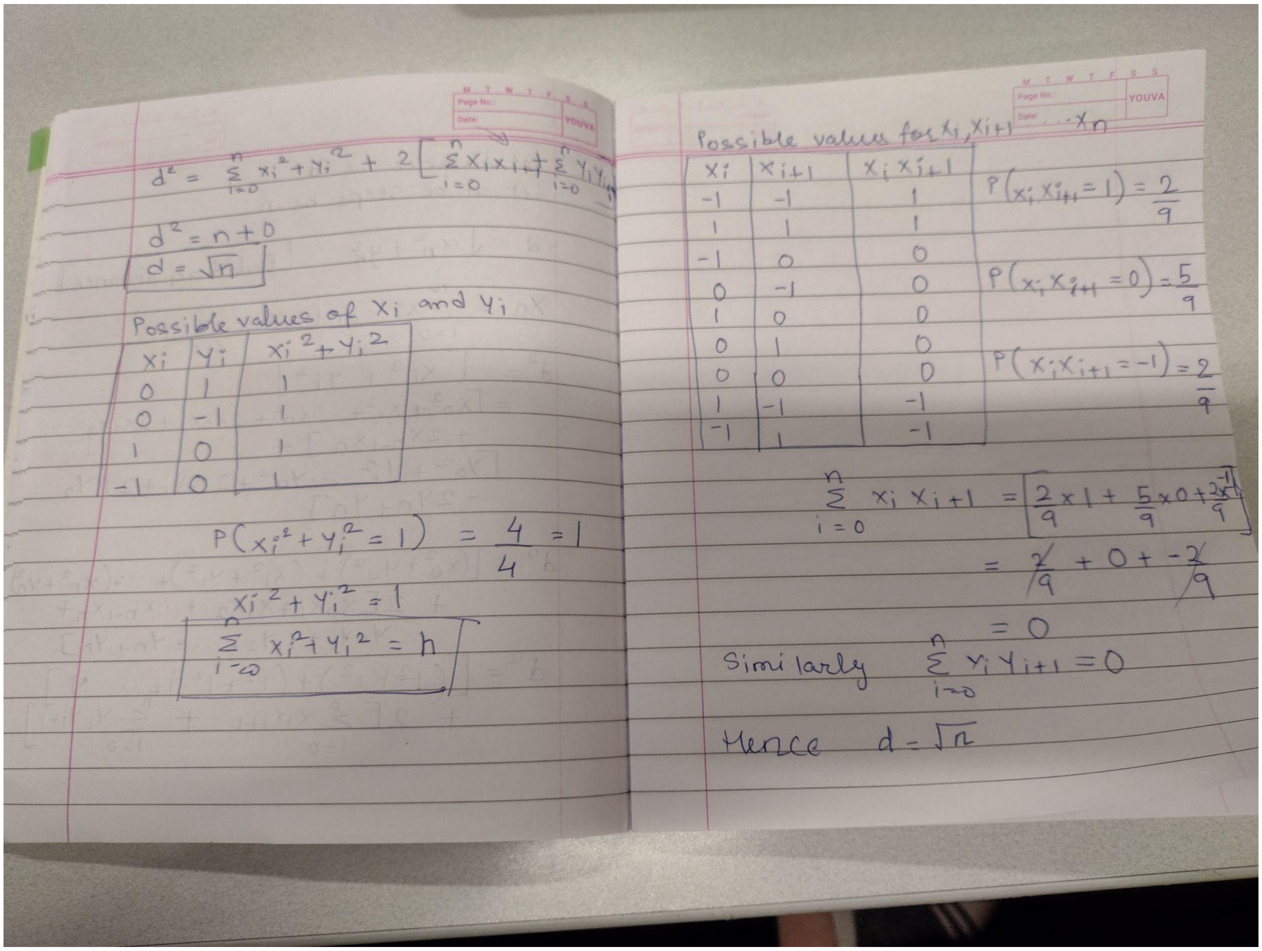
$X_n = \sum_{i=0}^{n} x_i$ and $Y_n = \sum_{i=0}^{n} Y_i$

$d^2 = [x_i^2 + y_i^2]$

$= [x_0^2 + x_1^2 + \cdots x_n^2 + 2x_0x_1 + 2x_1x_2 \cdots$
$+ 2x_{n-1}x_n] +$
$[Y_0^2 + Y_1^2 \cdots Y_n^2 + 2Y_0Y_1 + 2Y_1Y_2 \cdots$
$+ 2Y_{n-1}Y_n]$

$d^2 = [(x_0^2 + y_0^2) + (x_1^2 + y_1^2) + \cdots + (x_n^2 + y_n^2)]$
$+ 2[x_0x_1 + x_1x_2 + \cdots x_{n-1}x_n +$
$y_0y_1 + y_1y_2 + \cdots + y_{n-1}y_n]$

$d^2 = [(1^2 + 1^2) + (1^2 + 1^2) + \cdots N]$
$+ 2[\sum_{i=0}^{n} x_ix_{i+1} + \sum_{i=0}^{n} y_iy_{i+1}]$



$$d^2 = \sum_{i=0}^{n} x_i^2 + y_i^2 + 2\left[\sum_{i=0}^{n} x_i x_{i+1} + \sum_{i=0}^{n} y_i y_{i+1}\right]$$

$$d^2 = n + 0$$

$$\boxed{d = \sqrt{n}}$$

Possible values of x_i and y_i

x_i	y_i	$x_i^2 + y_i^2$
0	1	1
0	-1	1
1	0	1
-1	0	1

$$P(x_i^2 + y_i^2 = 1) = \frac{4}{4} = 1$$

$$x_i^2 + y_i^2 = 1$$

$$\boxed{\sum_{i=0}^{n} x_i^2 + y_i^2 = n}$$

Possible values for $x_i, x_{i+1} \ldots x_n$

x_i	x_{i+1}	$x_i x_{i+1}$
-1	-1	1
1	1	1
-1	0	0
0	-1	0
1	0	0
0	1	0
0	0	0
1	-1	-1
-1	1	-1

$$P(x_i x_{i+1} = 1) = \frac{2}{9}$$

$$P(x_i x_{i+1} = 0) = \frac{5}{9}$$

$$P(x_i x_{i+1} = -1) = \frac{2}{9}$$

$$\sum_{i=0}^{n} x_i x_{i+1} = \left[\frac{2}{9} \times 1 + \frac{5}{9} \times 0 + \frac{2}{9} \times \overset{-1}{\cancel{1}}\right]$$

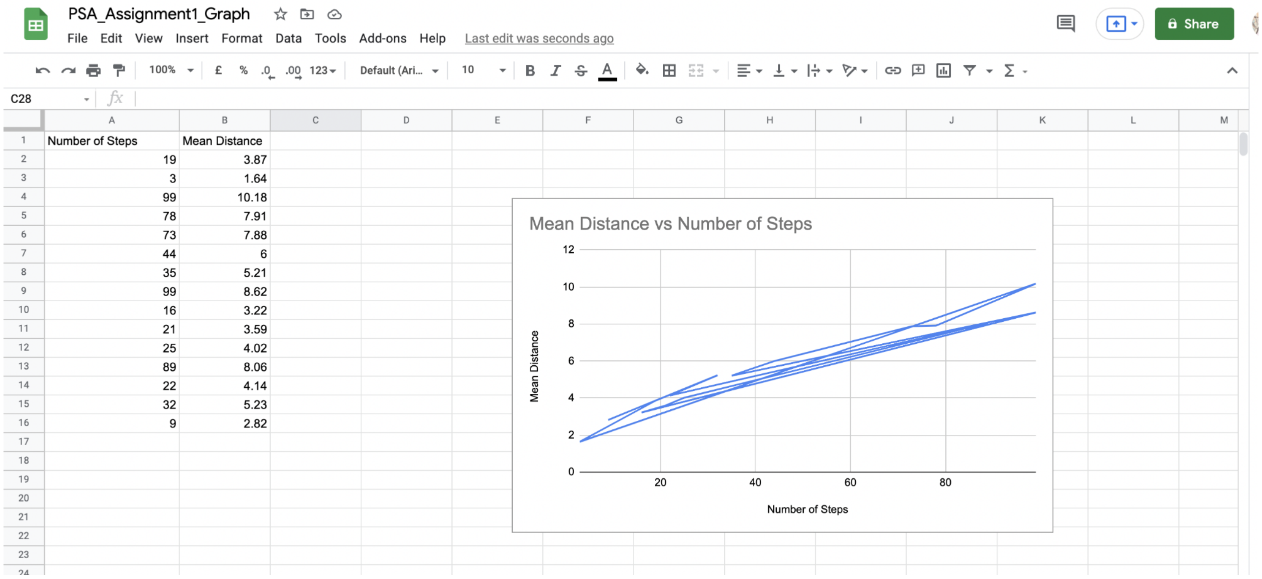
$$= \frac{\cancel{2}}{\cancel{9}} + 0 + -\frac{\cancel{2}}{\cancel{9}}$$

$$= 0$$

Similarly $\sum_{i=0}^{n} y_i y_{i+1} = 0$

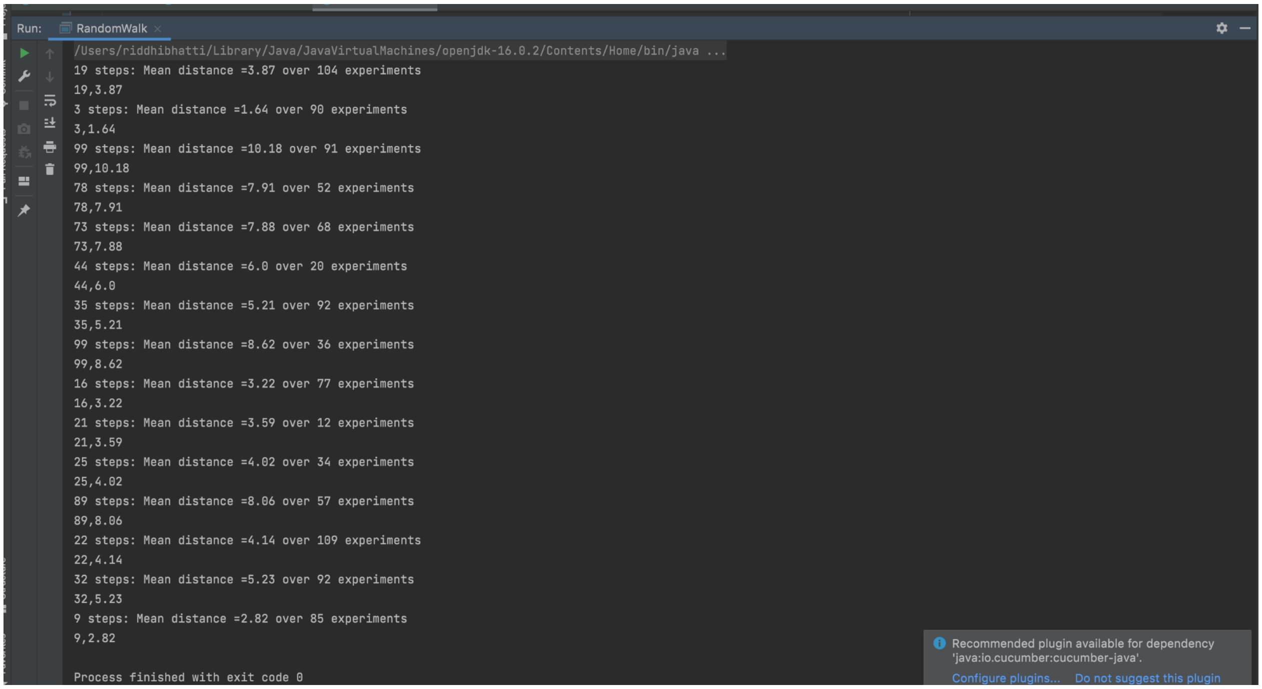
Hence $d = \sqrt{n}$

3. Evidence to support conclusion



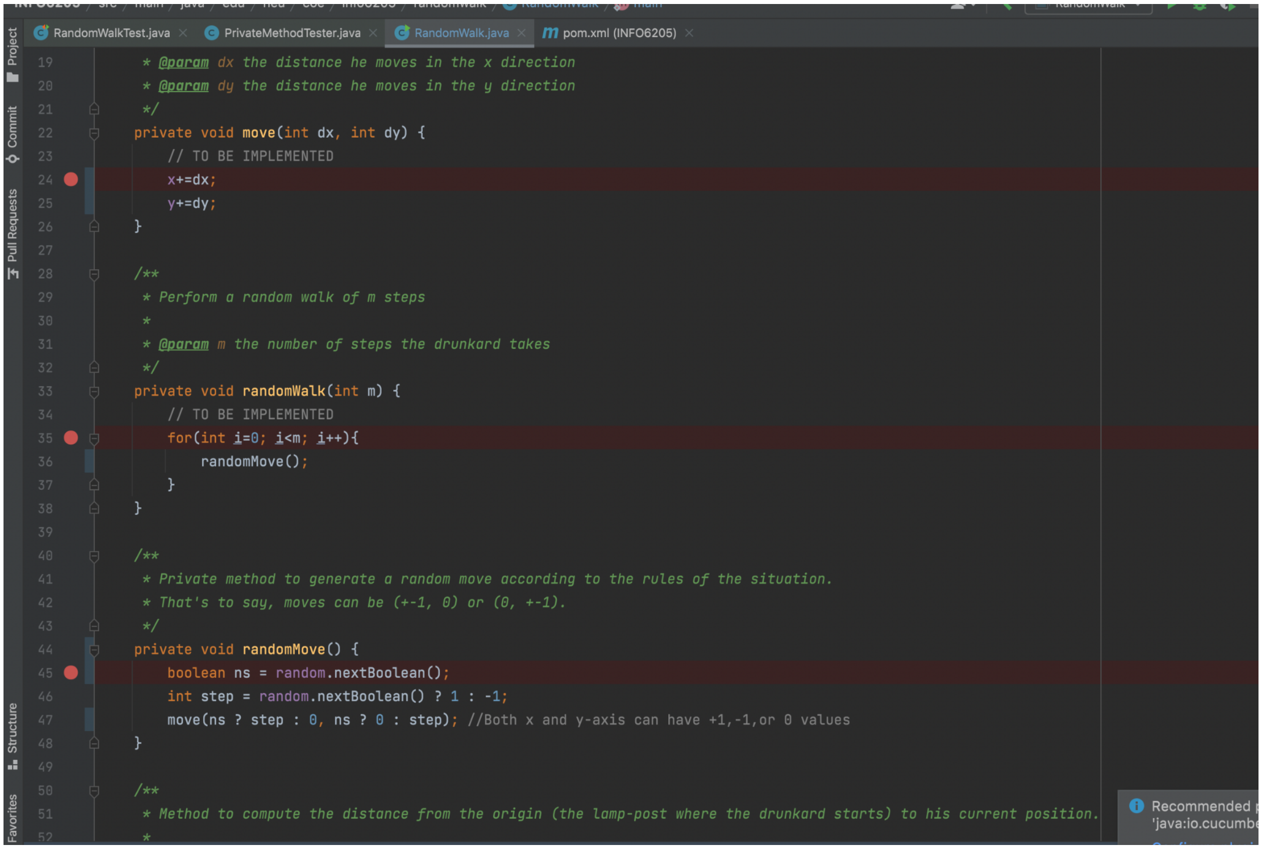
Number of Steps	Mean Distance
19	3.87
3	1.64
99	10.18
78	7.91
73	7.88
44	6
35	5.21
99	8.62
16	3.22
21	3.59
25	4.02
89	8.06
22	4.14
32	5.23
9	2.82

Mean Distance vs Number of Steps



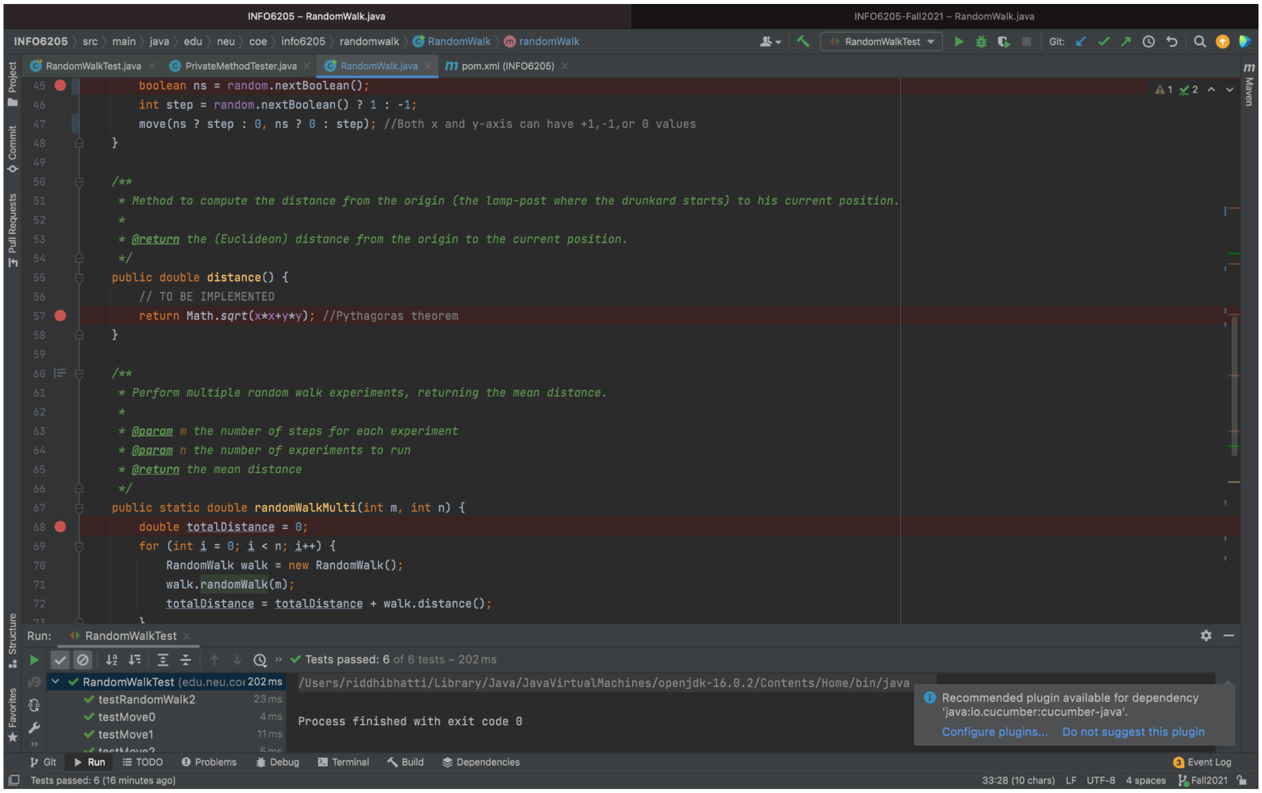
```
/Users/riddhibhatti/Library/Java/JavaVirtualMachines/openjdk-16.0.2/Contents/Home/bin/java ...
19 steps: Mean distance =3.87 over 104 experiments
19,3.87
3 steps: Mean distance =1.64 over 90 experiments
3,1.64
99 steps: Mean distance =10.18 over 91 experiments
99,10.18
78 steps: Mean distance =7.91 over 52 experiments
78,7.91
73 steps: Mean distance =7.88 over 68 experiments
73,7.88
44 steps: Mean distance =6.0 over 20 experiments
44,6.0
35 steps: Mean distance =5.21 over 92 experiments
35,5.21
99 steps: Mean distance =8.62 over 36 experiments
99,8.62
16 steps: Mean distance =3.22 over 77 experiments
16,3.22
21 steps: Mean distance =3.59 over 12 experiments
21,3.59
25 steps: Mean distance =4.02 over 34 experiments
25,4.02
89 steps: Mean distance =8.06 over 57 experiments
89,8.06
22 steps: Mean distance =4.14 over 109 experiments
22,4.14
32 steps: Mean distance =5.23 over 92 experiments
32,5.23
9 steps: Mean distance =2.82 over 85 experiments
9,2.82

Process finished with exit code 0
```

Recommended plugin available for dependency 'java:io.cucumber:cucumber-java'.



```java
     * @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) {
        // TO BE IMPLEMENTED
        x+=dx;
        y+=dy;
    }

    /**
     * Perform a random walk of m steps
     *
     * @param m the number of steps the drunkard takes
     */
    private void randomWalk(int m) {
        // TO BE IMPLEMENTED
        for(int i=0; i<m; i++){
            randomMove();
        }
    }

    /**
     * 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;
        move(ns ? step : 0, ns ? 0 : step); //Both x and y-axis can have +1,-1,or 0 values
    }

    /**
     * Method to compute the distance from the origin (the lamp-post where the drunkard starts) to his current position.
     *
```

Recommended p
'java:io.cucumbe


INFO6205 – RandomWalk.java
INFO6205-Fall2021 – RandomWalk.java
INFO6205 〉 src 〉 main 〉 java 〉 edu 〉 neu 〉 coe 〉 info6205 〉 randomwalk 〉 RandomWalk 〉 randomWalk
RandomWalkTest
Git:
RandomWalkTest.java
PrivateMethodTester.java
RandomWalk.java
pom.xml (INFO6205)
Project
Commit
Pull Requests
Structure
Favorites
Maven
⚠1 ✓2

```
        boolean ns = random.nextBoolean();
        int step = random.nextBoolean() ? 1 : -1;
        move(ns ? step : 0, ns ? 0 : step); //Both x and y-axis can have +1,-1,or 0 values
    }

    /**
     * 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() {
        // TO BE IMPLEMENTED
        return Math.sqrt(x*x+y*y); //Pythagoras theorem
    }

    /**
     * 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();
```

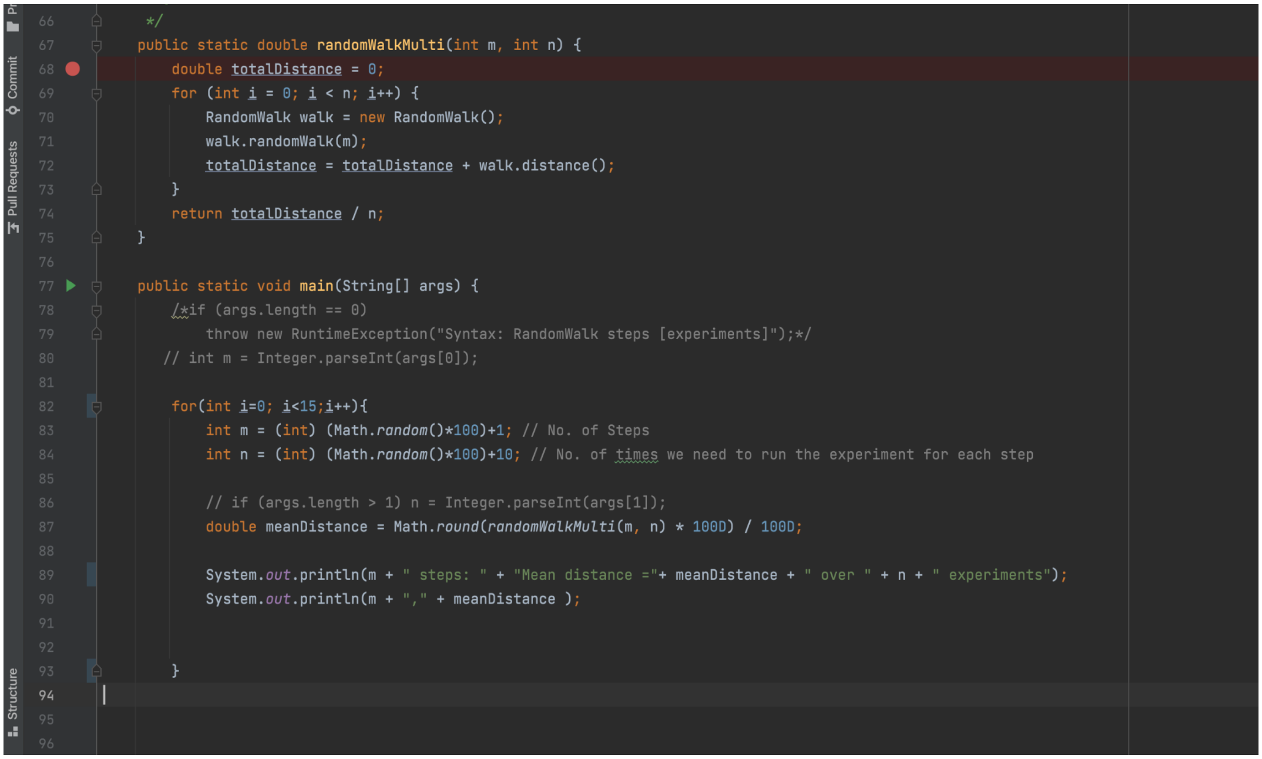
✓ Tests passed: 6 of 6 tests – 202 ms

✓ RandomWalkTest (edu.neu.coe	202 ms
✓ testRandomWalk2	23 ms
✓ testMove0	4 ms
✓ testMove1	11 ms
✓ testMove2	5 ms

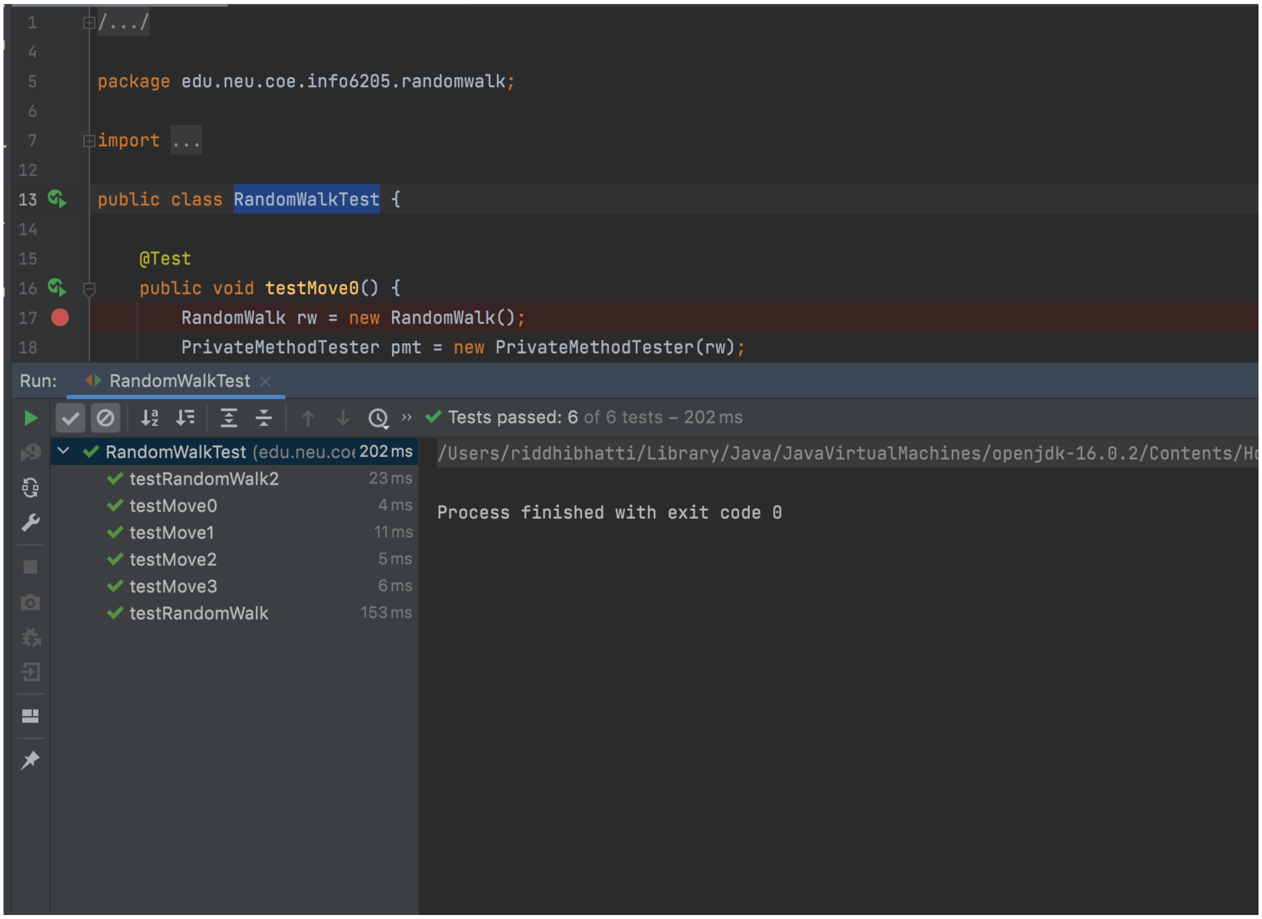
```
/Users/riddhibhatti/Library/Java/JavaVirtualMachines/openjdk-16.0.2/Contents/Home/bin/java

Process finished with exit code 0
```

Recommended plugin available for dependency 'java:io.cucumber:cucumber-java'.
Configure plugins... Do not suggest this plugin

Tests passed: 6 (16 minutes ago)



```
     */
    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) {
        /*if (args.length == 0)
            throw new RuntimeException("Syntax: RandomWalk steps [experiments]");*/
       // int m = Integer.parseInt(args[0]);

        for(int i=0; i<15;i++){
            int m = (int) (Math.random()*100)+1; // No. of Steps
            int n = (int) (Math.random()*100)+10; // No. of times we need to run the experiment for each step

            // if (args.length > 1) n = Integer.parseInt(args[1]);
            double meanDistance = Math.round(randomWalkMulti(m, n) * 100D) / 100D;

            System.out.println(m + " steps: " + "Mean distance ="+ meanDistance + " over " + n + " experiments");
            System.out.println(m + "," + meanDistance );


        }

```



```
/.../

package edu.neu.coe.info6205.randomwalk;

import ...

public class RandomWalkTest {

    @Test
    public void testMove0() {
        RandomWalk rw = new RandomWalk();
        PrivateMethodTester pmt = new PrivateMethodTester(rw);
```

```
Tests passed: 6 of 6 tests – 202 ms

RandomWalkTest (edu.neu.coe  202 ms
    testRandomWalk2           23 ms
    testMove0                  4 ms
    testMove1                 11 ms
    testMove2                  5 ms
    testMove3                  6 ms
    testRandomWalk           153 ms
```

```
/Users/riddhibhatti/Library/Java/JavaVirtualMachines/openjdk-16.0.2/Contents/Ho

Process finished with exit code 0
```