Apoorva Kethamaranahalli (001090928)

Program Structures & Algorithms

Fall 2021

Assignment No. 1

- **Task:**

  Imagine a drunken man who, starting out leaning against a lamp post in the middle of an open space, takes a series of steps of the same length: 1 meter. The direction of these steps is randomly chosen from North, South, East, or West. **After n steps, how far (*d*) is the man from the lamp post?** Note that *d* is the Euclidean distance of the man from the lamppost.

  Task list:

  1. implement the code for the experiment

  2. To deduce the relationship between the distance(d) of the drunken man from the pole, and the number of steps he takes(n)

- **Relationship Conclusion:**

  On running the main method in the RandomWalk.java class multiple times, I established the following conclusion between number of steps(n) and distance(d) after analysis in an excel sheet by plotting the values of distance and square root of the number of steps.

  $d = \sqrt{n} + V$

  where V = variance

  $\therefore d \propto \sqrt{n}$

  i.e.,

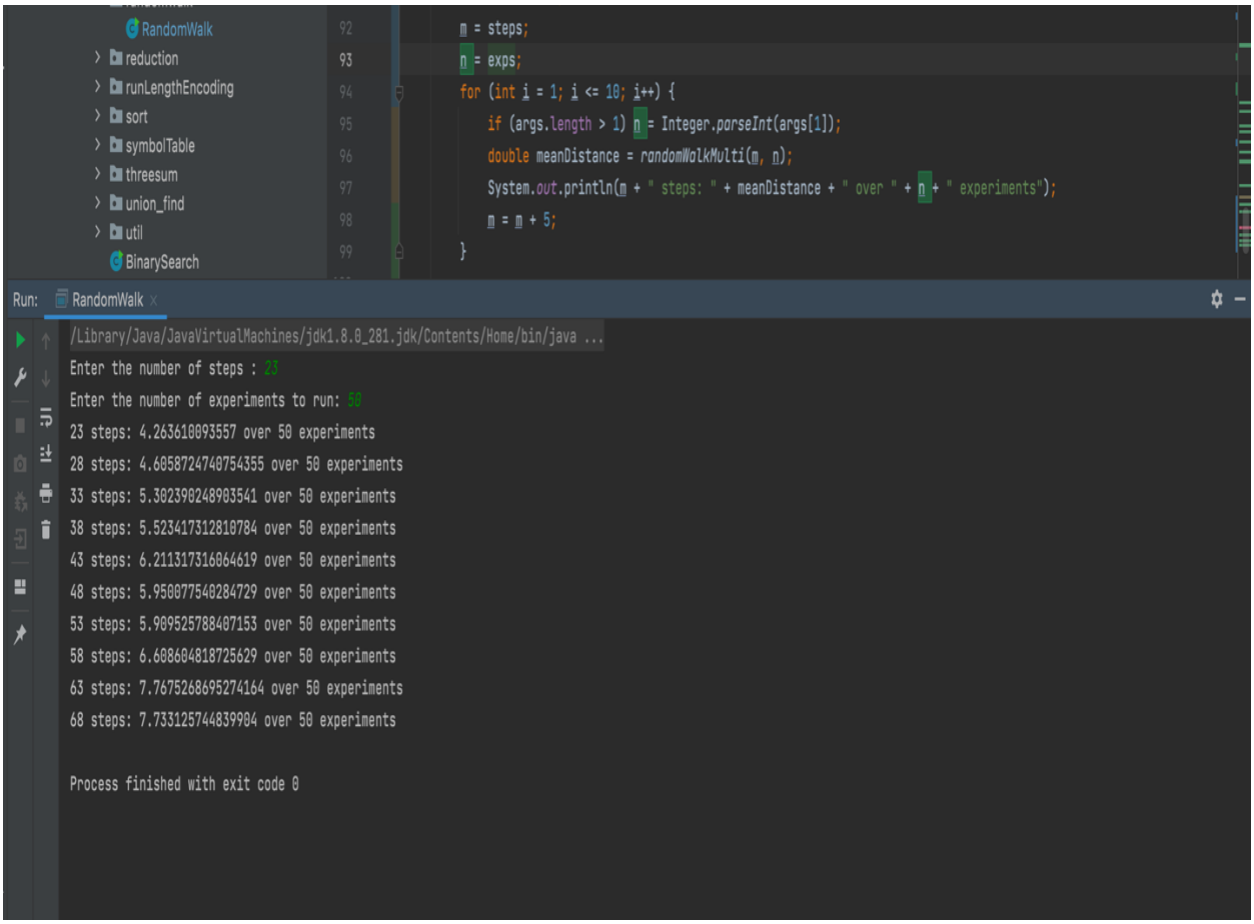
  The distance (d) is directly proportional to the square root of the number of steps(n) the drunken man takes.

- **Evidence to support Conclusion:**

1.Output

In order to test different values of n(steps) along with varying the experiments, I have made changes to the main function to run the different values in a single run. The following 3 experiments are shown below:

a) Experiment 1 : Number of experiments = 50 with 10 different values of n



b) Experiment 2: Number of experiments = 100 with 20 different values of n

```
INFO6205 > src > main > java > edu > neu > coe > info6205 > randomwalk > RandomWalk > main
Project
  life
  pq
  randomwalk
    RandomWalk
  reduction
  runLengthEncoding
RandomWalk.java   RandomWalkTest.java
n                                                    18 results
        if (args.length > 1) n = Integer.parseInt(args[1]);
        double meanDistance = randomWalkMulti(m, n);
        System.out.println(m + " steps: " + meanDistance + " over " + n + " experiments");
        m = m + 2;
    }

Run: RandomWalk
Enter the number of steps : 50
Enter the number of experiments to run: 100
50 steps: 6.136191268048002 over 100 experiments
52 steps: 6.873862450892359 over 100 experiments
54 steps: 6.719992469746939 over 100 experiments
56 steps: 6.641421367884346 over 100 experiments
58 steps: 7.024240547937369 over 100 experiments
60 steps: 6.6584726948220325 over 100 experiments
62 steps: 7.064217778117521 over 100 experiments
64 steps: 7.193199905100232 over 100 experiments
66 steps: 6.8627627743061215 over 100 experiments
68 steps: 7.43359560487113 over 100 experiments
70 steps: 6.726607465486414 over 100 experiments
72 steps: 7.714880421672039 over 100 experiments
74 steps: 8.213535364733751 over 100 experiments
76 steps: 7.932064558069302 over 100 experiments
78 steps: 7.305749537398409 over 100 experiments
80 steps: 7.686958913230428 over 100 experiments
82 steps: 7.31350193969255 over 100 experiments
84 steps: 7.9782194663017325 over 100 experiments
86 steps: 8.461067247629167 over 100 experiments
88 steps: 8.531966022229826 over 100 experiments

Process finished with exit code 0
```

c) Experiment 3: Number of experiments= 100 with 10 different values of n

```
  life
  pq
  randomwalk
    RandomWalk
  reduction
  runLengthEncoding
  sort
  symbolTable
  threesum
        n = exps;
        for (int i = 1; i <= 10; i++) {
            if (args.length > 1) n = Integer.parseInt(args[1]);
            double meanDistance = randomWalkMulti(m, n);
            System.out.println(m + " steps: " + meanDistance + " over " + n + " experiments");
            m = m + 100;
        }

RandomWalk
/Library/Java/JavaVirtualMachines/jdk1.8.0_281.jdk/Contents/Home/bin/java ...
Enter the number of steps : 50
Enter the number of experiments to run: 100
50 steps: 5.463764285566255 over 100 experiments
150 steps: 11.027566312055251 over 100 experiments
250 steps: 14.873023572232812 over 100 experiments
350 steps: 16.62976123996525 over 100 experiments
450 steps: 17.62556048570808 over 100 experiments
550 steps: 21.12373592327517 over 100 experiments
650 steps: 23.11396621443269 over 100 experiments
750 steps: 24.005297735957416 over 100 experiments
850 steps: 25.049565663869508 over 100 experiments
950 steps: 28.465671861888396 over 100 experiments

Process finished with exit code 0
```

2. Graphical Representation:

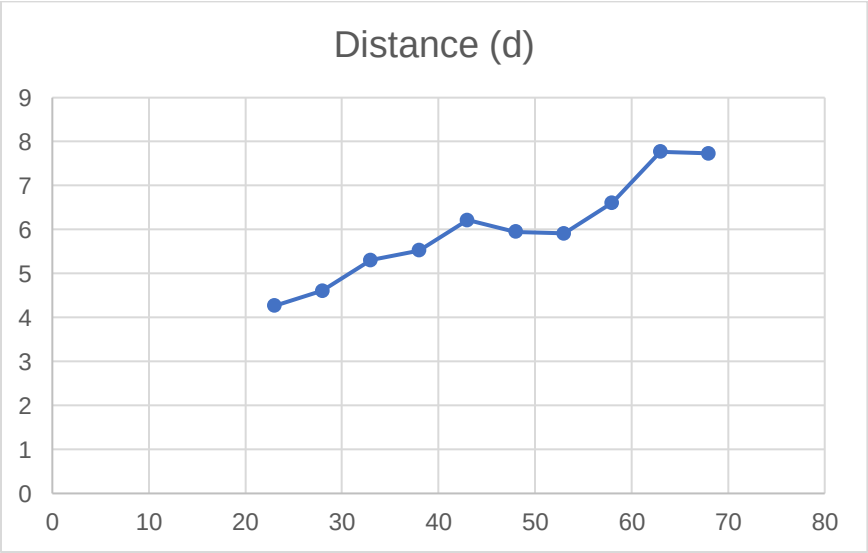
The graphical representations shown below are for three different experiments, as detailed in the excel file attached.

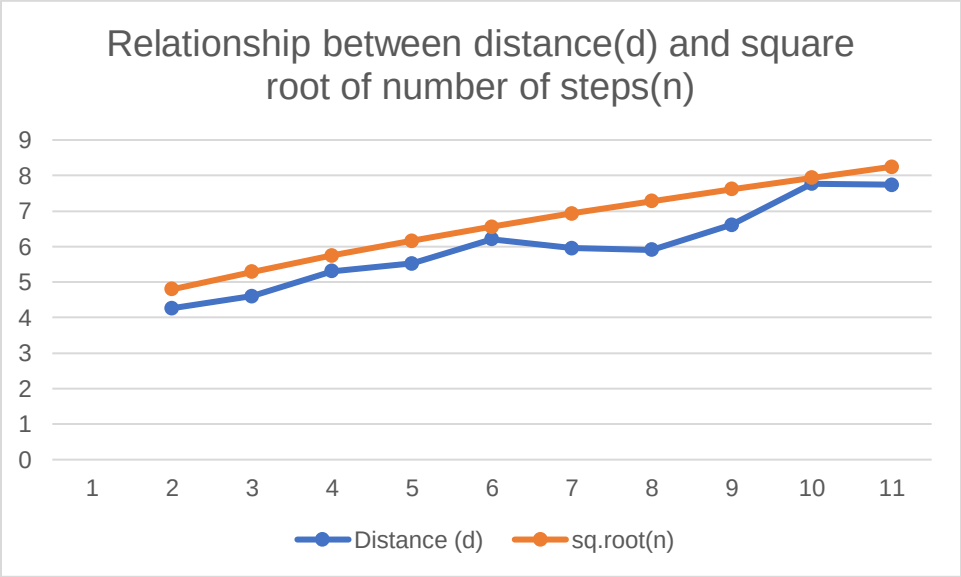
a) Experiment 1 :

Number of experiments = 50 ;

10 different values of n

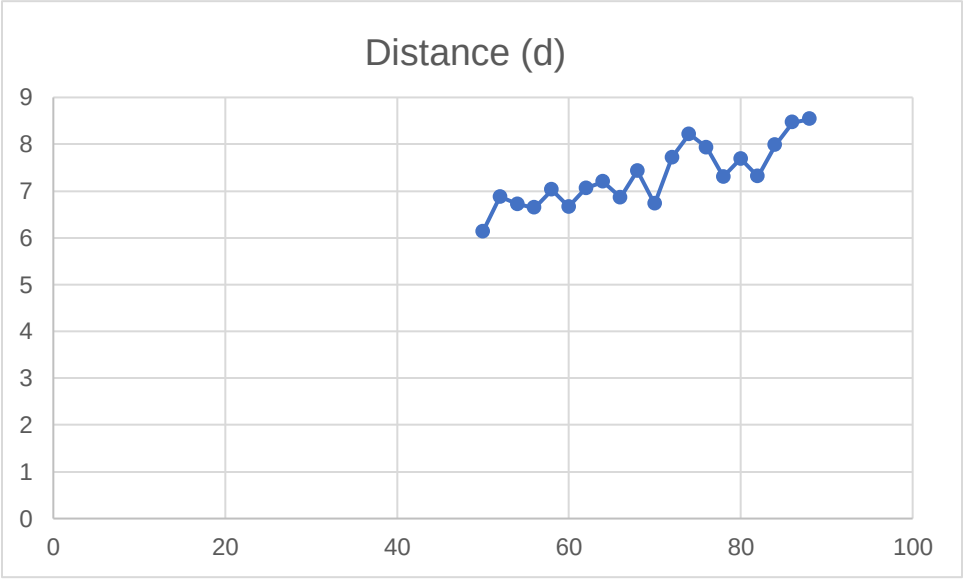
Steps (n)	Distance (d)	sq.root(n)
23	4.263610094	4.79583152
28	4.605872474	5.29150262
33	5.302390249	5.74456265
38	5.523417313	6.164414
43	6.211317316	6.55743852
48	5.95007754	6.92820323
53	5.909525788	7.28010989
58	6.608604819	7.61577311
63	7.76752687	7.93725393
68	7.733125745	8.24621125

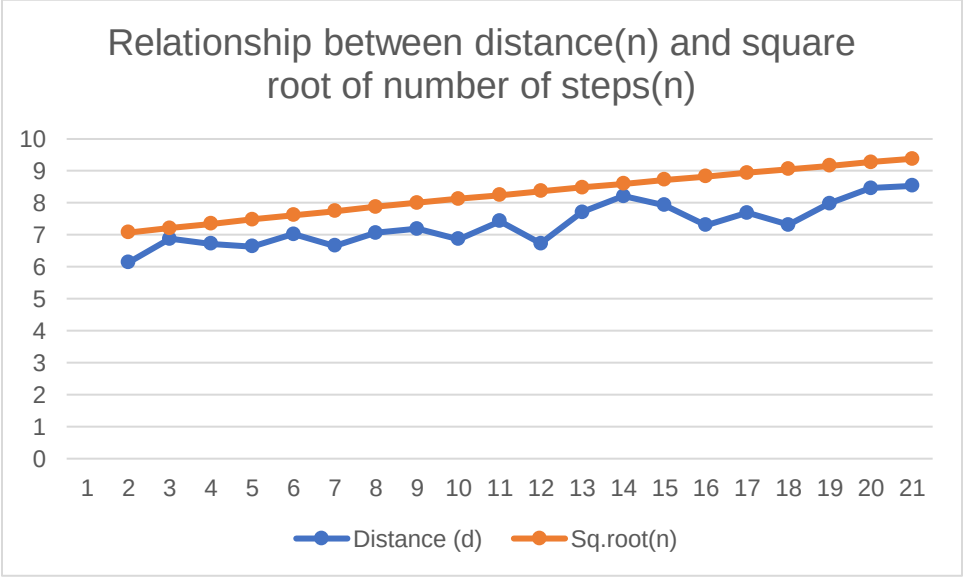




b) Experiment 2:
Number of experiments = 100;
20 different values of n

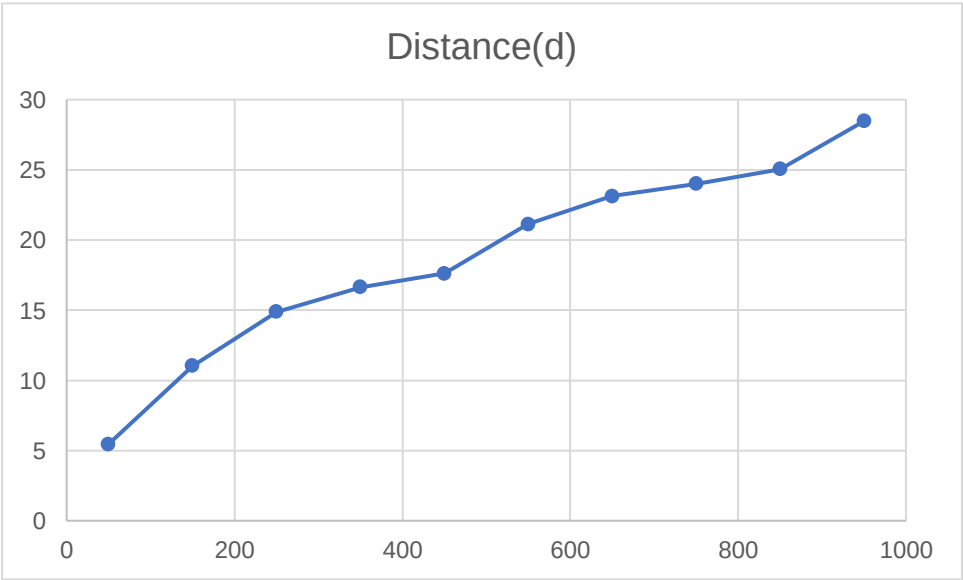
Steps(n)	Distance (d)	Sq.root(n)
50	6.136191268	7.07106781
52	6.873862451	7.21110255
54	6.71999247	7.34846923
56	6.641421368	7.48331477
58	7.024240548	7.61577311
60	6.658472695	7.74596669
62	7.064217778	7.87400787
64	7.193199905	8
66	6.862762774	8.12403841
68	7.433595605	8.24621125
70	6.726607465	8.36660027
72	7.714880422	8.48528137
74	8.213535365	8.60232527
76	7.932064558	8.71779789
78	7.305749537	8.83176087
80	7.686958913	8.94427191
82	7.31350194	9.05538514
84	7.978219466	9.16515139
86	8.461067248	9.2736185
88	8.531966022	9.38083152

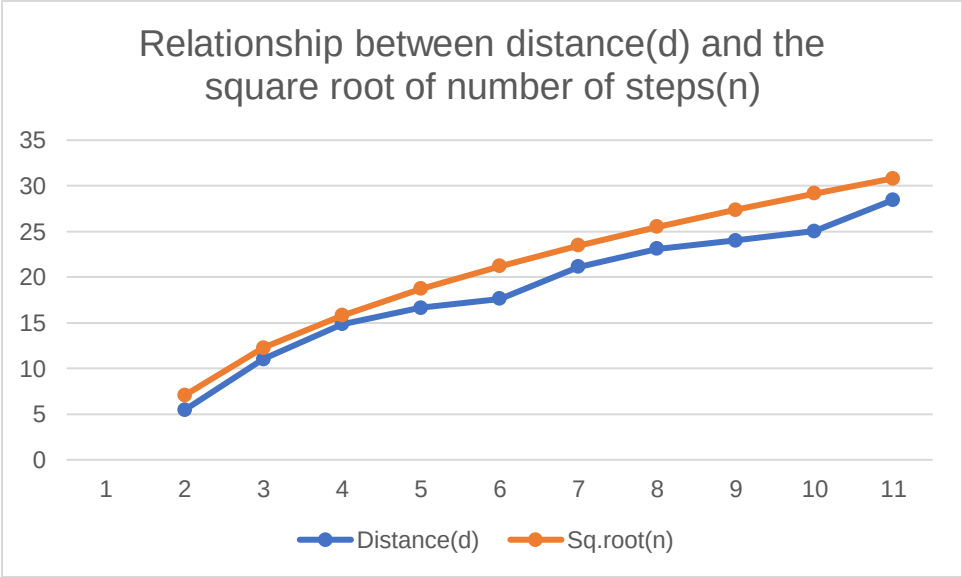

Distance (d)
9
8
7
6
5
4
3
2
1
0
0
20
40
60
80
100


Relationship between distance(n) and square root of number of steps(n)
10
9
8
7
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
Distance (d)
Sq.root(n)

c) Experiment 3:
Number of experiments = 100;
10 different values of n

Steps(n)	Distance(d)	Sq.root(n)
50	5.463764286	7.07106781
150	11.02756631	12.2474487
250	14.87302357	15.8113883
350	16.62976124	18.7082869
450	17.62556049	21.2132034
550	21.12373592	23.4520788
650	23.11396621	25.4950976
750	24.00529774	27.3861279
850	25.04956566	29.1547595
950	28.46567186	30.82207





- **Unit tests result:**