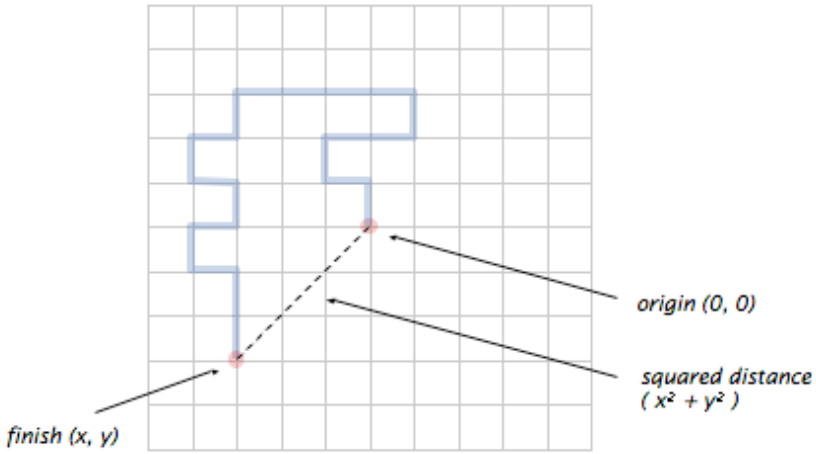
CSC 220 – Project #1

Project Details:
A drunkard's walk. A drunkard begins walking aimlessly, starting at a lamp post. At each time step, the drunkard forgets where he or she is, and takes one step **at random**, either north, east, south, or west, with probability 25%. How far will the drunkard be from the lamp post after *N* steps?

1. Write a program `RandomWalker.java` that takes a command-line argument `N` and simulates the motion of a random walker for `N` steps. After each step, print the location of the random walker, treating the lamp post as the origin (0, 0). Also, print the square of the distance from the origin.




```
$java RandomWalker 10
(0, 0)
(0, -1)
(0, 0)
(0, 1)
(0, 2)
(-1, 2)
(-2, 2)
(-2, 1)
(-1, 1)
(-2, 1)
(-3, 1)
squared distance = 10
```

```
$java RandomWalker 20
(0, 0)
(0, 1)
(-1, 1)
(-1, 2)
(0, 2)
(1, 2)
(1, 3)
(0, 3)
(-1, 3)
(-2, 3)
(-3, 3)
(-3, 2)
(-4, 2)
(-4, 1)
(-3, 1)
(-3, 0)
(-4, 0)
(-4, -1)
(-3, -1)
(-3, -2)
(-3, -3)
squared distance = 18
```

2. Write a program `RandomWalkers.java` that takes two command-line argument `N` and `T`. In each of `T` independent experiments, simulate a random walk of `N` steps and compute the squared distance. Output the *mean squared distance* (the average of the `T` squared distances).

```
% java RandomWalkers 100 100000
mean squared distance = 100.15086

% java RandomWalkers 100 100000
mean squared distance = 99.95274

% java RandomWalkers 200 100000
mean squared distance = 199.57664
```

```
% java RandomWalkers 400 100000
mean squared distance = 401.22024

% java RandomWalkers 800 100000
mean squared distance = 797.5106

% java RandomWalkers 1600 100000
mean squared distance = 1600.13064
```

As *N* increases, we expect the random walker to end up further and further away from the origin. But how much further? Use `RandomWalkers` to formulate a hypothesis as to how the mean squared distance grows as a function of *N*. Use *T* = 100,000 trials to get a sufficiently accurate estimate.

*Remark: this process is a discrete version of a natural phenomenon known as **Brownian motion**. It serves as a scientific model for an astonishing range of physical processes from the dispersion of ink flowing in water, to the formation of polymer chains in chemistry, to cascades of neurons firing in the brain.*

Readable code:
Make sure that your code is readable: use descriptive variable names. Indent your code. Comment your code according to the commenting guidelines.

What to turn in:
JAR your `*.java` and `*.class` files into a file called `Project1.jar`. Upload the jar file to Canvas under category `Project1`.

Hint:

Please note that you will submit two JAVA programs, `RandomWalker.java` and `RandomWalkers.java`. You will use for loop in this project. The first program will use a *for loop* to simulate the N steps. The second program will use two nested *for loops*, the outer loop to simulate T trials and the inner loop to simulate N steps.

The following the is the incomplete version of RandomWalker.java. Your program can be based on this framework.

```
/*************************************************************************
*
* Programmer: your name
* Instructor: Dr. Tao
* Course: CSC220-03
* Compilation:  javac RandomWalkers.java
*  Execution:    java RandomWalkers N
*
*  Simulates how far away after N steps random walk
*
*
*************************************************************************
/

public class RandomWalker {

   public static void main(String[] args) {
      int N = Integer.parseInt(args[0]);

      int x = 0;       // starting x position
      int y = 0;       // starting y position
      double r;

      // repeat until walk N steps
      int step;


   }
}
```