

MIT OpenCourseWare
<http://ocw.mit.edu>

6.00 Introduction to Computer Science and Programming
Fall 2008

For information about citing these materials or our Terms of Use, visit: <http://ocw.mit.edu/terms>.

6.00 Handout, Lecture 17
(Not intended to make sense outside of lecture)

```
import math, random, pylab

class Location(object):
    def __init__(self, x, y):
        self.x = float(x)
        self.y = float(y)
    def move(self, xc, yc):
        return Location(self.x+float(xc), self.y+float(yc))
    def getCoords(self):
        return self.x, self.y
    def getDist(self, other):
        ox, oy = other.getCoords()
        xDist = self.x - ox
        yDist = self.y - oy
        return math.sqrt(xDist**2 + yDist**2)

class CompassPt(object):
    possibles = ('N', 'S', 'E', 'W')
    def __init__(self, pt):
        if pt in self.possibles: self.pt = pt
        else: raise ValueError('in CompassPt.__init__')
    def move(self, dist):
        if self.pt == 'N': return (0, dist)
        elif self.pt == 'S': return (0, -dist)
        elif self.pt == 'E': return (dist, 0)
        elif self.pt == 'W': return (-dist, 0)
        else: raise ValueError('in CompassPt.move')

class Field(object):
    def __init__(self, drunk, loc):
        self.drunk = drunk
        self.loc = loc
    def move(self, cp, dist):
        oldLoc = self.loc
        xc, yc = cp.move(dist)
        self.loc = oldLoc.move(xc, yc)
    def getLoc(self):
        return self.loc
    def getDrunk(self):
        return self.drunk

class Drunk(object):
    def __init__(self, name):
        self.name = name
    def move(self, field, time = 1):
        if field.getDrunk() != self:
            raise ValueError('Drunk.move called with drunk not in field')
        for i in range(time):
            pt = CompassPt(random.choice(CompassPt.possibles))
            field.move(pt, 1)
```

```

def performTrial(time, f):
    start = f.getLoc()
    distances = [0.0]
    for t in range(1, time + 1):
        f.getDrunk().move(f)
        newLoc = f.getLoc()
        distance = newLoc.getDist(start)
        distances.append(distance)
    return distances

drunk = Drunk('Homer Simpson')
for i in range(3):
    f = Field(drunk, Location(0, 0))
    distances = performTrial(500, f)
    pylab.plot(distances)
pylab.title('Homer\'s Random Walk')
pylab.xlabel('Time')
pylab.ylabel('Distance from Origin')

def performSim(time, numTrials):
    distLists = []
    for trial in range(numTrials):
        d = Drunk('Drunk' + str(trial))
        f = Field(d, Location(0, 0))
        distances = performTrial(time, f)
        distLists.append(distances)
    return distLists

def ansQuest(maxTime, numTrials):
    means = []
    distLists = performSim(maxTime, numTrials)
    for t in range(maxTime + 1):
        tot = 0.0
        for distL in distLists:
            tot += distL[t]
        means.append(tot/len(distL))
    pylab.figure()
    pylab.plot(means)
    pylab.ylabel('distance')
    pylab.xlabel('time')
    pylab.title('Average Distance vs. Time (' + str(len(distLists)) + ' trials)')

ansQuest(500, 300)
pylab.show()

```