6.00 Handout, Lecture 18
(Not intended to make sense outside of lecture)

```
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, 100)
pylab.show()
```

```
from pylab import *
import random

plot([1,2,3,4])
plot([5,6,7,8])
plot([1,2,3,4], [1,4,9,16])
figure()
plot([1,2,3,4], [1,4,9,16], 'ro')
axis([0, 6, 0, 20])
title('Earnings')
xlabel('Days')
ylabel('Dollars')
figure()
xAxis = array([1,2,3,4])
print xAxis
test = arange(1,5)
print test
print test == xAxis
yAxis = xAxis**3
plot(xAxis, yAxis, 'ro')
figure()
vals = []
dieVals = [1,2,3,4,5,6]
for i in range(10000):
   vals.append(random.choice(dieVals)+random.choice(dieVals))
hist(vals, bins=11)
show()
```

```
class Drunk(object):
  def __init__(self, name):
    self.name = name
  def move(self, field, cp, dist = 1):
    if field.getDrunk().name != self.name:
      raise ValueError('Drunk.move called with drunk not in field')
    for i in range(dist):
      field.move(cp, 1)

class UsualDrunk(Drunk):
  def move(self, field, dist = 1):
    cp = random.choice(CompassPt.possibles)
    Drunk.move(self, field, CompassPt(cp), dist) #Note notation of call

class ColdDrunk(Drunk):
  def move(self, field, dist = 1):
    cp = random.choice(CompassPt.possibles)
    if cp == 'S':
      Drunk.move(self, field, CompassPt(cp), 2*dist)
    else:
      Drunk.move(self, field, CompassPt(cp), dist)

class EWDrunk(Drunk):
  def move(self, field, time = 1):
    cp = random.choice(CompassPt.possibles)
    while cp != 'E' and cp != 'W':
      cp = random.choice(CompassPt.possibles)
    Drunk.move(self, field, CompassPt(cp), time)

def performSim(time, numTrials, drunkType):
  distLists = []
  for trial in range(numTrials):
    d = drunkType('Drunk' + str(trial))
  …

def ansQuest(maxTime, numTrials, drunkType, title):
  means = []
  distLists = performSim(maxTime, numTrials, drunkType)
  …

ansQuest(500, 100, UsualDrunk, ‘UsualDrunk’)
```

```
class oddField(Field):
  def isChute(self):
    x, y = self.loc.getCoords()
    return abs(x) - abs(y) == 0
  def move(self, cp, dist):
    Field.move(self, cp, dist)
    if self.isChute():
      self.loc = Location(0, 0)
```