Microquiz 3

1.

```
def f():   ## deterministic
    random.seed(0)
    L = []
    for i in range(10000000):
        r = random.random()
        if r < 0.00001:
            L.append(i)
    return L

def g():  ## stochastic
    L = []
    random.seed()
    for i in range(10000000):
        r = random.random()
        if r < 0.00001:
            L.append(i)
    return L

def h():  ## deterministic
    r = random.randint(1,10)
    if r == 0:
        print("Done")
```

2.

Which of the following are implied by the central limit theorem? Choose all that apply.

- Given a sufficiently large set of samples drawn randomly from the same population, the means of the samples (the sample means) will be approximately uniformly distributed.
- → Given a sufficiently large set of samples drawn randomly from the same population, the means of the samples (the sample means) will be approximately normally distributed.
- → Given a sufficiently large set of samples drawn randomly from the same population, the mean of the sample means will be close to the mean of the population.
- Given a sufficiently large set of samples drawn randomly from the same population, the variance of the sample means will be close to the variance of the population.

3.

John wrote a Monte Carlo simulation to estimate the value of the constant K. He ran the simulation 1000 times. The mean estimate of the value of K was 11, and the standard deviation was 2. Which of the following conclusions can be drawn from this? Check all that apply.

- → If the simulation were run again, with a probability greater than 0.9 the estimate of K would be between 9 and 13.
- With a probability of approximately 0.9, the true value of K is between 11 and 13.
- With a probability of approximately 0.95, the true value of K is between 11 and 13.

4.

Assume the following classes are given, based on the Drunk class shown in lecture. The image shows a simulation of a drunk walking from the origin once. The simulation is repeated for 3 to 10 steps.

```
class ADrunk(Drunk):
    def takeStep(self):
```

```
        stepChoices = [(0.0,-0.5), (0.0,-0.5),
                       (-0.5,0.0), (-0.5,0.0)]
        return random.choice(stepChoices)

class BDrunk(Drunk):
    def takeStep(self):
        stepChoices = [(0.0,0.5), (0.0,0.5),
                       (0.5,0.0), (0.5,0.0)]
        return random.choice(stepChoices)
```

➔ It is not possible to tell which of the lines was generated by ADrunk or BDrunk.

5.

```
def ta_activities(trials, grading, teaching, attending):
    '''
    trials: integer, number of trials to run
    grading: probability a TA is grading, 0 <= p <= 1
    teaching: probability a TA is teaching, 0 <= p <= 1
    attending: probability a TA is attending class, 0 <= p <= 1

    Runs a Monte Carlo simulation 'trials' times. Returns: a tuple of
    (1) a float representing the mean num of days it takes to have a day in
        which all 3 actions take place
    (2) the total width of the 95% confidence interval around that mean
        (using stddev)
    '''
    days_list = []
    for trial in range(trials):
        days = 1
        while (random.random() > grading) or \
              (random.random() > teaching) or \
              (random.random() > attending):
            days += 1
        days_list.append(days)
    (mean, std) = get_mean_and_stddev(days_list)
    return (mean, 1.96*std*2)
```