**True/False**

Let D1 and D2 be two different normal distributions. If a sample, s1, is one standard deviation from the mean of D1 and a sample, s2, is two standard deviations from the mean of D2, then s2 is always further (in an absolute sense) from the mean of D2 than s1 is from the mean of D1.

False

Increasing the number of buckets in a hash table typically decreases the number of collisions.

True

For a Gaussian distribution decreasing the confidence interval (e.g., from +/-5 to +/-4) decreases the confidence level (e.g., from 95% to 90%).

True

Increasing the number of buckets in a hash table typically increases the amount of time needed to locate a value in the table.

False

“Normal distribution” and “Gaussian distribution” are different names for the same thing.

True

“Standard deviation” and “coefficient of variation” are different names for the same thing.

False

Given numpy arrays a = [4, 3] and b = [2, 1], c = a – b successfully evaluates to [2, 2].

True

A greedy algorithm reaches the overall optimal solution quicker than a brute force algorithm.

False or True

Given n independent fair coin tosses, the probability of getting at least 1 head is (n-1)/n.

False

Disjoint events are always dependent.

True

**Optimization**

If pylab.polyfit is used to fit an nth degree and an (n+1)th degree polynomial to the same data, is one guaranteed to provide a least squares fit that is at least as good as the other? If so, which and why? If not, why not?

Yes. The (n+1)th degree fit will be at least as tight, because it always possible to set unnecessary coefficients to 0.
def average_longest_streak(p, n, num_trials):
    """Runs a Monte Carlo simulation with a number of trials num_trials that finds the average length of the longest streak of heads when a coin is flipped n times."""
    length = 0
    for t in range(num_trials):
        h_times = 0
        h_max = 0
        for f in range(n):
            h = random.random()
            if h < p:
                h_times += 1
                if h_times > h_max:
                    h_max = h_times
            else:
                h_times = 0
        length += h_max
    return length/float(num_trials)

def plot_streaks(numTrials, longestStreak, title):
    """
    - numTrials and longestStreak, lists of numbers
    - title, the title of the plot
    - Creates a line plot with the x-axis label 'Number of Trials',
      the y-axis label 'Longest Streak', and the title passed in.
    - Does not return anything
    """
    pylab.xlabel("Number of Trials")
    pylab.ylabel("Longest Streak")
    pylab.title(title)
    pylab.plot(numTrials, longestStreak)
    pylab.show()