(1) The complement of this problem is: What is the probability that none of the numbers you pick match the winning numbers? You can compute this with: $Pr(\text{1st num is not winning}) \times Pr(\text{2nd num is not winning}) \times Pr(\text{3rd num is not winning})$, and then subtract from 1 to get the answer to the original problem:

$$\begin{aligned} Pr(\text{picking at least one winning number}) &= 1 - Pr(\text{pick no winning numbers}) \\ &= 1 - \frac{97}{100} \times \frac{96}{99} \times \frac{95}{98} = 0.088 \end{aligned}$$

(2) Compute the odds for each of the desribed draws:

$$Pr(\text{first 4 aces, 5th king}) = \frac{4}{52} \times \frac{3}{51} \times \frac{2}{50} \times \frac{1}{49} \times \frac{4}{48} = \frac{1}{3248700}$$

(3) There are $\binom{3}{2}$ ways to choose the two colors and 2^5 ways to paint all five fingernails with the two colors. However, if we just multiply these two terms, we will overcount the sequences in which the same color is used for all five fingernails. One method is to subtract 2 from 2^5 for each two-color configuration, so that we only count the number of ways in which exactly two colors can be used. Then, we can add back the 3 sequences of $RRRRR$, $YYYYY$, and $BBBBB$. This gives us

$$\left[\binom{3}{2}(2^5 - 2)\right] + 3 = 93$$

(4)

a. Let's compute the complement: what is the probability that California is not represented? The sample space is $\binom{100}{50}$. The event that no CA senator is selected is $\binom{98}{50}$, since there are 98 non-CA senators, and 50 spots to fill. Putting this altogether (including subtracting from 1), we get:

$$1 - \frac{\binom{98}{50}}{\binom{100}{50}} = \frac{149}{198}$$

b. Again, the sample space is $\binom{100}{50}$. There are now 50 spots, and each spot has exactly two choices of senators (in order to include one senator from each state). This gives us $2^{50}/\binom{100}{50}$.

(5)

a. The following values are computed below:

$$\begin{aligned} \mathbb{E}(X_1) &= \frac{1}{6}(1 + 2 + \cdots + 6) = \frac{7}{2} \\ var(X_1) &= \left[\frac{1}{6}(1^2 + 2^2 + \cdots + 6^2)\right] - \left[\frac{7}{2}\right]^2 = \frac{35}{12} \end{aligned}$$

b. In this problem, $X = X_1 - X_2$. To compute the expected value, we have $\mathbb{E}(X) = \mathbb{E}(X_1) - \mathbb{E}(X_2) = 7/2 - 7/2 = 0$. For the variance, it is important to note that, unlike expected value, variance is a non-negative number. Thus, $var(X - Y) \neq var(X) - var(Y)$. The variance can be computed as

follows:

$$\begin{aligned} var(X) = var(X_1 - X_2) &= var[(X_1) + (-1)(X_2)] && (1)\\ &= var(X_1) + var[(-1)X_2] && (2)\\ &= var(X_1) + (-1)^2 var(X_2) && (3)\\ &= var(X_1) + var(X_2) && (4)\\ &= 35/12 + 35/12 = 35/6 && (5) \end{aligned}$$

Line 2 is obtained from the property that $var(aX) = a^2 var(X)$, where a is some constant.

c. The expected value, is again 0. To compute the variance, we use the same ideas as in part b:

$$\begin{aligned} var(Y) = var(X_1 - 2X_2 + X_3) &= var(X_1) + 2var(X_2) + var(X_3)\\ &= 35/12 + 2(35/12) + (35/12) = 35/3 \end{aligned}$$

d. The expected value, is again 0. The variance is:

$$\begin{aligned} var(Z) &= var(X_1 - X_2 + X_3 - X_4 + \cdots + X_{99} - X_{100})\\ &= var(X_1) + var(X_2) + \cdots + var(X_{100}) = (100)(35/12) = 875/3 \end{aligned}$$

(6) The cumulative distribution function of Y is:

$$Pr(Y \leq a) = \frac{\pi a^2}{\pi(1)^2} = a^2$$

This is saying that the probability of throwing a dart at a location with distance less than or equal to a is the area of the circle with radius a over the sample space (which is the entire circle, with an area of $\pi r^2 = \pi(1)^2$).

(7) This is very similar to problem 7 on homework 5. Also, to clarify, there are k unique words, and for each of these unique words, there are m instances of them in the text. The goal is to find all of the k words at least once (no need to find all m instances of each word).

a. First, given that you have not found any real words yet the probability of finding the first real word is $\frac{mk}{n}$ since there are a total of mk secret words. Since this is the same probability on each try, the expected number of tries is just the inverse: $\frac{n}{mk}$.

b. When there are k words left to be found, the expected number of tries (as computed in part a) is $\frac{n}{mk}$. When there are $k-1$ words left, the expected number of tries should be $\frac{n}{m(k-1)}$. This continues until there is only one word left, and the expected number of tries is $\frac{n}{m(1)}$. The summation over all of these terms gives us the total expected number of tries before all k unique words are found:

$$\mathbb{E}(\text{num of tries to find all k words}) = \sum_{\ell=1}^{k} \frac{n}{m\ell} = \frac{n}{m}\sum_{\ell=1}^{k} \frac{1}{\ell} = \frac{n \ln k}{m}$$

The final terms comes from the approximation $\sum_{x=1}^{n} \frac{1}{x} = \ln n$.

(8) If x is prime, $\mathcal{A}(x)$ will always be correct. If x is not prime, $\mathcal{A}(x)$ has a $(1/2)$ chance of being correct. Thus, we first focus on the case that x is not prime. In this case, after k calls to the algorithm, the probability that $\mathcal{A}(x)$ never returns the correct answer is $(1 - 1/2)^k$. We want this probability to be

less than $1/(10^9)$. Thus, we find k that satisfies the following equation:

$$\begin{aligned} \frac{1}{2^k} &\leq \frac{1}{10^9} \\ k &\geq \log_{1/2} \frac{1}{10^9} = \log_2 10^9 \\ k &\approx 30 \end{aligned}$$

So, our algorithm is as follows: Run $\mathcal{A}(x)$ k number of times. If $\mathcal{A}(x)$ ever returns "not prime", then we know the true answer must be "not prime", since that is the only condition on which $\mathcal{A}(x)$ will ever output "not prime". If all k outputs are "prime", then it is most likely that the true answer is "prime". However, per our calculations, there is a $1/(10^9)$ chance that the true answer is actually "not prime".

(9) The idea for this problem is to change the widths of each of the n buckets such that each element still has $1/n$ of falling into each bucket. This way, the computation that we did in class – to prove that bucket sort happens in linear expected time – will still hold. Before, all bins had equal width, and thus, we would have set each a_i to be at equal intervals and trivially, $Pr(X_i \leq a_i) = F(a_i) = i/n$. But now, setting a_i at equal intervals would no longer yield $F(a_i) = i/n$ since $F(a)$ is no longer uniform. Thus, what we want to do is to pick each a_i (for all n buckets) such that $F(a_i) = i/n$, and then set the width of the bin to represent the interval from a_{i-1} to a_i.

(10) The algorithm proposed here simply splits all nodes V into two groups, with each node having a $1/2$ chance of belonging to either group.

a. Let $X_j = 1$ if edge j crosses the resulting cut. $\mathbb{E}(X_j) = Pr(X_j = 1) = 1/2$ since there is a $1/2$ chance that the two end nodes belong in different groups. Summing over the expected value of all edges in the graph gives us $\frac{|E|}{2}$.

b. The largest cut that any graph could possible have is if all edges in the graph end up crossing the cut, or $|E|$. We computed in part a that the expected size of the resulting cut is $\frac{|E|}{2}$. Thus, the ratio between the expected size and maximum size is *at least* $\frac{|E|/2}{|E|} = \frac{1}{2}$ since most graphs will have a max cut of less than $|E|$ (making the said ratio even larger). Thus, in expectation, the resulting cut from this algorithm is at least half as big as the maximum cut.