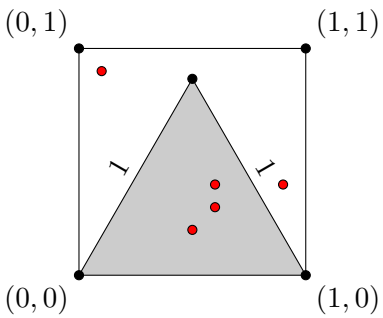
Solutions to Homework 10

Problems marked with **E** are graded on effort, which means that they are graded subjectively on the perceived effort you put into them, rather than on correctness. For bonus questions, we will not provide any insight during office hours or Piazza, and we do not guarantee anything about the difficulty of these questions. We strongly encourage you to typeset your solutions in LaTeX.

If you collaborated with someone, you must state their names. You must write your own solution and may not look at any other student's write-up.

0. If applicable, state the name(s) and uniqname(s) of your collaborator(s).

1. One way to estimate $\sqrt{3}$ is by randomly throwing darts (represented as red points) uniformly into a 1×1 square, similar to how we estimated π in class:



In the above diagram, the gray shaded region is an equilateral triangle.

For any given dart thrown at point (x, y), we can determine if the following are true:

- $3x^2 > y^2$
- $3(1-x)^2 > y^2$

A geometric analysis reveals that a dart thrown at (x, y) lies in the shaded triangle precisely when both of these inequalities are true.

Let p be the probability that a randomly thrown dart lies in the shaded region. In other words, p is the area of the triangle divided by the area of the square. By throwing many darts, we can estimate this p and use it to approximate $\sqrt{3}$.

(a) What is the theoretical value of p? You will need to use this for part (c).

> **Solution:** The probability that a randomly thrown dart falls in the shaded area is the quotient of their areas. The triangle's height is $\sin\frac{\pi}{3} = \frac{\sqrt{3}}{2}$ and its base is 1. Therefore, the area of the triangle is $\frac{1}{2} \cdot 1 \cdot \frac{\sqrt{3}}{2} = \frac{\sqrt{3}}{4}$. The area of the square is $1 \cdot 1 = 1$. So, $p = \frac{\sqrt{3}}{4}$.

(b) Suppose we run t trials of the above process and arrive at an estimate of τ for $\sqrt{3}$. Using Chernoff bounds, give an upper bound for the number of trials n necessary to estimate $\sqrt{3}$ using this method such that

i. $|\tau - \sqrt{3}| < 0.1 = 10^{-1}$ with probability 99%.
ii. $|\tau - \sqrt{3}| < 0.01 = 10^{-2}$ with probability 99%.

iii. $|\tau - \sqrt{3}| < 0.001 = 10^{-3}$ with probability 99%.

Solution: First, we note that $\sqrt{3} = 4p$. If we let σ be our estimate for p, then we get $\tau = 4\sigma$. Now, let Y be the sum of the indicator functions $Y_1, \ldots, Y_n$, where each Y_i is 1 precisely when the ith dart landed in the shaded area. Then $\sigma = \frac{1}{n}Y$. Rewriting so we can apply the Chernoff bound,

$$\begin{aligned}
\Pr\left[\left|\tau - \sqrt{3}\right| \geq \varepsilon\right] &= \Pr\left[\left|4\sigma - \sqrt{3}\right| \geq \varepsilon\right] \\
&= \Pr\left[\left|\sigma - \frac{\sqrt{3}}{4}\right| \geq \frac{1}{4}\varepsilon\right] \\
&= \Pr\left[\left|\frac{1}{n}Y - \frac{\sqrt{3}}{4}\right| \geq \frac{1}{4}\varepsilon\right] \\
&= \Pr\left[\left|\frac{1}{n}Y - p\right| \geq \frac{1}{4}\varepsilon\right].
\end{aligned}$$

Therefore

$$\begin{aligned}
\Pr\left[\left|\tau - \sqrt{3}\right| \geq \varepsilon\right] &\leq 2e^{-2\left(\frac{1}{4}\varepsilon\right)^2 n} \\
&= 2e^{-\frac{1}{8}\varepsilon^2 n}.
\end{aligned}$$

Because we want the left hand side to be at most 0.01 (i.e. $100\% - 99\% = 1\%$), we should pick a value of n that is large enough such that $2e^{-\frac{1}{8}\varepsilon^2 n} \leq 0.01$. Solving for n,

$$\begin{aligned}
e^{-\frac{1}{8}\varepsilon^2 n} &\leq 0.005 \\
-\frac{1}{8}\varepsilon^2 n &\leq \ln(0.005) \\
n &\geq -\frac{8\ln(0.005)}{\varepsilon^2} \\
&= \frac{8\ln(200)}{\varepsilon^2}
\end{aligned}$$

We now just need to plug in different values for ε to the formula $n = \left\lceil \frac{8\ln(200)}{\varepsilon^2} \right\rceil$.

i. $\varepsilon = 0.1$, so $n = 4239$ is an upper bound.

ii. $\varepsilon = 0.01$, so $n = 423866$ is an upper bound.

iii. $\varepsilon = 0.001$, so $n = 42386539$ is an upper bound.

E (c) On Canvas, we have provided a C++ program called `findSqrtThree.cpp` that calculates the value of $\sqrt{3}$ using a random number generator. If you encounter problems running on your computer, please compile and run it on CAEN (we tested that the program should terminate in a reasonable amount of time on CAEN). To use the program, run the commands

```
g++ findSqrtThree.cpp -o approx -std=c++11 -O3
./approx [n]
```

Supply the algorithm with the 3 different values you obtained from part (b) (replacing `[n]` in the above), and truthfully report both the program output (i.e. p value) and the corresponding estimations of $\sqrt{3}$.

Solution: An example set of runs would have output that looks similar to this (some output omitted for brevity):

```
Completed running 4239 iterations.
    Resulting p = 0.441613588

Completed running 423866 iterations.
    Resulting p = 0.433764444

Completed running 42386539 iterations.
    Resulting p = 0.43310132
```

This yields the approximations (found by calculating $p \times 4$):

- For $\varepsilon = 0.1$, we estimate $\sqrt{3} \approx 1.766454352$
- For $\varepsilon = 0.01$, we estimate $\sqrt{3} \approx 1.735057776$
- For $\varepsilon = 0.001$, we estimate $\sqrt{3} \approx 1.73240528$

Note that the actual value of $\sqrt{3}$ is $1.7320508075688772\ldots$

2. You are trying to connect a network of server clusters and decide to take a randomized approach for connecting one cluster to another. To represent this network, we denote by $G_{n,p}$ a random, undirected graph on n vertices constructed as follows: for each distinct set of two vertices $u, v \in V$, include the edge (u, v) in $G_{n,p}$ with probability p.

   (a) What is the expected number of edges in $G_{n,p}$? In other words, what is the expected number of inter-cluster connections in the network?

Solution: Note that there are $\binom{n}{2}$ possible edges in $G_{n,p}$, one for each pair of distinct vertices in $G_{n,p}$. Fix an enumeration of these possible edges. For any $1 \leq i \leq \binom{n}{2}$, let X_i be an indicator variable which is 1 is the ith possible edge is included in $G_{n,p}$ and 0 otherwise. By the construction of $G_{n,p}$, each X_i has $\Pr[X_i = 1] = p$ and all these X_i are mutually independent. The number of edges in $G_{n,p}$ is then

$$X = \sum_{i=1}^{\binom{n}{2}} X_i,$$

so the expected number of edges in $G_{n,p}$ is

$$\mathrm{E}[X] = \sum_{i=1}^{\binom{n}{2}} \mathrm{E}[X_i] = \sum_{i=1}^{\binom{n}{2}} \Pr[X_i = 1] = \sum_{i=1}^{\binom{n}{2}} p = \binom{n}{2} p.$$

(b) Given $v \in G_{n,p}$, what is the expected value of $\deg(v)$? In other words, what is the expected number of other clusters that an arbitrary cluster is connected with?

Solution: Let $v_1, \ldots, v_{n-1}$ be the $n-1$ vertices in $G_{n,p}$ which are distinct from v. For any $1 \le i \le n-1$, let

$$X_i = \begin{cases} 1, & \text{if } (v, v_i) \text{ is an edge in } G_{n,p} \\ 0, & \text{otherwise.} \end{cases}$$

By the construction of $G_{n,p}$, we know

$$\mathrm{E}[X_i] = \Pr[X_i = 1] = p.$$

The degree of v is the expected number of edges adjacent to v i,e, the expected number of vertices v_i where (v, v_i) is an edge in G, so it is given by

$$X = \sum_{i=1}^{n-1} X_i$$

Then the expected degree of v is

$$\mathrm{E}[X] = \sum_{i=1}^{n-1} \mathrm{E}[X_i] = \sum_{i=1}^{n-1} p = p(n-1).$$

Alternatively, we can derive the expected degree from our answer to part a). Note that in any graph $G = (V, E)$,

$$\sum_{v \in V} \deg(v) = 2 \cdot |E|$$

since any edge has two endpoints, and thus contributes 2 to the sum of the degrees of all vertices. If d is the expected degree of a single vertex, the expected value of this sum is then nd, so from part a) we have

$$nd = 2 \cdot \binom{n}{2} p \implies d = \frac{2}{n} \binom{n}{2} p = (n-1)p.$$

(c) Assume $n > 1$. Let D be a blackbox that picks a random vertex (server cluster) $v \in G_{n,p}$ and returns $\deg(v)$. Describe an algorithm that, using D as the only way to get information about edges in G, produces output whose expected value is p. In other words,

solely by looking at the completed network, the algorithm estimates the probability that originally was used to randomly connect server clusters to each other.

Solution: Define

```
1: function M
2:     Let d be the result of querying D.
3:     return d/(n − 1)
```

As described in *b*), the expected degree of a vertex $v \in G_{n,p}$ is $p(n-1)$, so the expected value of d is $p(n-1)$, thus by linearity of expectation

$$\mathrm{E}\left[\frac{d}{n-1}\right] = \frac{1}{n-1} \cdot \mathrm{E}[d] = \frac{1}{n-1} \cdot p(n-1) = p,$$

so the expected output of the algorithm is p.

(d) Let p' be the estimate for p returned by your algorithm in part (c). Use Chernoff bounds to give an upper bound on the probability that $p' \geq 3p$. Your answer should depend on p and n.

Solution: Let the degree of a vertex $v \in G_{n,p}$ as $X = X_1, \ldots, X_{n-1}$ as described in part a). Note that the output of our algorithm is $X/(n-1)$. Since X is a sum of $n-1$ random indicator variables with success probability p, by the upper-tail Chernoff bound we have

$$\Pr\left[\frac{X}{n-1} \geq 3p\right] = \Pr\left[\frac{X}{n-1} \geq p + 2p\right] \leq e^{-2(2p)^2(n-1)} = e^{-8p^2(n-1)}.$$

3. A hash function maps data objects to the indices of an array where they are stored; we would like the function to distribute the objects roughly evenly among the indices. Suppose we need to hash a large number n of objects to k indices. Consider an "ideal" function f that maps each object to a uniformly random and independent index in $\{1, 2, \ldots, k\}$.

(a) Let X_i be a random variable for the number of objects that f hashes to index i. Find, with proof, a closed-form expression for $\mathrm{Ex}[X_i]$.

Solution: Let $X_{i,j}$ be the indicator random variable that is 1 precisely when the jth object is hashed to bucket i. Then, since f is an ideal hash function, $\mathrm{Ex}[X_{i,j}] = \frac{1}{k}$ for every i, j. Now, $X_i = \sum_{j=1}^{n} X_{i,j}$. So, by linearity of expectation, $\mathrm{Ex}[X_i] = \sum_{j=1}^{n} \mathrm{Ex}[X_{i,j}] = \sum_{j=1}^{n} \frac{1}{k} = \frac{n}{k}$.

(b) Find, with proof, an upper bound on the probability that f maps at least $2 \cdot \mathrm{Ex}[X_i]$ different objects to index i:

i. Using Markov's inequality

ii. Using Chernoff bounds

iii. Under what condition on n and k does each method yield a tighter bound?

Solution:

i. By Markov's inequality (which is valid because X_i is a nonnegative random variable), $\Pr[X_i \geq 2\text{Ex}[X_i]] \leq \frac{\text{Ex}[X_i]}{2\text{Ex}[X_i]} = \frac{1}{2}$.

ii. Applying the Chernoff bound (which is valid because X_i is the sum of indicator variables $X_{i,j}$ for $j = 1, \ldots, n$, which are independent because each object is hash independently of all others), we have

$$\begin{aligned}\Pr[X_i \geq 2\text{Ex}[X_i]] &= \Pr\left[X_i \geq \frac{2n}{k}\right] \\ &= \Pr\left[\frac{X_i}{n} \geq \frac{1}{k} + \frac{1}{k}\right] \\ &\leq e^{-2(1/k)^2 n} \\ &= e^{-2n/k^2}.\end{aligned}$$

iii. The Chernoff bound is better exactly when $e^{-2n/k^2} < 1/2$, or equivalently, when $n > \frac{1}{2} \cdot k^2 \ln 2$. This is because, for an upper bound, a lower value yields a tighter bound; for example, if the real value is 0.5 and we have two upper bounds that are 0.6 and 0.7, the bound of 0.6 is closer to the real value and so is tighter.

(c) Assuming that n, k are such that the Chernoff bound is the tighter of the two bounds above, give an upper bound on the probability that f hashes at least $2 \cdot \text{Ex}[X_i]$ objects to index i for *some* index i.

Solution: Let A_i be the event that f hashes at least $2\text{Ex}[X_i] = 2n/k$ objects to index i. From part (b) and the assumption that the Chernoff bound is tighter, we know that $\Pr[A_i] \leq e^{-2n/k^2}$. Now, let A be the union of all of the A_i. In words, A is the event that f hashes at least $2n/k$ objects to *some* index. Then, by the union bound,

$$\begin{aligned}\Pr[A] &= \Pr\left[\bigcup_{i=1}^{k} A_i\right] \\ &\leq \sum_{i=1}^{k} \Pr[A_i] \\ &\leq \sum_{i=1}^{k} e^{-2n/k^2} \\ &= ke^{-2n/k^2}.\end{aligned}$$

E (d) Consider the probability that some index i has more than $2 \cdot \mathrm{Ex}[X_i]$ objects hashing to it. Assuming that $k \ll \sqrt{n}$, describe how this probability behaves as n grows.

Solution: By the assumption $k \ll \sqrt{n}$, we see that n/k^2 grows large as n grows, hence $\Pr[A] = ke^{-2n/k^2}$ will become quite small. We conclude that f is extremely likely to distribute objects "almost evenly" among the indices, i.e., it becomes extremely unlikely (as n grows) that any index will have $2n/k$ or more elements hashing to it.

4. Suppose you are flipping a biased coin n times and want to count how many times you land a tails. Let Y be the total number of tails. Let Y_i represent an indicator variable such that $Y_i = 0$ if the i-th coin toss is a heads and $Y_i = 1$ if the i-th coin toss is a tails. The odds of landing a tails is $\Pr[Y_i = 1] = p$.

   Then $Y = Y_1 + \cdots + Y_n$ is the sum of these independent indicator variables. The upper-tail Chernoff bound states that for any $\varepsilon > 0$ we have

$$\Pr\left[\frac{Y}{n} \geq p + \varepsilon\right] \leq e^{-2\varepsilon^2 n}.$$

   In other words, the probability, after flipping the coin n times, of having a fraction of coin tosses landing tails **at least** $p + \varepsilon$, is no greater than $e^{-2\varepsilon^2 n}$. Note that this bound is a consequence of the union bound and Markov's inequality.

   (a) Assuming the upper-tail Chernoff bound, prove the lower-tail Chernoff bound. That is, prove that for any $\varepsilon > 0$ we have

$$\Pr\left[\frac{Y}{n} \leq p - \varepsilon\right] \leq e^{-2\varepsilon^2 n}.$$

   In other words, prove that the upper-tail Chernoff bound implies that the probability, after flipping the coin n times, of having a fraction of coin tosses landing tails **at most** $p - \varepsilon$, is no greater than $e^{-2\varepsilon^2 n}$.

Solution: We are given a random variable $Y = Y_1 + \cdots + Y_n$, and we need to show that it satisfies the lower tail Chernoff bound. Since we are assuming the upper tail Chernoff bound, we are essentially going to create a new random variable X, which will "invert" Y, in the sense that the upper tail of X will be exactly the lower tail of Y. Then, applying the known upper tail Chernoff bound on X will give us exactly the lower tail Chernoff bound for Y.
For each $1 \leq i \leq n$, let $X_i = 1 - Y_i$, so

$$\Pr[X_i = 1] = \Pr[1 - Y_i = 1] = \Pr[Y_i = 0] = 1 - p.$$

Note that $X_1, \ldots, X_n$ are also independent indicator variables. Applying the upper-tail Chernoff bound to $X = X_1 + \cdots + X_n$, we have

$$\Pr\left[\frac{X}{n} \geq (1 - p) + \varepsilon\right] \leq e^{-2\varepsilon^2 n}.$$

Notice that $X_i + Y_i = 1$, so

$$X + Y = \sum_{i=1}^{n} (X_i + Y_i) = \sum_{i=1}^{n} 1 = n$$

The fact that $X + Y = n$ is very intuitive: what it means is that the number of Heads added to the number of Tails during n coin flips is always n. We divide this by n to find that $\frac{X}{n} = 1 - \frac{Y}{n}$. We can now algebraically show that the upper tail Chernoff bound on X is exactly the same as the lower tail Chernoff on Y.

$$\Pr\left[\frac{X}{n} \geq (1 - p) + \varepsilon\right] = \Pr\left[1 - \frac{Y}{n} \geq 1 - p + \varepsilon\right] = \Pr\left[\frac{Y}{n} \leq p - \varepsilon\right],$$

thus

$$\Pr\left[\frac{Y}{n} \leq p - \varepsilon\right] \leq e^{-2\varepsilon^2 n}.$$

(b) Assuming the lower-tail and upper-tail Chernoff bounds, prove that for any $\varepsilon > 0$ we have

$$\Pr\left[\left|\frac{Y}{n} - p\right| \geq \varepsilon\right] \leq 2e^{-2\varepsilon^2 n}.$$

Solution: We know that for any $\varepsilon > 0$, we have

$$\Pr\left[\frac{Y}{n} \geq p + \varepsilon\right] \leq e^{-2\varepsilon^2 n}.$$

and

$$\Pr\left[\frac{Y}{n} \leq p - \varepsilon\right] \leq e^{-2\varepsilon^2 n}.$$

From the union bound, we have that

$$\begin{aligned}
\Pr\left[\left|\frac{Y}{n} - p\right| \geq \varepsilon\right] &= \Pr\left[\frac{Y}{n} - p \geq \varepsilon \text{ or } \frac{Y}{n} - p \leq -\varepsilon\right] \\
&= \Pr\left[\frac{Y}{n} \geq p + \varepsilon \text{ or } \frac{Y}{n} \leq p - \varepsilon\right] \\
&\leq \Pr\left[\frac{Y}{n} \geq p + \varepsilon\right] + \Pr\left[\frac{Y}{n} \leq p - \varepsilon\right] \\
&\leq e^{-2\varepsilon^2 n} + e^{-2\varepsilon^2 n},
\end{aligned}$$

hence

$$\Pr\left[\left|\frac{Y}{n} - p\right| \geq \varepsilon\right] \leq 2e^{-2\varepsilon^2 n}.$$

(c) Using Reverse Markov's Inequality, find a **lower bound** for $\Pr\left[\frac{Y}{n} > p - \varepsilon\right]$. In other words, find a lower bound for the probability, after flipping the coin n times, of having a fraction of coin tosses landing tails **strictly greater than** $p - \varepsilon$. Then, compare the tightness of this lower bound to the lower bound that can be derived from the complement of the upper-tail Chernoff bound:

$$\Pr\left[\frac{Y}{n} > p - \varepsilon\right] \geq 1 - e^{-2\varepsilon^2 n}.$$

Solution: By the linearity of expectations, $\mathrm{E}[\frac{Y}{n}] = \mathrm{E}[\frac{\sum_i^n Y_i}{n}] = \frac{\sum_i^n \mathrm{E}[Y_i]}{n} = \frac{\sum_i^n p}{n} = \frac{np}{p} = p$. Furthermore, since the number of tails cannot exceed the number of coin tosses, $Y \leq n$, so $\frac{Y}{n} \leq 1$.

Therefore, $\Pr\left[\frac{Y}{n} > p - \varepsilon\right] \geq \frac{\mathrm{E}[\frac{Y}{n}]-(p-\varepsilon)}{1-(p-\varepsilon)} = \frac{p-p+\varepsilon}{1-p+\varepsilon} = \frac{\varepsilon}{1-p+\varepsilon}$.
Now let's explore when the Chernoff bound is better, i.e., it is greater than the bound provided by Reverse Markov's Inequality. Note that for a lower bound, a greater value provides a tighter bound.

$$\begin{aligned} 1 - e^{-2\varepsilon^2 n} &> \frac{\varepsilon}{1-p+\varepsilon} \\ 1 - p + \varepsilon &> \varepsilon + (1-p+\varepsilon)e^{-2\varepsilon^2 n} \\ \frac{1-p}{1-p+\varepsilon} &> e^{-2\varepsilon^2 n} \\ \ln\left(\frac{1-p+\varepsilon}{1-p}\right) &< 2\varepsilon^2 n \\ n &> \frac{1}{2\varepsilon^2}\ln\left(\frac{1-p+\varepsilon}{1-p}\right) \end{aligned}$$

Therefore the Chernoff bound is better when

$$n > \frac{1}{2\varepsilon^2}\ln\left(\frac{1-p+\varepsilon}{1-p}\right)$$

5. (a) Find values a, b such that $232a + 31b = \gcd(232, 31)$ by using the Extended Euclidean Algorithm. Show your work at each step.

Solution: This table provides relevant values when calculating EE(232,31). The bottom row has $\perp$ to indicate undefined values in the base case.

x	y	q	r	g	a	b
232	31	7	15	1	-2	15
31	15	2	1	1	1	-2
15	1	15	0	1	0	1
1	0	$\perp$	$\perp$	1	1	0

Therefore $232(-2) + 31(15) = 1$, so our values of a, b are $(-2, 15)$.

Note that these values are not unique - if (a, b) are numbers s.t $ax + by = \gcd(x, y)$ then so are

$$\left(a + \frac{ky}{\gcd(x,y)}, b - \frac{kx}{\gcd(x,y)}\right)$$

(b) Using your result from (a), find a value a such that $31 \cdot a \equiv 1 \mod 232$.

Solution: Using the coefficients a, b found above, we know that 232(-2) + 31(15) = 1. Then

$$31(15) = 1 + 232(2)$$

$$\iff$$
$$31(15) \mod 232 \equiv 1 + 232(2) \mod 232$$
$$\iff$$
$$31(15) \equiv 1 \mod 232$$

Thus one possible value of a is 15. Of course this means that $15 + 232k$ is a valid inverse, but usually we restrict ourselves to values in the interval $[0, n)$.

E (c) The last two questions demonstrate the concept of *modular inverse*. Consider the set $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$, the set of remainders modulo n. Given an element $a \in \mathbb{Z}_n$, we say that it has an inverse in $\mathbb{Z}_n$ if there exists an element $b \in \mathbb{Z}_n$ such that $a \cdot b = 1$, or equivalently, $ab \equiv 1 \pmod{n}$.

For example, using the Extended Euclidean Algorithm, we showed that in $\mathbb{Z}_{232}$, 31 has the inverse found in (b). However, not all elements $a \in \mathbb{Z}_{232}$ have an inverse in $\mathbb{Z}_{232}$.

Try using the same method used in (a) and (b) to find the modular inverse of 29 in $\mathbb{Z}_{232}$ and explain why this doesn't work. Briefly explain why 29 has no inverse in $\mathbb{Z}_{232}$.

Solution: The Extended Euclidean algorithm can help us find values a, b such that $232a + 29b = \gcd(232, 29)$. However, note that $\gcd(232, 29) = 29$. If we find a and b (which turn out to be 0 and 1, respectively), then this does not help us find a value b such that $29 \cdot b = 1$. Furthermore, note that $29b \pmod{232}$ is always equivalent to a multiple of 29. 1 is not a multiple of 29, so there cannot exist a b that causes this. Therefore, 29 has no inverse in $\mathbb{Z}_{232}$.

(d) Let p be a prime number. Prove that all elements of $\mathbb{Z}_p \setminus \{0\}$ have an inverse in $\mathbb{Z}_p$.
Hint: Recall Fermat's little theorem.

Solution:
By Fermat's little theorem, for an arbitrary non-zero element $a \in \mathbb{Z}_p$, $a^{p-1} \equiv 1 \pmod{p}$. By definition, a^{p-2} is an inverse for a in $\mathbb{Z}_p$. Therefore, all elements of $\mathbb{Z}_p \setminus \{0\}$ have an inverse in $\mathbb{Z}_p$.
Alternate solution:
Because p is prime, its GCD with any non-multiple of p is one. In particular, $\gcd(p, a) = 1$ for every $a \in \{1, \ldots, p-1\}$. Therefore, if we run the extended Euclidean algorithm on a and p, we obtain integers $x, y \in \mathbb{Z}$ such that $ax + py = 1$. It follows that $1 - ax = py$, so p divides $1 - ax$, i.e., $ax \equiv 1 \pmod{p}$. Equivalently, $x = x \bmod p$ (the mod-p congruence class of x) is the inverse of a in $\mathbb{Z}_p$.

Optional bonus questions

For bonus questions, we will not provide any insight during office hours or Piazza, and we do not guarantee anything about the difficulty of these questions. *Only attempt these questions **after** you have finished the rest of the homework.*

1. Recall that RP represents the class of decision problems for which there exist one-sided error Monte Carlo algorithms with no false positives that run in polynomial time (see section 5.3 of Section Notes 9).

   Recall that BPP represents the class of decision problems for which there exists an efficient probabilistic two-sided-error algorithm (see section 4 of Section Notes 10).

   Prove that if $\mathsf{NP} \subseteq \mathsf{BPP}$, then $\mathsf{NP} = \mathsf{RP}$.

   Submit your answer to this question on Gradescope.

Solution: Assume that $\mathsf{NP} \subseteq \mathsf{BPP}$. As discussed in Section Notes 10, we have $\mathsf{RP} \subseteq \mathsf{NP}$ unconditionally, so it suffices to show that $\mathsf{NP} \subseteq \mathsf{RP}$. First note that if $A \leq_p B$ and $B \in \mathsf{RP}$, then $A \in \mathsf{RP}$ (why?). Because of this, to show $\mathsf{NP} \subseteq \mathsf{RP}$ it suffices to prove that some NP-Hard problem is in RP, or in particular that SAT $\in \mathsf{RP}$. Since SAT $\in \mathsf{NP}$ and by assumption $\mathsf{NP} \subseteq \mathsf{BPP}$, there is some BPP algorithm B for SAT. Define

Input: A boolean formula ϕ with variables $x_1, \ldots, x_n$

```
function M(φ)
    for i = 1...n do
        if B(φ_{x_i=0}) accepts then
            φ ← φ_{x_i=0}
        else if B(φ_{x_i=1}) accepts then
            φ ← φ_{x_i=1}
        else
            reject
    if φ is true then accept
    else reject
```

Since B is efficient, M is efficient. As well, since M only accepts ϕ if a valid satisfying assignment of ϕ is produced, if $\phi \notin$ SAT then M will reject ϕ. Assume instead that $\phi \in$ SAT. Consider the probability that x_i is given a correct assignment, assuming $x_1, \ldots, x_{i-1}$ have been correctly assigned. Since ϕ is then still satisfiable at the start of the ith iteration, at least one of $\phi_{x_i=0}$ and $\phi_{x_i=1}$ is also satisfiable. If only $\phi_{x_i=0}$ is satisfiable, then x_i is correctly assigned if and only if $B(\phi_{x_i=0})$ accepts i,e, with probability at least 2/3. If both $\phi_{x_i=0}$ and $\phi_{x_i=1}$ are satisfiable, then

$$\begin{aligned}\Pr[x_i \text{ is correctly assigned}] &= \Pr[B(\phi_{x_i=0}) \text{ accepts}] \\ &\quad + \Pr[B(\phi_{x_i=0}) \text{ rejects and } B(\phi_{x_i=1}) \text{ accepts}] \\ &\geq \frac{1}{3} + 0 = \frac{1}{3}.\end{aligned}$$

Lastly, if only $\phi_{x_i=1}$ is satisfiable, then

$$\begin{aligned}\Pr[x_i \text{ is correctly assigned}] &= \Pr[B(\phi_{x_i=0}) \text{ rejects and } B(\phi_{x_i=1}) \text{ accepts}] \\ &= \Pr[B(\phi_{x_i=0}) \text{ rejects}] \cdot \Pr[B(\phi_{x_i=1}) \text{ accepts}] \\ &\geq \frac{2}{3} \cdot \frac{2}{3} = \frac{4}{9}.\end{aligned}$$

In any case, x_i is correctly assigned with probability at least $1/3$, so the probability that all of the variables are correctly assigned is $(1/3)^n$, hence $M(\phi)$ accepts $\phi \in \text{SAT}$ with probability at least $(1/3)^n$. Then M is an RP algorithm for SAT, so $\text{SAT} \in \mathsf{RP}$. As explained previously, this implies $\mathsf{NP} \subseteq \mathsf{RP}$, so $\mathsf{NP} = \mathsf{RP}$ as desired.