# Solutions to CS 70 Challenge Problems: Countability and Counting

Other worksheets and solutions at https://alextseng.net/teaching/cs70/

Alex Tseng

---

## 1 Finiteness, Countable Infinity, Uncountable Infinity

Classify the following sets as finite, countably infinite, or uncountably infinite. Give a short justification.

(a) Set of all prime numbers.
Countably infinite.
There are infinite prime numbers (if this is not obvious, consider the fact that we can always construct another prime number by multiplying all existing prime numbers and adding 1). All prime numbers are natural numbers. That is, this is subset of $\mathbb{N}$. Therefore, it is countable.

(b) Set of all functions from $\{0, 1\}$ to $\mathbb{N}$.
Countably infinite.
Each function in this set is uniquely defined by a two-tuple of natural numbers. That is, this set of functions has a bijection to $\mathbb{N}^2$. For example, $(x, y)$ means that $f(0) = x, f(1) = y$. We know that $\mathbb{N}^2$ is countably infinite..

(c) Set of all functions from $\mathbb{N}$ to $\{0, 1\}$.
Uncountably infinite.
Every function in this set can be mapped to a unique infinite-length bitstring. The $i$th digit in this bitstring is what $f(i)$ evaluates to. We know that the set of all infinite bitstrings is uncountably infinite, so this set is also uncountably infinite.

(d) Set of all possible colors that can be encoded by standard HTML RGB (Each color is a 3-tuple of values from `00` to `FF`).
Finite.
There are only $256^3$ possible colors that can be encoded in this method.

(e) Set of all possible colors that we can see outside in the natural world.
Finite.
This is a silly question, just for fun. If all colors we can see were on a continuous scale (e.g. like $\mathbb{R}$), then the set would be uncountably infinite. However, because of light quantization and other quantum phenomena, it's actually finite (we are restricting ourselves to the visual section of the EM spectrum).

## 2 Countability

(a) Prove that the Cartesian product of any (finite) number of countable sets is countable.
We proceed by induction on $k$, the number of finite countable sets we take the Cartesian product of. Note that Cartesian products are associative, so it does not matter how we take the product.
Base Case: For $k = 1$, we just have one countable set, which is countable (obviously).
Inductive Hypothesis: Assume that the Cartesian product of $k$ countable sets is countable.
Inductive Step: Consider $k+1$ countable sets. Due to the associativity of taking Cartesian products, the Cartesian product of $k + 1$ countable sets is equivalent to the Cartesian product of $S$ and $T$, where $S$ is the Cartesian product of the first $k$ countable sets, and $T$ is the last countable set. By the Inductive Hypothesis, $S$ is countable. Because $S$ and $T$ are both countable, they both can be mapped uniquely to $\mathbb{N}$. We know that $\mathbb{N} \times \mathbb{N}$ is countable, therefore $S \times T$ is also countable.
Therefore, the Cartesian product of any finite number of countable sets is countable.

(b) Consider a perfectly balanced binary tree of infinite depth. How many leaves are there?
$\aleph_1$. Note that the depth of this tree is countably infinite (we can number off the depth). Thus, the depth has size $\aleph_0$. The number of leaves on a perfectly balanced binary tree is equal to $2^n$ if the depth is $n$. Thus, this tree has exactly $2^{\aleph_0} = \aleph_1$ leaves, making it uncountably infinite (the smallest uncountable infinity, if we assume the continuum hypothesis).
Another way to see this is to encode each leaf as an infinite bitstring. To specify a leaf, we simply traverse the tree from root and downward forever, following the bitstring for directions: 0 for left, 1 for right. Thus, each bitstring is uniquely mapped to a leaf (and vice versa). That is, there is a bijection from leaves to infinite bitstrings, which we know is an uncountably infinite set. Thus, there are also uncountably infinite leaves (specifically $\aleph_1$).

(c) Consider a square with side length 1. Are there more points inside the square than on one side of the square? Formally justify your response.
There are exactly the same number of points. In order to prove two sets have the same cardinality, we simply need to find a bijection. Consider putting the square on a Cartesian plane in Q1 so that one of its corners is at the origin. Then any point inside the square is defined by an ordered pair $(x, y)$, where $x, y \in [0, 1]$. Now we uniquely construct a single point on one of the edges. To do this, we interleave the digits (like shuffling cards). For example, if $x = 0.1234...$ and $y = 0.6789...$, then we construct a new value $z = 0.16273849...$, which must also be in the interval $[0, 1]$. Note that this is a bijection because every point in the square can be mapped to some unique point on an edge, and we can invert this process by un-interleaving the digits to retrieve the constituent points. Thus, the two sets have the same size.

# 3 Counting

(a) How many non-decreasing sequences of $k$ numbers are there if all the numbers are drawn (repetition allowed) from the set $\{1, ..., n\}$? For example, one such sequence is $\{1, 3, 3, 6, 9\}$ if $n = 9$ and $k = 5$.
$\binom{n+k-1}{n-1}$. Note that if we draw any $k$ numbers from our set (with replacement), there is exactly one way to put them in a non-decreasing sequence. Thus, the number of non-decreasing sequences is equal to the number of ways we can draw $k$ numbers from $\{1, ..., n\}$ with replacement. This is equivalent to throwing $k$ balls into $n$ numbered bins and seeing where they fall. This is also equivalent to a stars-and-bars problem, where we have $k$ stars and $n - 1$ bars and want to separate them however we want:
$| * * | * | * | * | | * * * | *$
This is equivalent to having $n + k - 1$ spots and choosing $n - 1$ of them to be separators (the rest are stars). Note that this method only counts how many ways to partition the stars (e.g. 3 stars is 3 stars, no matter what the order).

(b) How many ways are there to put $n$ distinct keys on a keyring?
$\frac{(n-1)!}{2}$ if $n \geq 3$, and 1 otherwise. We select one of the $n$ keys to be the anchor, and place that on the keyring. Having that anchor allows us to think of the ring as a straight line instead of a circle. There are now two distinct ends. Thus, we can place the $n - 1$ remaining keys in any permutation around that anchor key. There are $(n - 1)!$ ways to do this. We also realize that if we flip over the keyring, we may see a different permutation, but they're really the same keyring. Thus, we divide by 2. Note that if we were looking for the number of ways to put $n$ distinct animals on a carousel ride, then we would not divide by 2. The case where we only have 1 or 2 keys a bit of an exception. There is only 1 way for each.

(c) How many ways are there to put $n$ distinct keys on a keyring, where exactly two of those keys cannot be right next to each other? Assume $n \geq 4$
$\frac{(n-3)(n-2)!}{2}$. Again, we select one of the keys to be the anchor. This anchor is not one of the keys that have the extra restriction imposed. As for the rest of the $n - 1$ keys, we are equivalently finding how many ways we can permute them while having two specific keys (let's call them $k$ and $\ell$) not be adjacent. To do this, we find the total number of permutations and subtract the number of permutations where $k$ and $\ell$ are together. There are $(n - 1)!$ total permutations for these $n - 1$ keys around the anchor. Now we find the number of permutations where $k$ and $\ell$ are adjacent. If $k$ and $\ell$ are adjacent, then we can

lump them together into 1 key. Then there would be $n-2$ keys, and thus $(n-2)!$ ways of permuting them. But note that $k$ and $\ell$ may be adjacent in two ways (either one of them could be first). Then there are $2(n-2)!$ ways. Thus, the number of ways to permute the $n-1$ keys so that $k$ and $\ell$ are *not* adjacent is $(n-1)! - 2(n-2)! = (n-3)(n-2)!$. Finally, we divide by 2 as before because a keyring is the same keyring viewed from above or below.

(d) *Challenge* How many ways are there to arrange $n$ distinct elements, where $k$ of those elements can't be adjacent to each other? For example, for $n = 9, k = 3$, this is the number of anagrams of "COMPUTERS", where no two vowels are adjacent. You may assume this is always possible ($k \leq \lceil \frac{n}{2} \rceil$).

$k!\binom{n-k}{k-1}\binom{n-k+1}{k}(n-k)!$.

This can be viewed as a complex stars-and-bars problem. Here, we have $k$ elements that cannot be adjacent. So we can think of it has having $k-1$ bars, and we need to put the rest of the elements ($n-k+1$ stars) around these bars, with the stipulation that no two bars are empty.

We will solve this problem using our concrete example. Our end-goal is to take the vowels of "COMPUTERS", and separate them by consonants in some way: __O__U__E__

Firstly, we fix the order of the vowels. If there are $k$ vowels that cannot be adjacent, then there are $k!$ orderings.

Next, we need to guarantee that these $k$ vowels are separated by something. If there are $k$ vowels, there there are $k-1$ spaces between them. Thus, we need to choose $k-1$ of the consonants (there are $n-k$ of them) to be separating those vowels, and preemptively "throw them into those bins". There are $\binom{n-k}{k-1}$ to choose these consonants. Within these consonants, we can order them however we wish, *but we will not impose an order on them yet.*

Then, we can put the remaining $n-2k+1$ consonants into the mix wherever we want. This problem is basically a problem of $k+1$ bins and $n-2k+1$ balls, or $k$ bars and $n-2k+1$ stars (where order does matter, but we will deal with that later). In this case, there are $\binom{n-k+1}{k}$ ways to partition the consonants.

Now we have selected which consonants are in which bins. Note that this partitioning only tells us how many consonants are in each section. We had put down $k-1$ consonants as preemptive separators (on which we have not imposed an order yet), and we just put down $n-2k+1$ more consonants. Now that the distribution of consonants (how many are in each "bin") is fixed, we can now impose an order on all $n-k$ consonants. Thus, we multiply by $(n-k)!$.

Putting it all together, that's $k!\binom{n-k}{k-1}\binom{n-k+1}{k}(n-k)!$.

Note that our example of "COMPUTERS" worked because there are no repeated letters. If our example had repeated letters, we would have to also divide out the repeats, and our analogy would fall apart.