--- Day 1: Not Quite Lisp ---

Santa was hoping for a white Christmas, but his weather machine's "snow" function is powered by stars, and he's fresh out! To save Christmas, he needs you to collect *fifty stars* by December 25th.

Collect stars by helping Santa solve puzzles. Two puzzles will be made available on each day in the Advent calendar; the second puzzle is unlocked when you complete the first. Each puzzle grants *one star*. Good luck!

Here's an easy puzzle to warm you up.

Santa is trying to deliver presents in a large apartment building, but he can't find the right floor - the directions he got are a little confusing. He starts on the ground floor (floor `0`) and then follows the instructions one character at a time.

An opening parenthesis, `(`, means he should go up one floor, and a closing parenthesis, `)`, means he should go down one floor.

The apartment building is very tall, and the basement is very deep; he will never find the top or bottom floors.

For example:

- `(())` and `()()` both result in floor `0`.
- `(((` and `(()(()(` both result in floor `3`.
- `))(((((` also results in floor `3`.
- `())` and `))(` both result in floor `-1` (the first basement level).
- `)))` and `)())())` both result in floor `-3`.

To *what floor* do the instructions take Santa?

Your puzzle answer was `280`.

--- Part Two ---

Now, given the same instructions, find the *position* of the first character that causes him to enter the basement (floor `-1`). The first character in the instructions has position `1`, the second character has position `2`, and so on.

For example:

- `)` causes him to enter the basement at character position `1`.
- `()())` causes him to enter the basement at character position `5`.

What is the *position* of the character that causes Santa to first enter the basement?

Your puzzle answer was `1797`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 2: I Was Told There Would Be No Math ---

The elves are running low on wrapping paper, and so they need to submit an order for more. They have a list of the dimensions (length `l`, width `w`, and height `h`) of each present, and only want to order exactly as much as they need.

Fortunately, every present is a box (a perfect right rectangular prism), which makes calculating the required wrapping paper for each gift a little easier: find the surface area of the box, which is `2*l*w + 2*w*h + 2*h*l`. The elves also need a little extra paper for each present: the area of the smallest side.

For example:

- A present with dimensions `2x3x4` requires `2*6 + 2*12 + 2*8 = 52` square feet of wrapping paper plus `6` square feet of slack, for a total of `58` square feet.
- A present with dimensions `1x1x10` requires `2*1 + 2*10 + 2*10 = 42` square feet of wrapping paper plus `1` square foot of slack, for a total of `43` square feet.

All numbers in the elves' list are in feet. How many total *square feet of wrapping paper* should they order?

Your puzzle answer was `1588178`.

--- Part Two ---

The elves are also running low on ribbon. Ribbon is all the same width, so they only have to worry about the length they need to order, which they would again like to be exact.

The ribbon required to wrap a present is the shortest distance around its sides, or the smallest perimeter of any one face. Each present also requires a bow made out of ribbon as well; the feet of ribbon required for the perfect bow is equal to the cubic feet of volume of the present. Don't ask how they tie the bow, though; they'll never tell.

For example:

- A present with dimensions `2x3x4` requires `2+2+3+3 = 10` feet of ribbon to wrap the present plus `2*3*4 = 24` feet of ribbon for the bow, for a total of `34` feet.
- A present with dimensions `1x1x10` requires `1+1+1+1 = 4` feet of ribbon to wrap the present plus `1*1*10 = 10` feet of ribbon for the bow, for a total of `14` feet.

How many total *feet of ribbon* should they order?

Your puzzle answer was `3783758`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 3: Perfectly Spherical Houses in a Vacuum ---

Santa is delivering presents to an infinite two-dimensional grid of houses.

He begins by delivering a present to the house at his starting location, and then an elf at the North Pole calls him via radio and tells him where to move next. Moves are always exactly one house to the north (`^`), south (`v`), east (`>`), or west (`<`). After each move, he delivers another present to the house at his new location.

However, the elf back at the north pole has had a little too much eggnog, and so his directions are a little off, and Santa ends up visiting some houses more than once. How many houses receive *at least one present*?

For example:

- `>` delivers presents to `2` houses: one at the starting location, and one to the east.
- `^>v<` delivers presents to `4` houses in a square, including twice to the house at his starting/ending location.
- `^v^v^v^v^v` delivers a bunch of presents to some very lucky children at only `2` houses.

Your puzzle answer was `2081`.

--- Part Two ---

The next year, to speed up the process, Santa creates a robot version of himself, *Robo-Santa*, to deliver presents with him.

Santa and Robo-Santa start at the same location (delivering two presents to the same starting house), then take turns moving based on instructions from the elf, who is eggnoggedly reading from the same script as the previous year.

This year, how many houses receive *at least one present*?

For example:

- `^v` delivers presents to `3` houses, because Santa goes north, and then Robo-Santa goes south.
- `^>v<` now delivers presents to `3` houses, and Santa and Robo-Santa end up back where they started.
- `^v^v^v^v^v` now delivers presents to `11` houses, with Santa going one direction and Robo-Santa going the other.

Your puzzle answer was `2341`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 4: The Ideal Stocking Stuffer ---

Santa needs help mining some AdventCoins (very similar to bitcoins) to use as gifts for all the economically forward-thinking little girls and boys.

To do this, he needs to find MD5 hashes which, in hexadecimal, start with at least *five zeroes*. The input to the MD5 hash is some secret key (your puzzle input, given below) followed by a number in decimal. To mine AdventCoins, you must find Santa the lowest positive number (no leading zeroes: `1`, `2`, `3`, ...) that produces such a hash.

For example:

- If your secret key is `abcdef`, the answer is `609043`, because the MD5 hash of `abcdef609043` starts with five zeroes (`000001dbbfa...`), and it is the lowest such number to do so.
- If your secret key is `pqrstuv`, the lowest number it combines with to make an MD5 hash starting with five zeroes is `1048970`; that is, the MD5 hash of `pqrstuv1048970` looks like `000006136ef...`.

Your puzzle answer was `117946`.

--- Part Two ---

Now find one that starts with *six zeroes*.

Your puzzle answer was `3938038`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

Your puzzle input was `ckczppom`.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 5: Doesn't He Have Intern-Elves For This? ---

Santa needs help figuring out which strings in his text file are naughty or nice.

A *nice string* is one with all of the following properties:

- It contains at least three vowels (`aeiou` only), like `aei`, `xazegov`, or `aeiouaeiouaeiou`.

- It contains at least one letter that appears twice in a row, like `xx`, `abcdde` (`dd`), or `aabbccdd` (`aa`, `bb`, `cc`, or `dd`).
- It does *not* contain the strings `ab`, `cd`, `pq`, or `xy`, even if they are part of one of the other requirements.

For example:

- `ugknbfddgicrmopn` is nice because it has at least three vowels (`u...i...o...`), a double letter (`...dd...`), and none of the disallowed substrings.
- `aaa` is nice because it has at least three vowels and a double letter, even though the letters used by different rules overlap.
- `jchzalrnumimnmhp` is naughty because it has no double letter.
- `haegwjzuvuyypxyu` is naughty because it contains the string `xy`.
- `dvszwmarrgswjxmb` is naughty because it contains only one vowel.

How many strings are nice?

Your puzzle answer was `255`.

--- Part Two ---

Realizing the error of his ways, Santa has switched to a better model of determining whether a string is naughty or nice. None of the old rules apply, as they are all clearly ridiculous.

Now, a nice string is one with all of the following properties:

- It contains a pair of any two letters that appears at least twice in the string without overlapping, like `xyxy` (`xy`) or `aabcdefgaa` (`aa`), but not like `aaa` (`aa`, but it overlaps).
- It contains at least one letter which repeats with exactly one letter between them, like `xyx`, `abcdefeghi` (`efe`), or even `aaa`.

For example:

- `qjhvhtzxzqqjkmpb` is nice because is has a pair that appears twice (`qj`) and a letter that repeats with exactly one letter between them (`zxz`).
- `xxyxx` is nice because it has a pair that appears twice and a letter that repeats with one between, even though the letters used by each rule overlap.
- `uurcxstgmygtbstg` is naughty because it has a pair (`tg`) but no repeat with a single letter between them.
- `ieodomkazucvgmuy` is naughty because it has a repeating letter with one between (`odo`), but no pair that appears twice.

How many strings are nice under these new rules?

Your puzzle answer was `55`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 6: Probably a Fire Hazard ---

Because your neighbors keep defeating you in the holiday house decorating contest year after year, you've decided to deploy one million lights in a 1000x1000 grid.

Furthermore, because you've been especially nice this year, Santa has mailed you instructions on how to display the ideal lighting configuration.

Lights in your grid are numbered from 0 to 999 in each direction; the lights at each corner are at `0,0`, `0,999`, `999,999`, and `999,0`. The instructions include whether to `turn on`, `turn off`, or `toggle` various inclusive ranges given as coordinate pairs. Each coordinate pair represents opposite corners of a rectangle, inclusive; a coordinate pair like `0,0 through 2,2` therefore refers to 9 lights in a 3x3 square. The lights all start turned off.

To defeat your neighbors this year, all you have to do is set up your lights by doing the instructions Santa sent you in order.

For example:

- `turn on 0,0 through 999,999` would turn on (or leave on) every light.
- `toggle 0,0 through 999,0` would toggle the first line of 1000 lights, turning off the ones that were on, and turning on the ones that were off.
- `turn off 499,499 through 500,500` would turn off (or leave off) the middle four lights.

After following the instructions, *how many lights are lit*?

Your puzzle answer was `377891`.

--- Part Two ---

You just finish implementing your winning light pattern when you realize you mistranslated Santa's message from Ancient Nordic Elvish.

The light grid you bought actually has individual brightness controls; each light can have a brightness of zero or more. The lights all start at zero.

The phrase `turn on` actually means that you should increase the brightness of those lights by `1`.

The phrase `turn off` actually means that you should decrease the brightness of those lights by `1`, to a minimum of zero.

The phrase `toggle` actually means that you should increase the brightness of those lights by `2`.

What is the *total brightness* of all lights combined after following Santa's instructions?

For example:

- `turn on 0,0 through 0,0` would increase the total brightness by `1`.
- `toggle 0,0 through 999,999` would increase the total brightness by `2000000`.

Your puzzle answer was `14110788`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 7: Some Assembly Required ---

This year, Santa brought little Bobby Tables a set of wires and bitwise logic gates! Unfortunately, little Bobby is a little under the recommended age range, and he needs help assembling the circuit.

Each wire has an identifier (some lowercase letters) and can carry a 16-bit signal (a number from `0` to `65535`). A signal is provided to each wire by a gate, another wire, or some specific value. Each wire can only get a signal from one source, but can provide its signal to multiple destinations. A gate provides no signal until all of its inputs have a signal.

The included instructions booklet describes how to connect the parts together: `x AND y -> z` means to connect wires `x` and `y` to an AND gate, and then connect its output to wire `z`.

For example:

- `123 -> x` means that the signal `123` is provided to wire `x`.
- `x AND y -> z` means that the bitwise AND of wire `x` and wire `y` is provided to wire `z`.
- `p LSHIFT 2 -> q` means that the value from wire `p` is left-shifted by `2` and then provided to wire `q`.
- `NOT e -> f` means that the bitwise complement of the value from wire `e` is provided to wire `f`.

Other possible gates include `OR` (bitwise OR) and `RSHIFT` (right-shift). If, for some reason, you'd like to *emulate* the circuit instead, almost all programming languages (for example, C, JavaScript, or Python) provide operators for these gates.

For example, here is a simple circuit:

```
123 -> x
456 -> y
x AND y -> d
x OR y -> e
x LSHIFT 2 -> f
y RSHIFT 2 -> g
NOT x -> h
NOT y -> i
```

After it is run, these are the signals on the wires:

```
d: 72
e: 507
f: 492
g: 114
h: 65412
i: 65079
x: 123
y: 456
```

In little Bobby's kit's instructions booklet (provided as your puzzle input), what signal is ultimately provided to *wire* `a`?

Your puzzle answer was `16076`.

--- Part Two ---

Now, take the signal you got on wire `a`, override wire `b` to that signal, and reset the other wires (including wire `a`). What new signal is ultimately provided to wire `a`?

Your puzzle answer was `2797`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 8: Matchsticks ---

Space on the sleigh is limited this year, and so Santa will be bringing his list as a digital copy. He needs to know how much space it will take up when stored.

It is common in many programming languages to provide a way to escape special characters in strings. For example, C, JavaScript, Perl, Python, and even PHP handle special characters in very similar ways.

However, it is important to realize the difference between the number of characters *in the code representation of the string literal* and the number of characters *in the in-memory string itself.*

For example:

- `""` is `2` characters of code (the two double quotes), but the string contains zero characters.
- `"abc"` is `5` characters of code, but `3` characters in the string data.
- `"aaa\"aaa"` is `10` characters of code, but the string itself contains six "a" characters and a single, escaped quote character, for a total of `7` characters in the string data.
- `"\x27"` is `6` characters of code, but the string itself contains just one - an apostrophe (`'`), escaped using hexadecimal notation.

Santa's list is a file that contains many double-quoted string literals, one on each line. The only escape sequences used are `\\` (which represents a single backslash), `\"` (which represents a lone double-quote character), and `\x` plus two hexadecimal characters (which represents a single character with that ASCII code).

Disregarding the whitespace in the file, what is *the number of characters of code for string literals* minus *the number of characters in memory for the values of the strings* in total for the entire file?

For example, given the four strings above, the total number of characters of string code (`2 + 5 + 10 + 6 = 23`) minus the total number of characters in memory for string values (`0 + 3 + 7 + 1 = 11`) is `23 - 11 = 12`.

Your puzzle answer was `1333`.

--- Part Two ---

Now, let's go the other way. In addition to finding the number of characters of code, you should now *encode each code representation as a new string* and find the number of characters of the new encoded representation, including the surrounding double quotes.

For example:

- `""` encodes to `"\"\""`, an increase from `2` characters to `6`.
- `"abc"` encodes to `"\"abc\""`, an increase from `5` characters to `9`.
- `"aaa\"aaa"` encodes to `"\"aaa\\\"aaa\""`, an increase from `10` characters to `16`.
- `"\x27"` encodes to `"\"\\x27\""`, an increase from `6` characters to `11`.

Your task is to find *the total number of characters to represent the newly encoded strings* minus *the number of characters of code in each original string literal.* For example, for the strings above, the total encoded length (`6 + 9 + 16 + 11 =`

42) minus the characters in the original code representation (23, just like in the first part of this puzzle) is `42 - 23 = 19`.

Your puzzle answer was `2046`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 9: All in a Single Night ---

Every year, Santa manages to deliver all of his presents in a single night.

This year, however, he has some new locations to visit; his elves have provided him the distances between every pair of locations. He can start and end at any two (different) locations he wants, but he must visit each location exactly once. What is the *shortest distance* he can travel to achieve this?

For example, given the following distances:

```
London to Dublin = 464
London to Belfast = 518
Dublin to Belfast = 141
```

The possible routes are therefore:

```
Dublin -> London -> Belfast = 982
London -> Dublin -> Belfast = 605
London -> Belfast -> Dublin = 659
Dublin -> Belfast -> London = 659
Belfast -> Dublin -> London = 605
Belfast -> London -> Dublin = 982
```

The shortest of these is `London -> Dublin -> Belfast = 605`, and so the answer is `605` in this example.

What is the distance of the shortest route?

Your puzzle answer was `207`.

--- Part Two ---

The next year, just to show off, Santa decides to take the route with the *longest distance* instead.

He can still start and end at any two (different) locations he wants, and he still must visit each location exactly once.

For example, given the distances above, the longest route would be `982` via (for example) `Dublin -> London -> Belfast`.

What is the distance of the longest route?

Your puzzle answer was `804`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 10: Elves Look, Elves Say ---

Today, the Elves are playing a game called look-and-say. They take turns making sequences by reading aloud the previous sequence and using that reading as the next sequence. For example, `211` is read as "one two, two ones", which becomes `1221` (`1` `2`, `2` `1`s).

Look-and-say sequences are generated iteratively, using the previous value as input for the next step. For each step, take the previous value, and replace each run of digits (like `111`) with the number of digits (`3`) followed by the digit itself (`1`).

For example:

- `1` becomes `11` (`1` copy of digit `1`).
- `11` becomes `21` (`2` copies of digit `1`).
- `21` becomes `1211` (one `2` followed by one `1`).
- `1211` becomes `111221` (one `1`, one `2`, and two `1`s).
- `111221` becomes `312211` (three `1`s, two `2`s, and one `1`).

Starting with the digits in your puzzle input, apply this process 40 times. What is *the length of the result*?

Your puzzle answer was `492982`.

--- Part Two ---

Neat, right? You might also enjoy hearing John Conway talking about this sequence (that's Conway of *Conway's Game of Life* fame).

Now, starting again with the digits in your puzzle input, apply this process *50* times. What is *the length of the new result*?

Your puzzle answer was `6989950`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

Your puzzle input was `1321131112`.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 11: Corporate Policy ---

Santa's previous password expired, and he needs help choosing a new one.

To help him remember his new password after the old one expires, Santa has devised a method of coming up with a password based on the previous one. Corporate policy dictates that passwords must be exactly eight lowercase letters (for security reasons), so he finds his new password by *incrementing* his old password string repeatedly until it is valid.

Incrementing is just like counting with numbers: `xx`, `xy`, `xz`, `ya`, `yb`, and so on. Increase the rightmost letter one step; if it was `z`, it wraps around to `a`, and repeat with the next letter to the left until one doesn't wrap around.

Unfortunately for Santa, a new Security-Elf recently started, and he has imposed some additional password requirements:

- Passwords must include one increasing straight of at least three letters, like `abc`, `bcd`, `cde`, and so on, up to `xyz`. They cannot skip letters; `abd` doesn't count.
- Passwords may not contain the letters `i`, `o`, or `l`, as these letters can be mistaken for other characters and are therefore confusing.
- Passwords must contain at least two different, non-overlapping pairs of letters, like `aa`, `bb`, or `zz`.

For example:

- `hijklmmn` meets the first requirement (because it contains the straight `hij`) but fails the second requirement requirement (because it contains `i` and `l`).
- `abbceffg` meets the third requirement (because it repeats `bb` and `ff`) but fails the first requirement.
- `abbcegjk` fails the third requirement, because it only has one double letter (`bb`).
- The next password after `abcdefgh` is `abcdffaa`.
- The next password after `ghijklmn` is `ghjaabcc`, because you eventually skip all the passwords that start with `ghi...`, since `i` is not allowed.

Given Santa's current password (your puzzle input), what should his *next password* be?

Your puzzle answer was `vzbxxyzz`.

--- Part Two ---

Santa's password expired again. What's the next one?

Your puzzle answer was `vzcaabcc`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

Your puzzle input was `vzbxkghb`.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 12: JSAbacusFramework.io ---

Santa's Accounting-Elves need help balancing the books after a recent order. Unfortunately, their accounting software uses a peculiar storage format. That's where you come in.

They have a JSON document which contains a variety of things: arrays (`[1,2,3]`), objects (`{"a":1, "b":2}`), numbers, and strings. Your first job is to simply find all of the *numbers* throughout the document and add them together.

For example:

- `[1,2,3]` and `{"a":2,"b":4}` both have a sum of `6`.
- `[[[3]]]` and `{"a":{"b":4},"c":-1}` both have a sum of `3`.
- `{"a":[-1,1]}` and `[-1,{"a":1}]` both have a sum of `0`.
- `[]` and `{}` both have a sum of `0`.

You will not encounter any strings containing numbers.

What is the *sum of all numbers* in the document?

Your puzzle answer was `191164`.

The first half of this puzzle is complete! It provides one gold star: *

--- Part Two ---

Uh oh - the Accounting-Elves have realized that they double-counted everything *red*.

Ignore any object (and all of its children) which has any property with the value `"red"`. Do this only for objects (`{...}`), not arrays (`[...]`).

- `[1,2,3]` still has a sum of `6`.
- `[1,{"c":"red","b":2},3]` now has a sum of `4`, because the middle object is ignored.
- `{"d":"red","e":[1,2,3,4],"f":5}` now has a sum of `0`, because the entire structure is ignored.
- `[1,"red",5]` has a sum of `6`, because `"red"` in an array has no effect.

Answer:

Although it hasn't changed, you can still get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 13: Knights of the Dinner Table ---

In years past, the holiday feast with your family hasn't gone so well. Not everyone gets along! This year, you resolve, will be different. You're going to find the *optimal seating arrangement* and avoid all those awkward conversations.

You start by writing up a list of everyone invited and the amount their happiness would increase or decrease if they were to find themselves sitting next to each other person. You have a circular table that will be just big enough to fit everyone comfortably, and so each person will have exactly two neighbors.

For example, suppose you have only four attendees planned, and you calculate their potential happiness as follows:

```
Alice would gain 54 happiness units by sitting next to Bob.
Alice would lose 79 happiness units by sitting next to Carol.
Alice would lose 2 happiness units by sitting next to David.
Bob would gain 83 happiness units by sitting next to Alice.
Bob would lose 7 happiness units by sitting next to Carol.
Bob would lose 63 happiness units by sitting next to David.
Carol would lose 62 happiness units by sitting next to Alice.
Carol would gain 60 happiness units by sitting next to Bob.
Carol would gain 55 happiness units by sitting next to David.
David would gain 46 happiness units by sitting next to Alice.
David would lose 7 happiness units by sitting next to Bob.
David would gain 41 happiness units by sitting next to Carol.
```

Then, if you seat Alice next to David, Alice would lose `2` happiness units (because David talks so much), but David would gain `46` happiness units (because Alice is such a good listener), for a total change of `44`.

If you continue around the table, you could then seat Bob next to Alice (Bob gains `83`, Alice gains `54`). Finally, seat Carol, who sits next to Bob (Carol gains `60`, Bob loses `7`) and David (Carol gains `55`, David gains `41`). The arrangement looks like this:

```
     +41 +46
+55   David    -2
Carol       Alice
+60    Bob    +54
     -7  +83
```

After trying every other seating arrangement in this hypothetical scenario, you find that this one is the most optimal, with a total change in happiness of `330`.

What is the *total change in happiness* for the optimal seating arrangement of the actual guest list?

Your puzzle answer was `709`.

--- Part Two ---

In all the commotion, you realize that you forgot to seat yourself. At this point, you're pretty apathetic toward the whole thing, and your happiness wouldn't really go up or down regardless of who you sit next to. You assume everyone else would be just as ambivalent about sitting next to you, too.

So, add yourself to the list, and give all happiness relationships that involve you a score of `0`.

What is the *total change in happiness* for the optimal seating arrangement that actually includes yourself?

Your puzzle answer was `668`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 14: Reindeer Olympics ---

This year is the Reindeer Olympics! Reindeer can fly at high speeds, but must rest occasionally to recover their energy. Santa would like to know which of his reindeer is fastest, and so he has them race.

Reindeer can only either be *flying* (always at their top speed) or *resting* (not moving at all), and always spend whole seconds in either state.

For example, suppose you have the following Reindeer:

- Comet can fly *14 km/s for 10 seconds*, but then must rest for *127 seconds*.
- Dancer can fly *16 km/s for 11 seconds*, but then must rest for *162 seconds*.

After one second, Comet has gone 14 km, while Dancer has gone 16 km. After ten seconds, Comet has gone 140 km, while Dancer has gone 160 km. On the eleventh second, Comet begins resting (staying at 140 km), and Dancer continues on for a total distance of 176 km. On the 12th second, both reindeer are resting. They continue to rest until the 138th second, when Comet flies for another ten seconds. On the 174th second, Dancer flies for another 11 seconds.

In this example, after the 1000th second, both reindeer are resting, and Comet is in the lead at *`1120`* km (poor Dancer has only gotten `1056` km by that point). So, in this situation, Comet would win (if the race ended at 1000 seconds).

Given the descriptions of each reindeer (in your puzzle input), after exactly `2503` seconds, *what distance has the winning reindeer traveled*?

Your puzzle answer was `2660`.

--- Part Two ---

Seeing how reindeer move in bursts, Santa decides he's not pleased with the old scoring system.

Instead, at the end of each second, he awards one point to the reindeer currently in the lead. (If there are multiple reindeer tied for the lead, they each get one point.) He keeps the traditional 2503 second time limit, of course, as doing otherwise would be entirely ridiculous.

Given the example reindeer from above, after the first second, Dancer is in the lead and gets one point. He stays in the lead until several seconds into Comet's second burst: after the 140th second, Comet pulls into the lead and gets his first point. Of course, since Dancer had been in the lead for the 139 seconds before that, he has accumulated 139 points by the 140th second.

After the 1000th second, Dancer has accumulated *`689`* points, while poor Comet, our old champion, only has `312`. So, with the new scoring system, Dancer would win (if the race ended at 1000 seconds).

Again given the descriptions of each reindeer (in your puzzle input), after exactly `2503` seconds, *how many points does the winning reindeer have*?

Your puzzle answer was `1256`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 15: Science for Hungry People ---

Today, you set out on the task of perfecting your milk-dunking cookie recipe. All you have to do is find the right balance of ingredients.

Your recipe leaves room for exactly `100` teaspoons of ingredients. You make a list of the *remaining ingredients you could use to finish the recipe* (your puzzle input) and their *properties per teaspoon*:

- `capacity` (how well it helps the cookie absorb milk)
- `durability` (how well it keeps the cookie intact when full of milk)
- `flavor` (how tasty it makes the cookie)
- `texture` (how it improves the feel of the cookie)
- `calories` (how many calories it adds to the cookie)

You can only measure ingredients in whole-teaspoon amounts accurately, and you have to be accurate so you can reproduce your results in the future. The *total score* of a cookie can be found by adding up each of the properties (negative totals become `0`) and then multiplying together everything except calories.

For instance, suppose you have these two ingredients:

```
Butterscotch: capacity -1, durability -2, flavor 6, texture 3, calories 8
Cinnamon: capacity 2, durability 3, flavor -2, texture -1, calories 3
```

Then, choosing to use `44` teaspoons of butterscotch and `56` teaspoons of cinnamon (because the amounts of each ingredient must add up to `100`) would result in a cookie with the following properties:

- A `capacity` of `44*-1 + 56*2 = 68`
- A `durability` of `44*-2 + 56*3 = 80`
- A `flavor` of `44*6 + 56*-2 = 152`
- A `texture` of `44*3 + 56*-1 = 76`

Multiplying these together (`68 * 80 * 152 * 76`, ignoring `calories` for now) results in a total score of `62842880`, which happens to be the best score possible given these ingredients. If any properties had produced a negative total, it would have instead become zero, causing the whole score to multiply to zero.

Given the ingredients in your kitchen and their properties, what is the *total score* of the highest-scoring cookie you can make?

Your puzzle answer was `21367368`.

--- Part Two ---

Your cookie recipe becomes wildly popular! Someone asks if you can make another recipe that has exactly `500` calories per cookie (so they can use it as a meal replacement). Keep the rest of your award-winning process the same (100 teaspoons, same ingredients, same scoring system).

For example, given the ingredients above, if you had instead selected `40` teaspoons of butterscotch and `60` teaspoons of cinnamon (which still adds to `100`), the total calorie count would be `40*8 + 60*3 = 500`. The total score would go down, though: only `57600000`, the best you can do in such trying circumstances.

Given the ingredients in your kitchen and their properties, what is the *total score* of the highest-scoring cookie you can make with a calorie total of `500`?

Your puzzle answer was `1766400`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 16: Aunt Sue ---

Your Aunt Sue has given you a wonderful gift, and you'd like to send her a thank you card. However, there's a small problem: she signed it "From, Aunt Sue".

You have 500 Aunts named "Sue".

So, to avoid sending the card to the wrong person, you need to figure out which Aunt Sue (which you conveniently number 1 to 500, for sanity) gave you the gift. You open the present and, as luck would have it, good ol' Aunt Sue got you a My First Crime Scene Analysis Machine! Just what you wanted. Or needed, as the case may be.

The My First Crime Scene Analysis Machine (MFCSAM for short) can detect a few specific compounds in a given sample, as well as how many distinct kinds of those compounds there are. According to the instructions, these are what the MFCSAM can detect:

- `children`, by human DNA age analysis.
- `cats`. It doesn't differentiate individual breeds.
- Several seemingly random breeds of dog: `samoyeds`, `pomeranians`, `akitas`, and `vizslas`.
- `goldfish`. No other kinds of fish.
- `trees`, all in one group.
- `cars`, presumably by exhaust or gasoline or something.
- `perfumes`, which is handy, since many of your Aunts Sue wear a few kinds.

In fact, many of your Aunts Sue have many of these. You put the wrapping from the gift into the MFCSAM. It beeps inquisitively at you a few times and then prints out a message on ticker tape:

```
children: 3
cats: 7
samoyeds: 2
pomeranians: 3
akitas: 0
vizslas: 0
goldfish: 5
trees: 3
cars: 2
perfumes: 1
```

You make a list of the things you can remember about each Aunt Sue. Things missing from your list aren't zero - you simply don't remember the value.

What is the *number* of the Sue that got you the gift?

Your puzzle answer was `40`.

--- Part Two ---

As you're about to send the thank you note, something in the MFCSAM's instructions catches your eye. Apparently, it has an outdated retroencabulator, and so the output from the machine isn't exact values - some of them indicate ranges.

In particular, the `cats` and `trees` readings indicates that there are *greater than* that many (due to the unpredictable nuclear decay of cat dander and tree pollen), while the `pomeranians` and `goldfish` readings indicate that there are *fewer than* that many (due to the modial interaction of magnetoreluctance).

What is the *number* of the real Aunt Sue?

Your puzzle answer was `241`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 17: No Such Thing as Too Much ---

The elves bought too much eggnog again - `150` liters this time. To fit it all into your refrigerator, you'll need to move it into smaller containers. You take an inventory of the capacities of the available containers.

For example, suppose you have containers of size `20`, `15`, `10`, `5`, and `5` liters. If you need to store `25` liters, there are four ways to do it:

- `15` and `10`
- `20` and `5` (the first `5`)
- `20` and `5` (the second `5`)
- `15`, `5`, and `5`

Filling all containers entirely, how many different *combinations of containers* can exactly fit all `150` liters of eggnog?

Your puzzle answer was `1638`.

--- Part Two ---

While playing with all the containers in the kitchen, another load of eggnog arrives! The shipping and receiving department is requesting as many containers as you can spare.

Find the minimum number of containers that can exactly fit all `150` liters of eggnog. *How many different ways* can you fill that number of containers and still hold exactly `150` litres?

In the example above, the minimum number of containers was two. There were three ways to use that many containers, and so the answer there would be `3`.

Your puzzle answer was `17`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 18: Like a GIF For Your Yard ---

After the million lights incident, the fire code has gotten stricter: now, at most ten thousand lights are allowed. You arrange them in a 100x100 grid.

Never one to let you down, Santa again mails you instructions on the ideal lighting configuration. With so few lights, he says, you'll have to resort to *animation*.

Start by setting your lights to the included initial configuration (your puzzle input). A `#` means "on", and a `.` means "off".

Then, animate your grid in steps, where each step decides the next configuration based on the current one. Each light's next state (either on or off) depends on its current state and the current states of the eight lights adjacent to it (including diagonals). Lights on the edge of the grid might have fewer than eight neighbors; the missing ones always count as "off".

For example, in a simplified 6x6 grid, the light marked `A` has the neighbors numbered `1` through `8`, and the light marked `B`, which is on an edge, only has the neighbors marked `1` through `5`:

```
1B5...
234...
......
..123.
..8A4.
..765.
```

The state a light should have next is based on its current state (on or off) plus the *number of neighbors that are on*:

- A light which is *on* stays on when `2` or `3` neighbors are on, and turns off otherwise.
- A light which is *off* turns on if exactly `3` neighbors are on, and stays off otherwise.

All of the lights update simultaneously; they all consider the same current state before moving to the next.

Here's a few steps from an example configuration of another 6x6 grid:

```
Initial state:
.#.#.#
...##.
#....#
..#...
#.#..#
####..

After 1 step:
..##..
..##.#
...##.
......
#.....
#.##..

After 2 steps:
..###.
......
..###.
......
.#....
.#....

After 3 steps:
...#..
......
...#..
..##..
......
......

After 4 steps:
......
......
..##..
..##..
......
......
```

After `4` steps, this example has four lights on.

In your grid of 100x100 lights, given your initial configuration, *how many lights are on after 100 steps*?

Your puzzle answer was `814`.

--- Part Two ---

You flip the instructions over; Santa goes on to point out that this is all just an implementation of Conway's Game of Life. At least, it was, until you notice that something's wrong with the grid of lights you bought: four lights, one in each corner, are *stuck on* and can't be turned off. The example above will actually run like this:

```
Initial state:
##.#.#
...##.
#....#
..#...
#.#..#
####.#

After 1 step:
#.##.#
####.#
...##.
......
#...#.
#.####

After 2 steps:
#..#.#
#....#
.#.##.
...##.
.#..##
##.###

After 3 steps:
#...##
####.#
..##.#
......
##....
####.#

After 4 steps:
#.####
#....#
...#..
.##...
#.....
```

```
#.#..#

After 5 steps:
##.###
.##..#
.##...
.##...
#.#...
##...#
```

After `5` steps, this example now has `17` lights on.

In your grid of 100x100 lights, given your initial configuration, but with the four corners always in the *on* state, *how many lights are on after 100 steps*?

Your puzzle answer was `924`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 19: Medicine for Rudolph ---

Rudolph the Red-Nosed Reindeer is sick! His nose isn't shining very brightly, and he needs medicine.

Red-Nosed Reindeer biology isn't similar to regular reindeer biology; Rudolph is going to need custom-made medicine. Unfortunately, Red-Nosed Reindeer chemistry isn't similar to regular reindeer chemistry, either.

The North Pole is equipped with a Red-Nosed Reindeer nuclear fusion/fission plant, capable of constructing any Red-Nosed Reindeer molecule you need. It works by starting with some input molecule and then doing a series of *replacements*, one per step, until it has the right molecule.

However, the machine has to be calibrated before it can be used. Calibration involves determining the number of molecules that can be generated in one step from a given starting point.

For example, imagine a simpler machine that supports only the following replacements:

```
H => HO
H => OH
O => HH
```

Given the replacements above and starting with `HOH`, the following molecules could be generated:

- `HOOH` (via `H => HO` on the first `H`).
- `HOHO` (via `H => HO` on the second `H`).
- `OHOH` (via `H => OH` on the first `H`).
- `HOOH` (via `H => OH` on the second `H`).
- `HHHH` (via `O => HH`).

So, in the example above, there are `4` *distinct* molecules (not five, because `HOOH` appears twice) after one replacement from `HOH`. Santa's favorite molecule, `HOHOHO`, can become `7` *distinct* molecules (over nine replacements: six from `H`, and three from `O`).

The machine replaces without regard for the surrounding characters. For example, given the string `H2O`, the transition `H => OO` would result in `OO2O`.

Your puzzle input describes all of the possible replacements and, at the bottom, the medicine molecule for which you need to calibrate the machine. *How many distinct molecules can be created* after all the different ways you can do one replacement on the medicine molecule?

Your puzzle answer was `535`.

The first half of this puzzle is complete! It provides one gold star: *

--- Part Two ---

Now that the machine is calibrated, you're ready to begin molecule fabrication.

Molecule fabrication always begins with just a single electron, `e`, and applying replacements one at a time, just like the ones during calibration.

For example, suppose you have the following replacements:

```
e => H
e => O
H => HO
H => OH
O => HH
```

If you'd like to make `HOH`, you start with `e`, and then make the following replacements:

- `e => O` to get `O`
- `O => HH` to get `HH`
- `H => OH` (on the second `H`) to get `HOH`

So, you could make `HOH` after *`3` steps*. Santa's favorite molecule, `HOHOHO`, can be made in *`6` steps*.

How long will it take to make the medicine? Given the available *replacements* and the *medicine molecule* in your puzzle input, what is the *fewest number of steps* to go from `e` to the medicine molecule?

Answer:

Although it hasn't changed, you can still get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 20: Infinite Elves and Infinite Houses ---

To keep the Elves busy, Santa has them deliver some presents by hand, door-to-door. He sends them down a street with infinite houses numbered sequentially: `1`, `2`, `3`, `4`, `5`, and so on.

Each Elf is assigned a number, too, and delivers presents to houses based on that number:

- The first Elf (number `1`) delivers presents to every house: `1`, `2`, `3`, `4`, `5`, ...
- The second Elf (number `2`) delivers presents to every second house: `2`, `4`, `6`, `8`, `10`, ...
- Elf number `3` delivers presents to every third house: `3`, `6`, `9`, `12`, `15`, ...

There are infinitely many Elves, numbered starting with `1`. Each Elf delivers presents equal to *ten times* his or her number at each house.

So, the first nine houses on the street end up like this:

```
House 1 got 10 presents.
House 2 got 30 presents.
House 3 got 40 presents.
House 4 got 70 presents.
House 5 got 60 presents.
House 6 got 120 presents.
House 7 got 80 presents.
House 8 got 150 presents.
House 9 got 130 presents.
```

The first house gets `10` presents: it is visited only by Elf `1`, which delivers `1 * 10 = 10` presents. The fourth house gets `70` presents, because it is visited by Elves `1`, `2`, and `4`, for a total of `10 + 20 + 40 = 70` presents.

What is the *lowest house number* of the house to get at least as many presents as the number in your puzzle input?

Your puzzle answer was `786240`.

--- Part Two ---

The Elves decide they don't want to visit an infinite number of houses. Instead, each Elf will stop after delivering presents to `50` houses. To make up for it, they decide to deliver presents equal to *eleven times* their number at each house.

With these changes, what is the new *lowest house number* of the house to get at least as many presents as the number in your puzzle input?

Your puzzle answer was `831600`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

Your puzzle input was `34000000`.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 21: RPG Simulator 20XX ---

Little Henry Case got a new video game for Christmas. It's an RPG, and he's stuck on a boss. He needs to know what equipment to buy at the shop. He hands you the controller.

In this game, the player (you) and the enemy (the boss) take turns attacking. The player always goes first. Each attack reduces the opponent's hit points by at least `1`. The first character at or below `0` hit points loses.

Damage dealt by an attacker each turn is equal to the attacker's damage score minus the defender's armor score. An attacker always does at least `1` damage. So, if the attacker has a damage score of `8`, and the defender has an armor score of `3`, the defender loses `5` hit points. If the defender had an armor score of `300`, the defender would still lose `1` hit point.

Your damage score and armor score both start at zero. They can be increased by buying items in exchange for gold. You start with no items and have as much gold as you need. Your total damage or armor is equal to the sum of those stats from all of your items. You have *100 hit points*.

Here is what the item shop is selling:

```
Weapons:    Cost  Damage  Armor
Dagger        8     4       0
Shortsword   10     5       0
Warhammer    25     6       0
Longsword    40     7       0
Greataxe     74     8       0

Armor:      Cost  Damage  Armor
Leather      13     0       1
Chainmail    31     0       2
Splintmail   53     0       3
Bandedmail   75     0       4
Platemail   102     0       5

Rings:      Cost  Damage  Armor
```

```
Damage +1     25     1       0
Damage +2     50     2       0
Damage +3    100     3       0
Defense +1    20     0       1
Defense +2    40     0       2
Defense +3    80     0       3
```

You must buy exactly one weapon; no dual-wielding. Armor is optional, but you can't use more than one. You can buy 0-2 rings (at most one for each hand). You must use any items you buy. The shop only has one of each item, so you can't buy, for example, two rings of Damage +3.

For example, suppose you have `8` hit points, `5` damage, and `5` armor, and that the boss has `12` hit points, `7` damage, and `2` armor:

- The player deals `5-2 = 3` damage; the boss goes down to 9 hit points.
- The boss deals `7-5 = 2` damage; the player goes down to 6 hit points.
- The player deals `5-2 = 3` damage; the boss goes down to 6 hit points.
- The boss deals `7-5 = 2` damage; the player goes down to 4 hit points.
- The player deals `5-2 = 3` damage; the boss goes down to 3 hit points.
- The boss deals `7-5 = 2` damage; the player goes down to 2 hit points.
- The player deals `5-2 = 3` damage; the boss goes down to 0 hit points.

In this scenario, the player wins! (Barely.)

You have *100 hit points*. The boss's actual stats are in your puzzle input. What is *the least amount of gold you can spend* and still win the fight?

Your puzzle answer was `121`.

--- Part Two ---

Turns out the shopkeeper is working with the boss, and can persuade you to buy whatever items he wants. The other rules still apply, and he still only has one of each item.

What is the *most* amount of gold you can spend and still *lose* the fight?

Your puzzle answer was `201`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 22: Wizard Simulator 20XX ---

Little Henry Case decides that defeating bosses with swords and stuff is boring. Now he's playing the game with a wizard. Of course, he gets stuck on another boss and needs your help again.

In this version, combat still proceeds with the player and the boss taking alternating turns. The player still goes first. Now, however, you don't get any equipment; instead, you must choose one of your spells to cast. The first character at or below `0` hit points loses.

Since you're a wizard, you don't get to wear armor, and you can't attack normally. However, since you do *magic damage*, your opponent's armor is ignored, and so the boss effectively has zero armor as well. As before, if armor (from a spell, in this case) would reduce damage below `1`, it becomes `1` instead - that is, the boss' attacks always deal at least `1` damage.

On each of your turns, you must select one of your spells to cast. If you cannot afford to cast any spell, you lose. Spells cost *mana*; you start with *500* mana, but have no maximum limit. You must have enough mana to cast a spell, and its cost is immediately deducted when you cast it. Your spells are Magic Missile, Drain, Shield, Poison, and Recharge.

- *Magic Missile* costs `53` mana. It instantly does `4` damage.
- *Drain* costs `73` mana. It instantly does `2` damage and heals you for `2` hit points.
- *Shield* costs `113` mana. It starts an *effect* that lasts for `6` turns. While it is active, your armor is increased by `7`.
- *Poison* costs `173` mana. It starts an *effect* that lasts for `6` turns. At the start of each turn while it is active, it deals the boss `3` damage.
- *Recharge* costs `229` mana. It starts an *effect* that lasts for `5` turns. At the start of each turn while it is active, it gives you `101` new mana.

Effects all work the same way. Effects apply at the start of both the player's turns and the boss' turns. Effects are created with a timer (the number of turns they last); at the start of each turn, after they apply any effect they have, their timer is decreased by one. If this decreases the timer to zero, the effect ends. You cannot cast a spell that would start an effect which is already active. However, effects can be started on the same turn they end.

For example, suppose the player has `10` hit points and `250` mana, and that the boss has `13` hit points and `8` damage:

```
-- Player turn --
- Player has 10 hit points, 0 armor, 250 mana
- Boss has 13 hit points
Player casts Poison.

-- Boss turn --
- Player has 10 hit points, 0 armor, 77 mana
- Boss has 13 hit points
Poison deals 3 damage; its timer is now 5.
Boss attacks for 8 damage.
```

```
-- Player turn --
- Player has 2 hit points, 0 armor, 77 mana
- Boss has 10 hit points
Poison deals 3 damage; its timer is now 4.
Player casts Magic Missile, dealing 4 damage.

-- Boss turn --
- Player has 2 hit points, 0 armor, 24 mana
- Boss has 3 hit points
Poison deals 3 damage. This kills the boss, and the player wins.
```

Now, suppose the same initial conditions, except that the boss has `14` hit points instead:

```
-- Player turn --
- Player has 10 hit points, 0 armor, 250 mana
- Boss has 14 hit points
Player casts Recharge.

-- Boss turn --
- Player has 10 hit points, 0 armor, 21 mana
- Boss has 14 hit points
Recharge provides 101 mana; its timer is now 4.
Boss attacks for 8 damage!

-- Player turn --
- Player has 2 hit points, 0 armor, 122 mana
- Boss has 14 hit points
Recharge provides 101 mana; its timer is now 3.
Player casts Shield, increasing armor by 7.

-- Boss turn --
- Player has 2 hit points, 7 armor, 110 mana
- Boss has 14 hit points
Shield's timer is now 5.
Recharge provides 101 mana; its timer is now 2.
Boss attacks for 8 - 7 = 1 damage!

-- Player turn --
- Player has 1 hit point, 7 armor, 211 mana
- Boss has 14 hit points
Shield's timer is now 4.
Recharge provides 101 mana; its timer is now 1.
Player casts Drain, dealing 2 damage, and healing 2 hit points.

-- Boss turn --
```

```
- Player has 3 hit points, 7 armor, 239 mana
- Boss has 12 hit points
Shield's timer is now 3.
Recharge provides 101 mana; its timer is now 0.
Recharge wears off.
Boss attacks for 8 - 7 = 1 damage!

-- Player turn --
- Player has 2 hit points, 7 armor, 340 mana
- Boss has 12 hit points
Shield's timer is now 2.
Player casts Poison.

-- Boss turn --
- Player has 2 hit points, 7 armor, 167 mana
- Boss has 12 hit points
Shield's timer is now 1.
Poison deals 3 damage; its timer is now 5.
Boss attacks for 8 - 7 = 1 damage!

-- Player turn --
- Player has 1 hit point, 7 armor, 167 mana
- Boss has 9 hit points
Shield's timer is now 0.
Shield wears off, decreasing armor by 7.
Poison deals 3 damage; its timer is now 4.
Player casts Magic Missile, dealing 4 damage.

-- Boss turn --
- Player has 1 hit point, 0 armor, 114 mana
- Boss has 2 hit points
Poison deals 3 damage. This kills the boss, and the player wins.
```

You start with *50 hit points* and *500 mana points*. The boss's actual stats are in your puzzle input. What is the *least amount of mana* you can spend and still win the fight? (Do not include mana recharge effects as "spending" negative mana.)

To begin, get your puzzle input.

Answer:

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 23: Opening the Turing Lock ---

Little Jane Marie just got her very first computer for Christmas from some unknown benefactor. It comes with instructions and an example program, but the computer itself seems to be malfunctioning. She's curious what the program does, and would like you to help her run it.

The manual explains that the computer supports two registers and six instructions (truly, it goes on to remind the reader, a state-of-the-art technology). The registers are named `a` and `b`, can hold any non-negative integer, and begin with a value of `0`. The instructions are as follows:

- `hlf r` sets register `r` to *half* its current value, then continues with the next instruction.
- `tpl r` sets register `r` to *triple* its current value, then continues with the next instruction.
- `inc r` *increments* register `r`, adding `1` to it, then continues with the next instruction.
- `jmp offset` is a *jump*; it continues with the instruction `offset` away *relative to itself*.
- `jie r, offset` is like `jmp`, but only jumps if register `r` is *even* ("jump if even").
- `jio r, offset` is like `jmp`, but only jumps if register `r` is `1` ("jump if *one*", not odd).

All three jump instructions work with an *offset* relative to that instruction. The offset is always written with a prefix `+` or `-` to indicate the direction of the jump (forward or backward, respectively). For example, `jmp +1` would simply continue with the next instruction, while `jmp +0` would continuously jump back to itself forever.

The program exits when it tries to run an instruction beyond the ones defined.

For example, this program sets `a` to `2`, because the `jio` instruction causes it to skip the `tpl` instruction:

```
inc a
jio a, +2
tpl a
inc a
```

What is *the value in register `b`* when the program in your puzzle input is finished executing?

Your puzzle answer was `307`.

--- Part Two ---

The unknown benefactor is *very* thankful for releasi-- er, helping little Jane Marie with her computer. Definitely not to distract you, what is the value in register `b` after the program is finished executing if register `a` starts as `1` instead?

Your puzzle answer was `160`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 24: It Hangs in the Balance ---

It's Christmas Eve, and Santa is loading up the sleigh for this year's deliveries. However, there's one small problem: he can't get the sleigh to balance. If it isn't balanced, he can't defy physics, and nobody gets presents this year.

No pressure.

Santa has provided you a list of the weights of every package he needs to fit on the sleigh. The packages need to be split into *three groups of exactly the same weight*, and every package has to fit. The first group goes in the passenger compartment of the sleigh, and the second and third go in containers on either side. Only when all three groups weigh exactly the same amount will the sleigh be able to fly. Defying physics has rules, you know!

Of course, that's not the only problem. The first group - the one going in the passenger compartment - needs *as few packages as possible* so that Santa has some legroom left over. It doesn't matter how many packages are in either of the other two groups, so long as all of the groups weigh the same.

Furthermore, Santa tells you, if there are multiple ways to arrange the packages such that the fewest possible are in the first group, you need to choose the way where the first group has *the smallest quantum entanglement* to reduce the chance of any "complications". The quantum entanglement of a group of packages is the product of their weights, that is, the value you get when you multiply their weights together. Only consider quantum entanglement if the first group has the fewest possible number of packages in it and all groups weigh the same amount.

For example, suppose you have ten packages with weights `1` through `5` and `7` through `11`. For this situation, some of the unique first groups, their quantum entanglements, and a way to divide the remaining packages are as follows:

```
Group 1;             Group 2; Group 3
11 9       (QE= 99); 10 8 2;  7 5 4 3 1
10 9 1     (QE= 90); 11 7 2;  8 5 4 3
10 8 2     (QE=160); 11 9;    7 5 4 3 1
10 7 3     (QE=210); 11 9;    8 5 4 2 1
10 5 4 1   (QE=200); 11 9;    8 7 3 2
10 5 3 2   (QE=300); 11 9;    8 7 4 1
10 4 3 2 1 (QE=240); 11 9;    8 7 5
9 8 3      (QE=216); 11 7 2;  10 5 4 1
9 7 4      (QE=252); 11 8 1;  10 5 3 2
9 5 4 2    (QE=360); 11 8 1;  10 7 3
```

```
8 7 5      (QE=280); 11 9;    10 4 3 2 1
8 5 4 3    (QE=480); 11 9;    10 7 2 1
7 5 4 3 1  (QE=420); 11 9;    10 8 2
```

Of these, although `10 9 1` has the smallest quantum entanglement (`90`), the configuration with only two packages, `11 9`, in the passenger compartment gives Santa the most legroom and wins. In this situation, the quantum entanglement for the ideal configuration is therefore `99`. Had there been two configurations with only two packages in the first group, the one with the smaller quantum entanglement would be chosen.

What is the *quantum entanglement* of the first group of packages in the ideal configuration?

Your puzzle answer was `10439961859`.

--- Part Two ---

That's weird... the sleigh still isn't balancing.

"Ho ho ho", Santa muses to himself. "I forgot the trunk".

Balance the sleigh again, but this time, separate the packages into *four groups* instead of three. The other constraints still apply.

Given the example packages above, this would be some of the new unique first groups, their quantum entanglements, and one way to divide the remaining packages:

```
11 4    (QE=44); 10 5;   9 3 2 1; 8 7
10 5    (QE=50); 11 4;   9 3 2 1; 8 7
9 5 1   (QE=45); 11 4;   10 3 2;  8 7
9 4 2   (QE=72); 11 3 1; 10 5;    8 7
9 3 2 1 (QE=54); 11 4;   10 5;    8 7
8 7     (QE=56); 11 4;   10 5;    9 3 2 1
```

Of these, there are three arrangements that put the minimum (two) number of packages in the first group: `11 4`, `10 5`, and `8 7`. Of these, `11 4` has the lowest quantum entanglement, and so it is selected.

Now, what is the *quantum entanglement* of the first group of packages in the ideal configuration?

Your puzzle answer was `72050269`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your Advent calendar and try another puzzle.

If you still want to see it, you can get your puzzle input.

You can also [Shareon Twitter Mastodon] this puzzle.

--- Day 25: Let It Snow ---

Merry Christmas! Santa is booting up his weather machine; looks like you might get a white Christmas after all.

The weather machine beeps! On the console of the machine is a copy protection message asking you to enter a code from the instruction manual. Apparently, it refuses to run unless you give it that code. No problem; you'll just look up the code in the--

"Ho ho ho", Santa ponders aloud. "I can't seem to find the manual."

You look up the support number for the manufacturer and give them a call. Good thing, too - that 49th star wasn't going to earn itself.

"Oh, that machine is quite old!", they tell you. "That model went out of support six minutes ago, and we just finished shredding all of the manuals. I bet we can find you the code generation algorithm, though."

After putting you on hold for twenty minutes (your call is *very* important to them, it reminded you repeatedly), they finally find an engineer that remembers how the code system works.

The codes are printed on an infinite sheet of paper, starting in the top-left corner. The codes are filled in by diagonals: starting with the first row with an empty first box, the codes are filled in diagonally up and to the right. This process repeats until the infinite paper is covered. So, the first few codes are filled in in this order:

```
   | 1   2   3   4   5   6
---+---+---+---+---+---+---+
 1 |  1   3   6  10  15  21
 2 |  2   5   9  14  20
 3 |  4   8  13  19
 4 |  7  12  18
 5 | 11  17
 6 | 16
```

For example, the 12th code would be written to row `4`, column `2`; the 15th code would be written to row `1`, column `5`.

The voice on the other end of the phone continues with how the codes are actually generated. The first code is `20151125`. After that, each code is generated by taking the previous one, multiplying it by `252533`, and then keeping the remainder from dividing that value by `33554393`.

So, to find the second code (which ends up in row `2`, column `1`), start with the previous value, `20151125`. Multiply it by `252533` to get `5088824049625`. Then, divide that by `33554393`, which leaves a remainder of `31916031`. That remainder is the second code.

"Oh!", says the voice. "It looks like we missed a scrap from one of the manuals. Let me read it to you." You write down his numbers:

```
   |    1         2         3         4         5         6
---+---------+---------+---------+---------+---------+---------+
 1 | 20151125  18749137  17289845  30943339  10071777  33511524
 2 | 31916031  21629792  16929656   7726640  15514188   4041754
 3 | 16080970   8057251   1601130   7981243  11661866  16474243
 4 | 24592653  32451966  21345942   9380097  10600672  31527494
 5 |    77061  17552253  28094349   6899651   9250759  31663883
 6 | 33071741   6796745  25397450  24659492   1534922  27995004
```

"Now remember", the voice continues, "that's not even all of the first few numbers; for example, you're missing the one at 7,1 that would come before 6,2. But, it should be enough to let your-- oh, it's time for lunch! Bye!" The call disconnects.

Santa looks nervous. Your puzzle input contains the message on the machine's console. *What code do you give the machine?*

To begin, get your puzzle input.

Answer:

You can also [Shareon Twitter Mastodon] this puzzle.