Ryan, these questions are for practice. A 45 minute technical interview might have two or three such questions. Questions marked with ⋆ are (inspired by) actual ones I've gotten. Each question has a main version as well as a followup twist that an interviewer might ask upon satisfactory solution of the main version. It might be good to avoid reading these yourself, instead asking Chloe to draw randomly from these questions during a practice interview.

Questions of the form *determine xyz given abc* stand for *construct a program that determines xyz given abc*. Also implicit is the expectation that you explain what assumptions you make and that you offer an informal complexity analysis for each solution. For example, you may want to remark on your assumptions about the types and probability distribution of input data and on which parameters you regard as asymptotically large.

Tools to have in your toolbox include **DFS-stacks**, **BFS-queues**, **dynamic programming**,

0. **legal parentheses.** ⋆ check whether a given string of `(`s and `)`s is well-formed. —— TWIST: ⋆ what if we permit both round and square brackets? we regard `([])` as ill-formed.
1. **missing numbers.** given a length-$(n-1)$ array L containing distinct integers in the range $R = [0, n)$, determine the element of R not in L. —— TWIST: now L has length $n-2$ and misses two elements of R. determine these missing elements.
2. **palindromic substring.** find a longest contiguous palindromic substring of a given string. —— TWIST: count the length-$\geq k$ contiguous palindromic substrings of a given string and for a given k.
3. **distinct elements.** consider a type that supports a comparator with respect to which it is totally ordered. determine whether a list of elements of this type has all elements distinct. —— TWIST: does your solution use as few comparisons as possible (asymptotically)?
4. **herding cows.** ⋆ a farmer tries to herd musical cows all onto some single hill in a finite set H of hills. each cow starts on some hill in H. the farmer may blare any s in a finite set S of songs; when this happens, all the cows on hill h move to the hill $f(s, h)$. given H, S, f determine whether the farmer may achieve her goal by blaring some sequence of songs. —— TWIST: what if the farmer's goal is to clear a specific hill $h_\star \in H$ of cows?
5. **sparse vectors.** ⋆ a sparse vector is a list of numbers most of which are zero. implement a space-saving sparse vector type along with a dot product. —— TWIST: ⋆ a sparse matrix is defined accordingly. represent and multiply large sparse matrices.
6. **lru cache.** ⋆ write an lru cache that maps small strings (say, urls) to large strings (say, html). that is, implement a data structure that at any time stores $\leq k$ many key-value pairs; that permits querying for whether a key is present and its value is so; and that permits insertion of new key-value pair, with the least-recently inserted-or-queried pair removed if space is needed. —— TWIST: ⋆ show us how you would increase your confidence that your code is correct.
7. **family tree.** ⋆ implement a binary-tree-of-strings type. ancestor trees are of this type. we display ancestor trees in this style: `george(katy(mike,carole),william(charles(elizabeth,philip),diana))`. compute the length of such a display. —— TWIST: compute two people's genetic relation given their ancestor trees. the genetic relation of siblings is 1/2; of cousins is 1/8; of strangers is 0; and so forth. the two trees' nodes name people, and we assume people have distinct and unique names.
8. **knight tours.** ⋆ how many ways can a knight on an otherwise empty chessboard move in exactly n steps from one corner to the opposite corner? —— TWIST: consider a simplified game of chess with three black knights, one white king, and no other pieces. the knights start as close as possible to one corner; the king, to the opposite corner. this game has no rules about draws — no stalemate, no draw by repetition, no draw upon 50 reversible moves. *can black force a checkmate?*
9. **an evil company.** a csv file is a text file representing a 2D grid of entries, with rows delimited by newlines and with elements of a row delimited by commas. this csv file's rows records a person's data on some website. its columns `abcdef` list: `a` the website, the person's `b` username and `c` password on that website, and `d` their email address, `e` phone number, and `f` year-of-birth. *parse the file; do some two rows represent the same person?* use reasonable heuristics to identify the aforementioned columns and to answer the question. —— TWIST: suppose now that some rows have missing entries (represented by the string `UNK`). heuristically impute what missing entries you can.

10. city hall. a new hall is to be built for a city enjoying a square grid of streets. we label the intersections by (x, y) for $0 \leq x, y < 100$. it takes $|x - x'| + |y - y'|$ minutes to walk between intersections at (x, y) and (x', y'). at intersection (x, y) live $p(x, y)$ residents. we wish to place the city hall at an intersection so as to minimize the walking time from residence to hall, averaged uniformly over residents. compute an ideal placement given p. —— TWIST: a rectangular central park with northwest corners (a, b) and southeast corner (A, B). we may not place city hall in the park or on its boundary. what then?