Design questions:

* Find scope. Ask questions
* Go BFS and draw basic end to end shapes on the board
* Read about infrastructure at the company and review white pages
  + Tasks, sharding, caching, etc…
* Practice back of envelope calculations
  + Can X pieces of data fit in memory

Algorithm Questions:

* Tries are common
* Practice coding by hand
  + Takes a different approach
* Big O
  + Don’t use N. Use specific values for units. Be very careful here.
* Be excited
  + Drive. Don’t quit. Pay attention to interviewer.
* Be collaborative and not teaching. Ask interviewer.

Steps:

1. Listen for clues. (sorted…no consecutive values, etc...)
2. Draw an example. Always do this!! Find a good example with edge cases.
3. Brute force or naive solution. Discuss this…don’t code it. State it.
4. Optimize
   1. Bottlenecks
      1. Space vs. runtime
      2. Hash Tables
         1. Precomputing.
         2. Very useful and can be used to solve most problems.
         3. Recursion
            1. Bottom up and not top down.
            2. Know when to see it and when to abandon it.
            3. Think of the call tree
            4. Backtracking -> recursion
   2. Unnecessary work
   3. Duplicate work
5. Walk through
   1. Know the variables and when they change. Don’t code just yet. Walk through your examples.
6. Write beautiful code
   1. Write straight. Start top right corner
   2. Use arrows if needed
   3. Error cases
      1. Boundary checks. Consider time…short cut some of these
   4. Good style
      1. Good variable names. Clean style and loops. Minimal code.
   5. Modularize upfront.
7. Testing
   1. You need to test your code!!
   2. Use smart small test cases, edge cases, abbreviate bigger test cases.
   3. Test your code and not the algorithm
   4. Think before you fix.

Questions for your interviewer:

* Have questions ready
  + Culture
  + Career path
  + Tech / Infra
  + Interviewers experience

Looking for good teammates and being a great engineer matter. Behavioral questions matter.

Read system design chapter