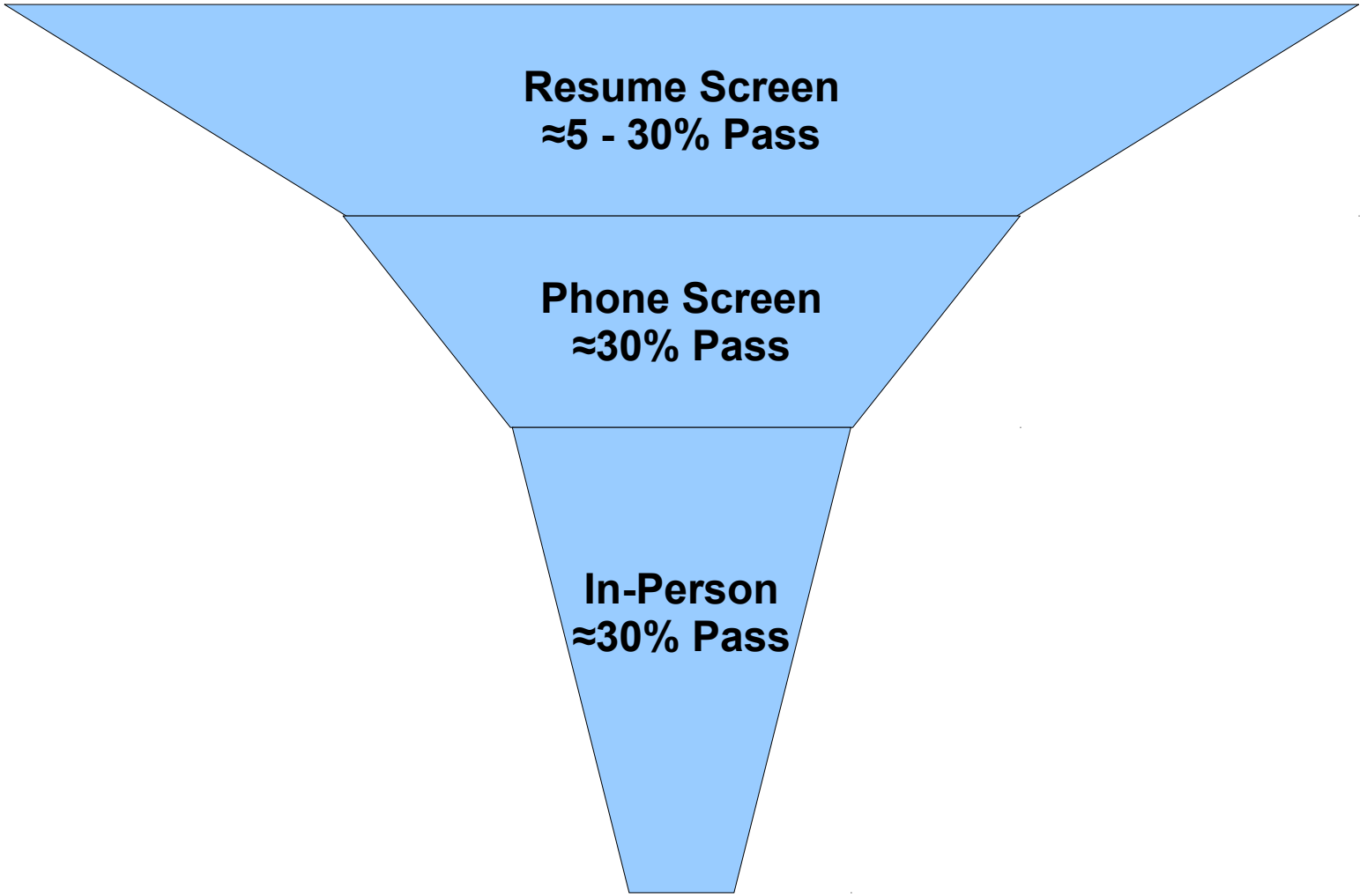
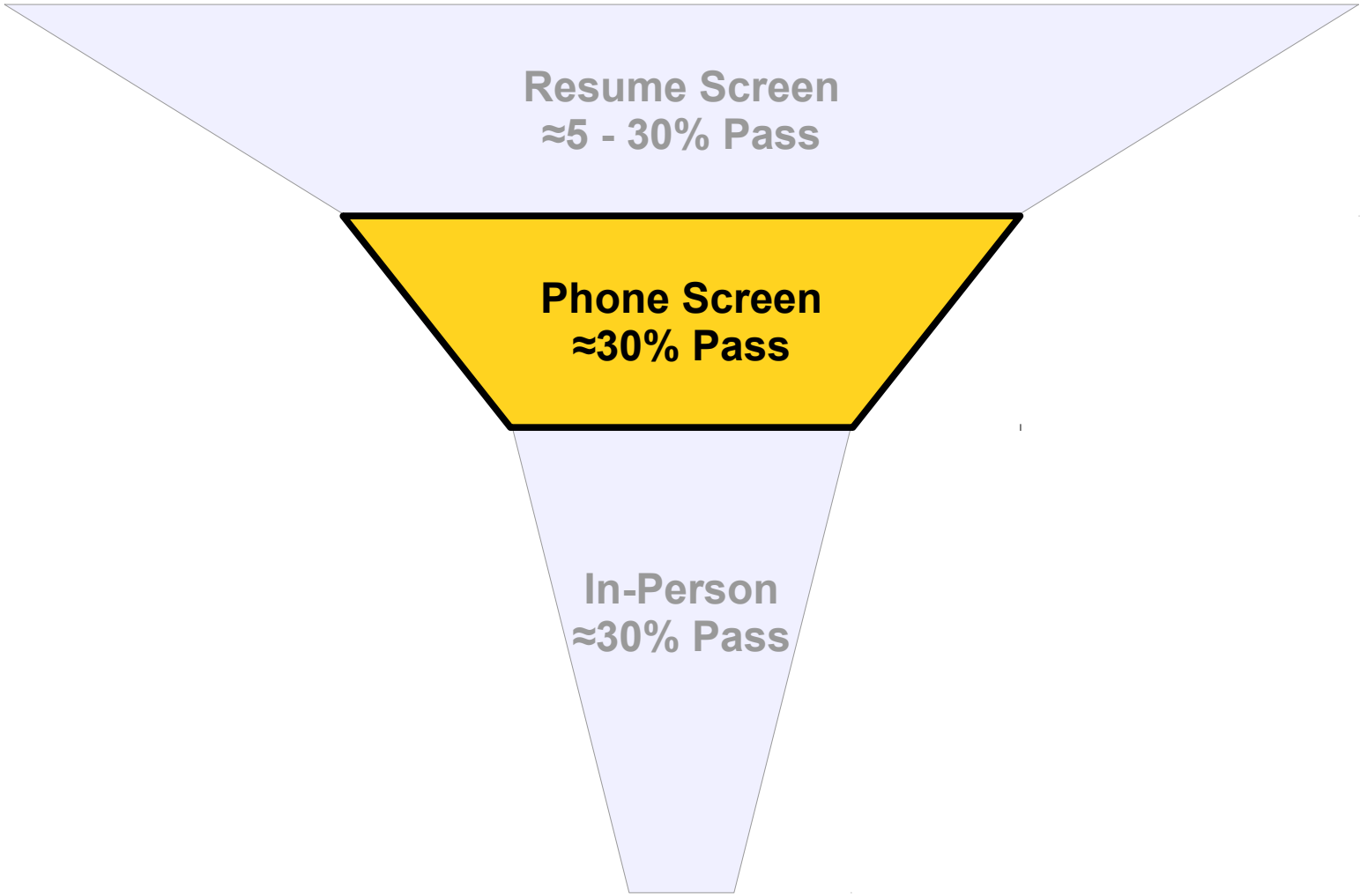
Coding Questions

The Funnel (Large Companies)



The Funnel (Large Companies)



Phone Screens

- Once you're past the resume screen, the company thinks that you may be qualified for the job.
- However, they need to filter out people who look good on paper and aren't qualified.
  - (Or people who are straight-up lying. ☺)
- Most technical phone screens will try to confirm that you have appropriate programming skills. They may also go into more depth.

Phone Screens

- Phone screens will look very different for internships than for full-time jobs.
- For internships, they are often looking for a basic programming competency.
- For full-time jobs, expect to get questions that a CS graduate would be able to answer.

Preparing for a Phone Screen

- Find a good spot for the phone call.
  - Have good cell signal, be away from loud noises, be somewhere where you can work without distractions, etc.
- Have a working network connection.
  - You may be asked to code in an online editor or to type out your thoughts.
- Get a Skype account and set up Google Hangouts.
  - Often times, recruiters will video-chat with you.
  - Look at the camera when you're talking (not your own image).
- Actually prepare for the interview.
  - We'll talk about that in a second. ☺

Technical Questions

- Technical questions can be broadly broken down into a few categories:
  - ***Coding questions***, where the goal is to write code that solves a common problem.
  - ***Algorithm questions***, where the goal is to solve an algorithmic problem, analyze efficiency, and (possibly) implement it.
  - ***Design questions***, where the goal is to design a software system for solving a problem (usually without coding).
  - ***Practicum questions***, where you'll be given some fixed amount of time to design and code something up from scratch on an actual computer.

Coding Questions

- Coding questions commonly focus on a few key areas:
  - Fundamental data structures: arrays, linked lists, binary search trees, hash tables, etc.
  - Fundamental algorithms: binary search, sequence reversal, counting, string search, etc.
  - Client use of data structures: using hash tables, using lists, etc.
  - Simple problem-solving techniques.
  - Object-oriented programming.
  - Regular expressions.

Brushing Up Your Coding

- Short-Term:
  - Briefly refresh yourself on core language syntax (variables, functions/methods, loops, classes, etc.)
  - Read up on the standard libraries. It looks good if you leverage your tools!
  - Consider setting up a Stack Overflow account so that you can ask questions if you have them.
  - Refresh yourself on object-oriented programming: encapsulation, inheritance, etc.
- Longer-Term:
  - For your primary language, pick up a good book on the subject (Effective C++, Effective Java, Dive into Python, Secrets of the JavaScript Ninja, etc.)
  - For your primary language, find a good online reference (cppreference.com, JavaDoc, Python Language Reference, etc.)

Reviewing Big-O Notation

- Most companies will expect that you have a basic familiarity with big-O notation.
- Short-Term:
  - Start reviewing the big-O time complexities of common data structure operations.
  - Know where to go to learn more (bigocheatsheet.com, for example)
- Longer-Term:
  - Have an intuitive understanding of where all the time complexities come from.
  - Become familiar with space complexity.
  - Take CS161.

Algorithms and Data Structures

- Short-Term:
  - Make sure you know about dynamic arrays, binary search trees, hash tables, stacks, queues, priority queues, and linked lists.
  - Make sure you know about binary search, some fast sorting algorithm, DFS, and BFS.
- Longer-Term:
  - Read up Dijkstra's algorithm and Kruskal's or Prim's algorithms.
  - Read up on counting sort and radix sort.
  - Understand how these algorithms work internally and how to use them.

Regular Expressions

- Many tasks these days require the use of regular expressions. If you aren't familiar with them, we strongly recommend spending some time getting acquainted with them.
- Medium-Term:
  - Practice writing regular expressions for common tasks (finding emails, finding phone numbers, etc.)
  - Learn how to use the regular expression support for your programming language of choice.

Today's Exercises

- We've prepared a packet of coding and coding-related problems for you to work on today.
- Feel free to look things up online or punch them into the computer if you'd like, but when possible write code on paper.
- If you're stuck or need help, flag us down!