Consider this your **Onsite Interview Guide**, in addition there are a number of resources compiled below & attached for your review. Remember it's a level playing field at this stage, preparation is key! Your interviews will start around ~10am and last around 5 hours including a 45-min lunch break, so please plan accordingly. You'll get an email from our coordinator soon about the targeted interview date, please reply back and CONFIRM your schedule.

We also offer online prep sessions:
Cracking the Facebook Coding Interview - *The Approach*: https://vimeo.com/157480836
Cracking the Facebook Coding Interview- *Problem Walk-Through*:
https://vimeo.com/158532188
Password: *FB_IPS*

**Interview Overview - (Coding x2, Design, Career + Coding, Lunch)**

**Coding – *2 sessions, 45-min each***

You will have two interviews that focus heavily on coding and will be similar to the initial phone except that these do tend to be a bit more challenging. In addition to code, they'll evaluate you on your thought process; so, be vocal and provide a narrative as you go through the code. You're welcome to code in whatever language you feel most comfortable but choosing one that is going to assist in getting an optimal solution in the most speedy and efficient manner is key.

When your given a problem: 1. Ask clarifying questions; 2. Present multiple solutions if possible; and talk through to the interviewer which solution your choosing and why; 3. Be vocal about how you plan to solve the problem; 4. Then code on whiteboard. These interviews are not just about coding up a solution, but also algorithms. Expect to further optimize your solution and/or expect the interviewer to tweak the problem a bit to test your knowledge to see if you could come up with another solution.

Here are KEY COMPONENTS we look for during coding interviews:

1. Communication
    - Are you asking for requirements/clarity when necessary?
    - Are you listening for hints or advice that interviewer may provide?
    - Are you able to describe your ideas/solutions to problem in such a way that interviewer can understands?
1. Problem-Solving

    - Drive the discussion and provide reasoning as to why you chose any one particular solution
    - Can you come up with solution(s) to problem and compare alternative answers?
    - Are you able to use appropriate data structures?
    - Are you able to speak about space and time complexity?
    - Are you able to optimize your solution even further?
1. Coding
    - Can you convert solutions to executable code?
    - Is the code organized and capture right logical structure?
1. Verification
    - Are you considering a reasonable amount of test cases or come up with good argument for why your code is correct?
    - If you have bugs, are you able to step through your own logic to find them and explain what the code is doing?

Know your stuff - practice coding and make sure to brush up on CS fundamentals especially basic data structures and algorithms. It may also help to review core Computer Science concepts (algorithms, data structures, OOA/D, design patterns etc.) as well as subjects pertaining to working in a large-scale environment.

*Suggested material:*
Introduction to Algorithms – Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (2009)

*Coding practice:*
http://geeksquiz.com
http://www.careercup.com/page
http://www.techinterview.org
https://leetcode.com/

**Lunch**: There will be a 45-minute lunch to break up your day. This session is NOT an interview; this is a great time to ask any questions you have about working at Facebook and enjoy the wonderful food!

**Architecture/ Design** – ***1-2 sessions, 45-min each***

There will be at least one interview that focuses on architecture. These interviews focus on systems – think distributed systems and APIs – very focused on

building/implementing a structure/product. One example of a question: How would you build a chat system that handles millions of concurrently connected users? Be sure to be very thorough in your explanation, we are generally looking for a boxes and arrows diagram on the whiteboard.

You'll also have a design aspect to your interview which will deal with building large scale systems. In 45 minutes, the goal is to be able to design a system, but it will be your approach, how you drive the discussion, and what tradeoffs you make that will determine your ability to think through the design of a complex system.

Some key areas that are reviewed in the design interview feedback:

- Are you asking clarifying questions and driving the discussion
- What is your approach to the design (start high level, then go narrow)
- TRADEOFFS – What tradeoffs are you making and why (support your answer with logical reasoning and/or algorithms)
- Pro/Con analysis of components
- Any bottlenecks/constraints? Storage issues? Etc.

Communication is key, you will be steering the conversation and it will be up to you to understand the problem and ask clarifying questions. Our engineers will be focusing on your familiarity with complex systems and will be looking to see if your able to discuss various solutions and discuss the pro/cons of each. Be sure to think as scale, and identify/discuss any bottlenecks (load balancer, storage, sharding, etc). After gathering requirements, the key is to answer the question by starting high level, then breaking your answer down into sub-components.

Some sample questions are:
– Design a key-value store
– Design Google search
– Architect a world-wide video distribution system
– Build Facebook chat

We may focus on some machine learning, networking system, iOS or JavaScript if you have that domain expertise on your resume. We don't expect you to know crazy algorithms that you likely wouldn’t know off the top of your head (like quad trees or Paxos). The emphasis is on how you perceive the system and problem space.

Some topics you should be familiar with:
– Networking (IPC, TCP/IP)

– Concurrency (threads, deadlock, starvation, consistency, coherence)
– Abstraction (understanding how OS, filesystem, and database works)
– Real--world performance (relative performance RAM, disk, your network, SSD)
– Availability and Reliability (durability, understanding how things can fail)
– Data storage (RAM vs. durable storage, compression, byte sizes)
– CAP Theorem
– byte math

Note that we're not looking for you to be an expert in ALL of these, but you should know enough of them to weigh design considerations and know when to consult an expert.

*For practice...*

Work with a fellow engineer on mock design sessions - take any well-known app and imagine you work for a competitor
Your job is to figure out:
– The general purpose and goals of their system
– What key resources and costs are required (hardware? people? bandwidth? storage?)
– The fundamental bottlenecks and growth hurdles of their system.

Answering those questions will necessarily force you to think about how a system is actually implemented.  Answering only cost and bottlenecks forces you to focus on important areas and not nerdy details of the design. Focus more on the system rather than the nitty-gritty details of the tech.
–  Dig into the implementation and performance of an open source system, understand things like how the system stores data on disk and how it compacts data
– Be familiar with how databases and operating systems work
– Practice on a whiteboard

**Helpful links:*
System design primer
Approaching a Systems Design Question
Introduction to distributed systems design
http://blog.gainlo.co
http://horicky.blogspot.com
https://www.hiredintech.com/classrooms/system-design/lesson/52
http://www.lecloud.net/tagged/scalability
http://tutorials.jenkov.com/software-architecture/index.html
http://highscalability.com/

**Career + Coding** – ***1 session, 45-min each***

This is the interviewer's way to learn about your background/interests; where your passions are in technology, desire for impact, why Facebook, and even possibly things you'd change/improve about Facebook. It always helps to read current news articles/blogs about Facebook for this interview.

Here are additional areas you MAY be asked to speak about:

- Motivation: A project your proud of (focus on your impact)
- Empathy: Talk about a peer you've had a difficult time working with
- Growth: Tell me about a time you received constructive feedback
- Conflict Resolution: Discuss a technical disagreement with a peer and what was end result?

The best way to prepare is to have a few work examples ready to speak about in depth. Keep in mind that we value peer to peer interaction, independent work, and high level of collaboration/partnership. We want engineers to work together to solve problems and we also look for engineers who are willing to take authority any time they identify issues (even outside of their team/responsibilities).

There may be a short coding element in this interview followed by Q&A for any questions you might have about projects, impact, etc.

**Below are additional links to glance through:*
Outline of Interview Process
Facebook Engineering Blog
Get that job at Facebook (useful tips from a Facebook engineer)
Tech talks
Open Source at Facebook
Video about Bootcamp
Newsroom
Interviewed at 5 companies