Microsoft

Preparation Guide –

Unlocking your potential to maximise your performance in your Microsoft interviews

Working Document
Compiled by: Beverly Baker September 2020

Agenda

- Preparing for your interview
- The Interview
- 3 types of questions
- How to approach your interview
- The STAR Framework
- Tips for the interview
- Why choose Microsoft
- Useful Resources

Tips to help you to prepare for your interview

Understand the process going into your interviews

Take time to understand the process and structure of the interviews with Microsoft. Whilst you don't want to sound overly rehearsed, it is a good idea to understand what it entails

- The interview structure for Software Engineers will consist of 4 one on one 45 – 60 minute interviews
- During the interview both Technical Excellence and Core competencies will be covered
- Microsoft takes a mix of an academic, industry and behavioral approach to the interviewing process
- If your interview is virtual, please refer to our career site for best practices in order to have a great interview experience

Practice coding and refresh your knowledge

Prepare for the coding part of the interview well, but don't stress yourself out. Give yourself time to prepare, take breaks and stay active

- Read up on data structures (maps, lists, arrays, stacks, strings, trees, graphs etc.)
- Read up on algorithms – complexity and pseudo code level methods
- You may use pseudo code to break down the problem initially but do let your interviewer know that this is what you are doing
- Practice writing code on a white board and walking a friend through the steps
- Use functions built into your language of choice as much as possible unless otherwise specified
- Work through your code and use examples to validate its correctness. Consider edge cases and how you would test it

Be thoughtful and self reflective

During the interview we will be evaluating Microsoft's core competencies. Be open to sharing your story with us

- Through out our assessment process Microsoft aspires to assess fundamental knowledge, capability and learning potential. You don't need to know everything – Nobody does!
- Be mindful of your career experience, your most memorable achievements, as well as the challenges and obstacles that you have faced
- Think of why these achievements were significant for you and why they were profound. Share your excitement, passion and goals for the future
- Think about what you did to overcome any obstacles or challenges – what drives and motivates you to succeed

The Online Coding Challenge

This is one of the first steps in your interview process at Microsoft. At this stage you will be asked to complete an online assessment through the Codility platform

- We recommend that you try the freely available Codility lessons and read the Codility FAQ before beginning the task
- It's a good idea to familiarise yourself with the Codility tool
- We are looking for well commented code and the use of user created test input
- You should submit as efficient a solution as possible

Areas to prepare direct from our engineers

Technical Preparation

Algorithm Complexity

You need to know Big-O. For more information on Algorithms please click here

Coding

You will be expected to write code in some or all of your interviews. You should be proficient in at least one coding language (preferably C/C++, C#, Java, JavaScript or Python). We strongly recommend you read Programming Interviews Exposed: Secrets to Landing Your Next Job by John Monagan and Noah Suojanen

Sorting

Know how to sort. Don’t do a bubble-sort. You should know the details of at least one n*log(n) sorting algorithm, preferably two (e.g., quicksort; merge sort). Merge sort can be very useful in situations where quicksort is impractical

Hash Tables

Arguably the single most important data structure known to mankind. You absolutely should know how they work. You might want to explore how to implement one using only arrays in your favourite language, in about the space of one interview

Trees

Know about trees - basic tree construction, traversal and manipulation algorithms. Familiarise yourself with binary tree: n-ary trees; trie-trees. Be familiar with at least one type of balanced binary tree, whether it is a red/black tree, a splay tree or an AVL tree. Know how it is implemented. Understand tree traversal algorithms: BFS and DFS, and know the difference between inorder, postorder, and preorder

Graphs

Graphs are important at Microsoft. There are 3 basic ways to represent a graph in memory: objects & pointers, matrix, and adjacency list. Familiarise yourself with each representation and its pros & cons. Know the basic graph traversal algorithms: breadth-first search and depth-first search. Know their computational complexity, their trade-offs, and how to implement them in real code. Try to study up on algorithms, such as Dijkstra or A*

Other Data Structures

You should study up on as many other data structures and algorithms as possible. You should especially know about the most famous classes of NP-complete problems, such as traveling salesman and the knapsack problem, and be able to recognise them when an interviewer asks you them in disguise. Find out what NP-complete means

Mathematics

Interviewers can ask basic discrete math questions. This is more prevalent at Microsoft than at other companies because we are surrounded by counting problems, probability problems, and other Discrete Math 101 situations. Spend some time refreshing your memory on (or teaching yourself) the essentials of combinatorics and probability. You should be familiar with n-choose-k problems and their ilk – the more the better

System Design and Distributed Systems

How not to design Netflix in 45minutes; Anatomy of a System Design interview; A great GitHub repo about system design

These blog-posts summarise the dos and don’ts of a system design interview. We are looking more for best practices around distributed system design. Resiliency, high availability, auto-scaling, replication, CAP theory, partitioning all goes in

Operating Systems

Know about processes, threads and concurrency issues. Know locks, mutexes, semaphores, monitors and how they work. Know deadlock and livelock and how to avoid them. Know what resources a process needs, a thread needs, and how context switching works, how it's initiated by the operating system and underlying hardware. Know about scheduling. The world is rapidly moving towards multi-core, so know the fundamentals of "modern" concurrency constructs

The Interview

What is a technical interview?

- A technical interview is problem-solving based
- You can expect to be tested on algorithms and data structures, coding, systems and design, as well as problem-solving in general
- You will be expected to solve these problems using a white board
- The interview will also include other competency-based questions, as well as questions on your resume

Why does Microsoft interview this way?

- It shows us your technical agility and your ability to think strategically to solve complex problems
- It shows us how you approach problem-solving and your thought process as well as your knowledge on technical principles and methods
- It gives us the opportunity to get to know you better and learn about your experiences in a group dynamic
- It gives us an opportunity to understand how you ask questions and partner to engage with us
- It gives us an overview of the projects you have been working on

How you do in your interview demonstrates certain capabilities

- Your passion for technology
- Your ability to articulate complex ideas
- Your ability to think independently
- Your curiosity and drive
- Your determination to overcome challenges and your adaptability
- Your ability to work in a team
- Leadership
- Confidence
- Your communication skills

Typically there are 3 types of questions

Technical

The technical part of your interview will mostly involve coding questions, building and developing complex algorithms and analysing their performance characteristics, logic problems, and core computer science principles

- Questions to gauge your technical knowledge
- Algorithms and data structures
- Software engineering and coding – in most cases you can choose the
- programming language you want to use
- Systems design
- Problem-solving
- Networking or web focussed
- Questions to help us determine your passion and vision for technology in the future
  - How do you keep your skills up to date and what do you do to learn new things

Behavioural

At Microsoft we're not looking for just great technical skills but also someone who can improve the way we work and be a great team player

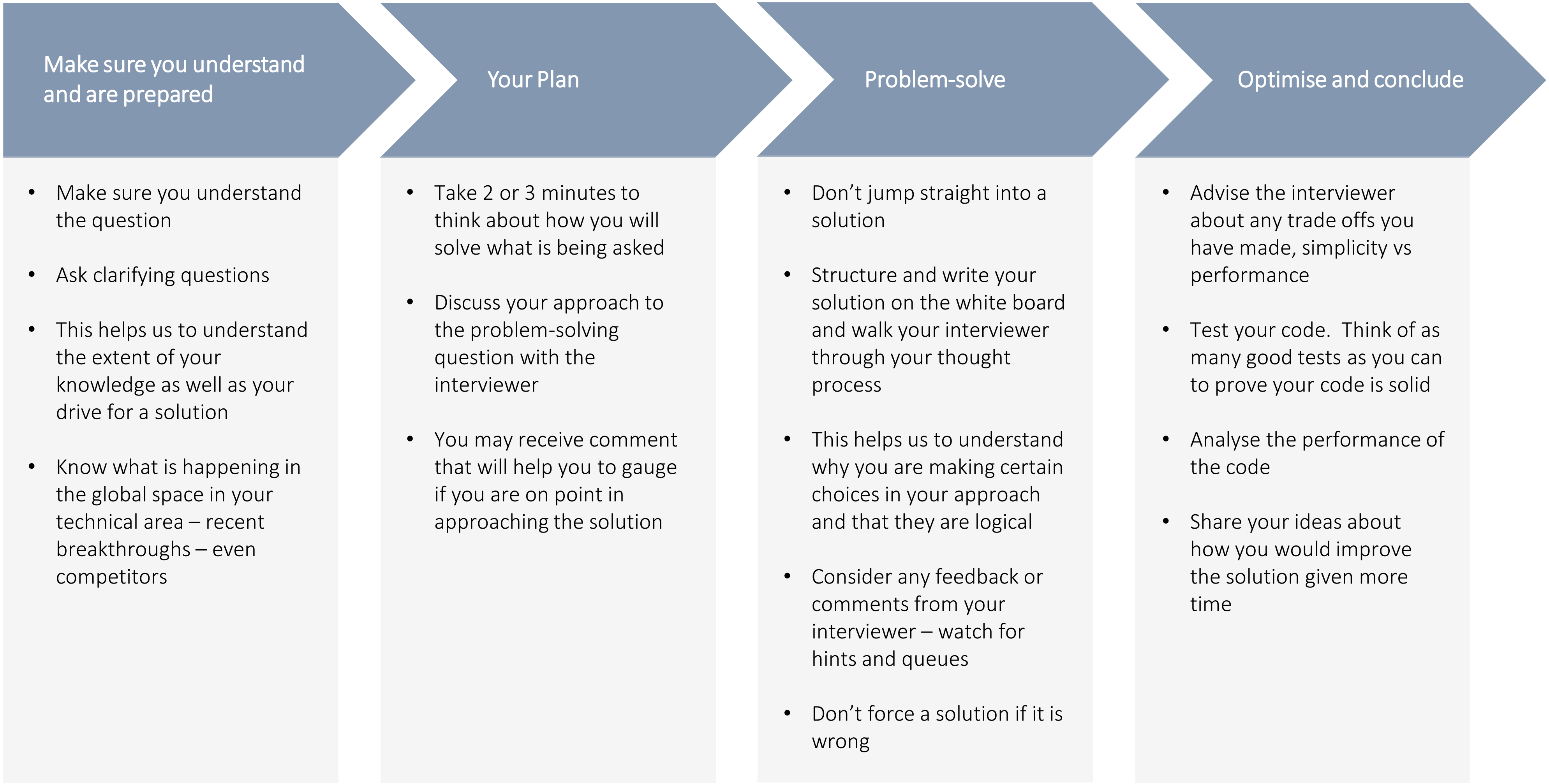
- Questions about your style of working and team dynamic. E.g.,
  - Tell me about a time when you felt something could be improved in a process, how did you go about changing it?
  - Tell me about a time when you felt your idea was better than the one proposed by others, what did you do to convince them?
  - Tell me about a time when you made a mistake, how did you go about rectifying it?
- Questions to help us gauge your growth mindset and interdisciplinary collaboration

Experience

Each of your interviewers will ask you to explain what you currently do at work within the parameters of your confidentiality agreement. It is a good idea to focus on your specific responsibilities, deliverables and the impact you had

- Assume anything on your resume is open for questions
- Be open to discussing your experience
  - Tell me about a piece of work that you have completed recently?
  - Describe a difficult problem that you have recently solved?
- If you contribute to any open source projects, please consider sharing this with the interview panel

How to approach your interview



Make sure you understand and are prepared

- Make sure you understand the question
- Ask clarifying questions
- This helps us to understand the extent of your knowledge as well as your drive for a solution
- Know what is happening in the global space in your technical area – recent breakthroughs – even competitors

Your Plan

- Take 2 or 3 minutes to think about how you will solve what is being asked
- Discuss your approach to the problem-solving question with the interviewer
- You may receive comment that will help you to gauge if you are on point in approaching the solution

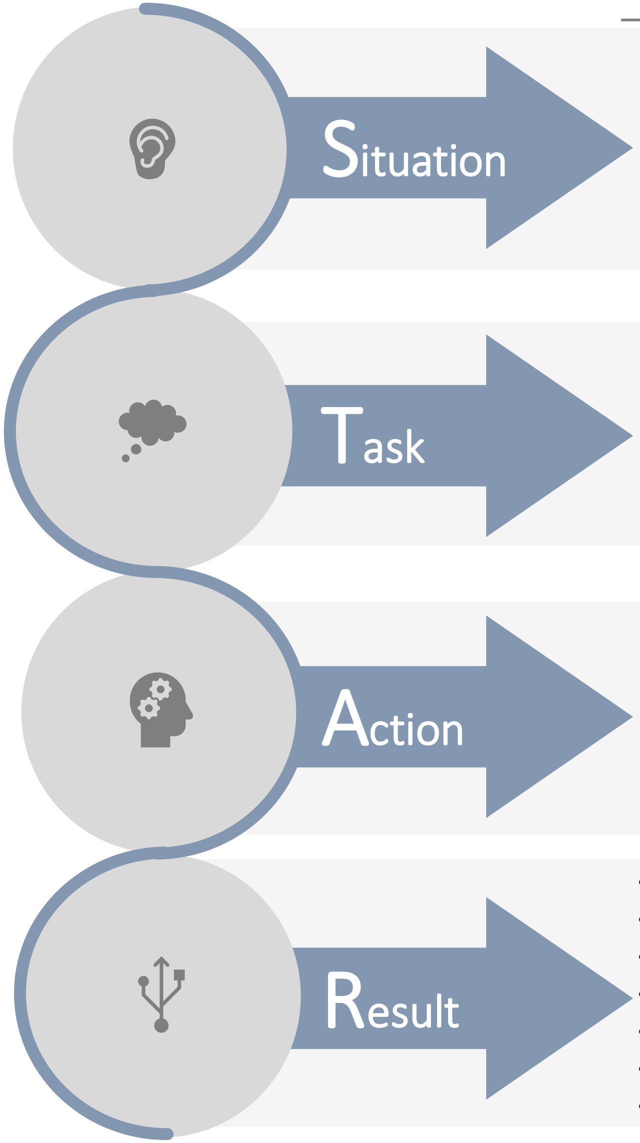
Problem-solve

- Don't jump straight into a solution
- Structure and write your solution on the white board and walk your interviewer through your thought process
- This helps us to understand why you are making certain choices in your approach and that they are logical
- Consider any feedback or comments from your interviewer – watch for hints and queues
- Don't force a solution if it is wrong

Optimise and conclude

- Advise the interviewer about any trade offs you have made, simplicity vs performance
- Test your code. Think of as many good tests as you can to prove your code is solid
- Analyse the performance of the code
- Share your ideas about how you would improve the solution given more time

Structuring your problem-solving – The STAR Framework



	Technical example	Core Competencies example
Situation	• About understanding the problem • Listen to the problem and make sure you understand it • You may want to write down the problem to help you to remember what you are solving for • Ask clarifying questions • Agree on inputs and outputs with your interviewer	Describe and give context to the situation • Use an example from previous work experience to demonstrate its significance • Summarise what was involved in two sentences
Task	• About coming up with a working solution • Run your ideas out loud to your interviewer • Start with a simple case, then proceed to a working solution • Start to come up with different examples to make sure your solution works – keep these for testing later	About the objective, goal or what was required • Briefly explain what you were tasked with • Share your responsibilities, and assignment for the task
Action	• About actual coding and optimising your solution • Start coding in a language that you are comfortable with • Write clean code using clear variable and function names • It is ok to have bugs, but catch them early through testing • Ask before using any libraries • Modularise your algorithm by writing help functions	About the actions you took to resolve the problem • List the actions you took in logical order • Share your thought process and how you reached conclusions
Result	• About testing your code and getting feedback • Testing is best practice • Run through earlier examples (line by line on the board) • Listen for feedback from your interviewer and use it • Talk about how to optimise your code further • Analyse the performance of your code • Talk about trade offs of space and time complexity	About the impact of your actions, your learning and the result • What was accomplished, share evidence to prove success • Share any feedback that you received • Share how the experience impacted your growth, and what and why you learned from it

Tips for your interview

- Be transparent and authentic. If you are stuck, discuss this with your interviewer

- Treat the interview day as a normal day at work, and don't try too hard to impress us

- Take time to structure your thoughts and communication, and walk your interviewer through your thought process step by step – for all the problem-solving tasks
  - Practice writing your code on a white board
  - We are more interested in the approach taken to solve the problem
  - Focus on getting the answer right before you try to optimise

- Consider any comments from your interviewer – she/he may be giving you hints and getting you back on track if you are off point
  - Listen for feedback, don't ignore it and force a solution – adapt your problem-solving
  - Incorporate any actionable feedback into your next interview – it shows you are listening as well as improvement and trajectory
  - If you are asked if you are sure about something, it is probably worth checking your answer
  - Sometimes interviewers may think you are wrong when you are right, other times they may tell you that you are wrong when you are wrong

- Be open to different types of problem-solving – not just coding
  - Sometimes it may be simpler than you think
  - Follow the same steps of structuring, communicating thoughts and walking the interviewer through your thought process

- Think of 1 or 2 thoughtful questions to ask your interviewer

Why choose Microsoft

🎥 'Our people are our best resources . . .' – Satya Nadella

🎥 Microsoft's mission is to empower every person and every organisation on the planet to achieve more. If this resonates with your beliefs, together we can make a difference in our communities and the world. The possibilities are limitless when we're free to be ourselves, trusted to pursue sometimes wild ideas, and surrounded by others who share our passion

At Microsoft you have the chance to:

🎥 Make a difference – our employees have access to the latest technology and tools, the power to build on the company's far-reaching momentum and the drive to change the world

💻 Experience the re-invented company culture - we will only achieve our mission if we live our culture. We start with becoming learners in all things - having a growth mindset. Then we apply that mindset to learning about our customers, being diverse & inclusive, working together as one, and – ultimately - making a difference in the world

🎥 Work in a diverse & inclusive environment – we believe that our continued success depends on the diverse skills, experiences, and backgrounds that our employees bring to the company

💻 Have an amazing range of opportunities - you might become an expert in a particular field or build proficiencies across many areas. You might be an individual contributor or become a manager. Because we have so many kinds of jobs in so many different places, you can stay in one building, city, or country, or you can cross borders - in person, or virtually

🎥 Be part of truly global company - We are a community of individuals, united by a single mission. We work together, building on each other's ideas and collaborating across boundaries to bring the best of Microsoft to our customers and the world

Useful Resources

- https://careers.microsoft.com/us/en/interviewtips
- https://www.hiredintech.com/courses/interview-strategies
- http://www.kegel.com/academy/getting-hired.html
- https://leetcode.com/
- https://www.topcoder.com/
- https://www.hackerrank.com/
- https://www.interviewcake.com/