Interview Preparation 101

Go-to recommendations and best practices to ace a technical job interview. It's by Akvelon Interview Preparation team, for all Akvelon engineers! Questions or recommendations? - please contact [ivanovo.interviewprep@akvelon.com](mailto:ivanovo.interviewprep@akvelon.com)

# Why prepare for an interview?

Firstly, the interview takes about 45-60 minutes in one round (sometimes two), so you are very limited in time and resources to work on all your skills and results. Remember, you only get one chance to do well.

Secondly, at the interview, you will find questions and situations that you may encounter in your normal work and that baffle you - installation with "Your Biggest Mistake" to consider algorithmic tasks without an IDE and IntelliSense and discussing aloud.

Thirdly, candidates often don't discover how they are assessed and what the interviewer pays attention to, so their answers simply do not hit the mark.

We are sure that passing an interview is the same skill as, for example, writing high-quality code that needs to be developed and trained.

But there is also good news - we have a lot of experience in preparing for an interview (general, plus knowledge of interview specifications for specific companies) and we are happy to help :). In the process of training employees, companies will take time to practice answers, find gaps in knowledge, solve potential tasks, and make recommendations, **so we take a serious attitude and give the time allocated for preparation in return**.

# Briefly about the technical interview

A technical interview usually includes the following blocks that we will train.

1. General questions from the category “Tell me about yourself”, and “Your strengths”. This also includes “Your questions for the interviewer” in the conclusion. In addition, in this preparation block, we will rehearse various emergency situations (bad communication, etc.) and give other general recommendations on how to behave in an interview.
2. Technical questions - Questions about specific technologies to confirm qualifications and experience.
3. A practical task for writing or modifying code.

Most likely, when preparing for an interview, some questions may seem strange (for example, “How would your colleagues describe you”), but these are exactly the questions that we often meet in interviews :)

## How is a candidate evaluated?

Of course, it all depends on the specific position (expected seniority) and the culture of the company, but in any company, there is a factor of the incomprehensibility of tasks, so the better the candidate copes with tasks that are incomprehensible, the more valuable the person is. Therefore, the decision is usually made on the basis of an assessment of the following qualities.

1. Seniority (can work independently; can handle open-ended tasks, well-structured big to small answers; ask right questions)
2. Required technical skills/background (has expertise in required technologies or capable of learning quickly)
3. Easy to work with (friendly, good communication, answers are well structured, team player, etc.)
4. Ability and desire to learn new things
5. Passion and energy

Therefore, when preparing and at the interview, try to track how your answers characterize you based on the criteria above.

## General recommendations

1. Do not be afraid of the interviewer, he was once the same developer :)
2. Positive attitude, smile more :)
3. Being ready for a video interview (with the camera on) is a requirement for a lot of companies.
4. Show, ask clarifying questions of interest and be sure of what he says.
5. Address the interviewer by name whenever possible.
6. Provide full information about the company, and product (for example, understand what the result is, its pros and cons, and could be increased or added) before the interview (this will show your interest).
7. 20 minutes before the interview, check readiness: a link to the interview (a client for a conference call may be added), microphone, camera (if necessary), and readiness to share the screen.
8. No extraneous noise (for example, an open window to the street) and people in the room.
9. In this regard, you need to find an interviewer for a long time, say that you were very happy to talk and look forward to the results, for example

“It was great talking to you and thank you very much for your time, looking forward to the results. Thank you." or "Thank you for your time and I would love to join your team" or Thank you for your time, I really enjoy working with <Python> and would love to join your project."

1. Write in a nutshell how the HR interview went immediately after completion. If something happened (bad connection / I was not heard / the interviewer's TV channel accent/interview ended faster than it happened / irrelevant questions) - say hr. This is important to say right away to properly communicate such things with the client.
2. If the connection is bad, report this interview and try to call back.
3. **Practice, practice, practice** - on your own and in our trial interviews. Free practice testing platform in English <https://www.pramp.com/>

## Cheating

Cheating of any kind in the interview is prohibited (help from friends, searching for an answer to a question on the Internet).

If you do not remember the signature of some function, for example, then it is enough to say about it and ask if it is possible to peep on the Internet. This is a fairly standard normal situation.

Other Tips to Avoid Accidentally Being Suspected of Cheating

1. There is no need for any unnecessary rustling, noise in neighboring rooms, cheat sheets and strangers, unnecessary keystrokes, etc.
2. Be ready to turn on the camera (required by some clients for interviews).
3. Put your phone on silent mode
4. It is not recommended to mute the microphone during the interview.

Remember, it’s not at all scary if you don’t know how to solve a problem or answer a specific technical question on the fly; It doesn't mean that you failed the interview :)

# Basic / Personality Questions

In this block, the main thing is the ability to correctly present oneself (see [How is a candidate evaluated?](#_heading=h.3znysh7) above) and the ability to answer uncomfortable questions.

## General recommendations

1. **It is very important to focus on relevant experience** (technologies, languages, similar tasks that are required for this position), therefore:
   1. It is important to know the details of the position (at least technologies, languages, and frameworks; it is better to have examples of future tasks).
   2. The more times the interviewer hears the right technologies in your answers, the better :). At least they should be in “Introduce yourself”.
   3. Think in advance about how you would respond if you were asked to share your experience related to the key requirements of the position. For example, “Please describe your Python experience in more detail”, “Have you worked with ETL/Kafka/Hadoop/etc?”
2. Structure answers well - answer from top to bottom (big picture, then details) and [the STAR method (Situation, Task, Action, Result)](https://www.thebalancecareers.com/what-is-the-star-interview-response-technique-2061629). **Structured responses demonstrate your seniority** and that you will be easy to work with (one of the main qualities along with knowledge of specific technologies).
3. A good way to demonstrate a skill is to tell a short real story/situation from the past (there’s nothing better than a real-life story to engage your interviewer and show what you can do).
4. When answering questions about your experience, focus on what you were doing (responsibilities, technologies) and more specifics on technologies, rather than talking about the functionality of the product for a long time.
5. If there are interesting experiences or strengths, lead the interviewer to discuss these things, for example, “Last X years I was responsible for rebuilding caching infrastructure … or migrating existing architecture to micro-services .. or that was a great experience as I learned a lot about handling tasks end-to-end/cross-team collaboration/customer obsession/proper planning/adding incremental value/etc .. or it was a great time as I had many challenging tasks”. After that, the interviewer will most likely want to know more details**. Purposefully weave things into your answers that show your strengths and what you are good at**.
6. If you are asked if you are familiar with a technology with which you have no experience, instead of a simple “No”, it is better to answer “No, but I always wanted to know” (if you have heard/know about such a technology) or that you have no production experience, but you have experimented yourself (pet project) and understand the basics.
7. Include your technical hobbies (if any): hackathons, coding competitions, pet projects, and contributions to open source projects. This will show that you are a very passionate person and love to code and learn new things.
8. If you have previously worked on a popular application or participated in the development of some cool features, then be sure to tell us about it, for example, “If you used Reddit and you received notifications, then this is exactly what I did.”
9. For Difficult situation/problem/disagreement questions, be prepared to answer 2nd and 3rd follow up questions after an initial high-level answer

## Introduce yourself

95% of interviews begin with this question and it is from this question that the first impression of you is created, so it should be given special attention.

* It is better to aim for 2-3 minutes (the most common mistake is to tell your entire journey as a developer from the very beginning; avoid long tedious narratives in which the interviewer cannot insert a word)
* Don't start with “My name is <your name>” as the interviewer already knows your name and you've probably already exchanged a few phrases before, so My name... sounds rather strange in this context
* Start with general experience and your current specialization
* Share your experience, focusing on technology/skills that are relevant to the position. The rest can simply be listed in the spirit of “I also worked with C#, Java, etc”
* Better group technologies/experiences into areas when speaking, such as Frontend, Backend, Databases, Microservice architecture, Telemetry, logging, monitoring, Auto testing, etc.
* Talk about relevant skills in more detail (frameworks, related technologies), for example, not just ReactJS, but ReactJS with Redux and MobX.
* If you have an interesting project with good relevant experience and you want to shift the conversation in this direction, then instead of talking about your skills/technologies, you can say “My most exciting projects were .. (related to the position)” and tell in a nutshell about projects.
* If appropriate (indeed, it is), in conclusion, you can voice your plans and that this position is in good agreement with them and therefore you are very excited, for example
* I’m really interested in building my Python and Frontend skills further and I think that this position is a great place to apply my current skills and learn further. So I'm very excited about this opportunity :)
* This way you will have a very good ‘present-past-future’ narrative

○ Present - the first 1-2 sentences about you

○ Past - your experience in more detail

○ Future - conclusion

* In the end, you can ask a question about whether it is worth telling something about the previous project.
* Use the following checklist to evaluate your answer.
* As an interviewer, I see and hear the candidate first time does indicate looks like a good match for this job based on his short intro?
* What key terms interviewer extract from the introduction?
* Years of experience front end, back end
* Largest/memorable project
* …
* 1-3 strong/passionate strengths

### Front End example

1. I am a Front End developer

2. My development experience is more than <number> years

3. My primary expertise is building web apps using <React, Redux, AngularJS > (focus on required technologies and languages)

4. I have solid experience using TypeScript and JavaScript

5. On most of my projects, I worked with <REST, GraphQL> services

6. I also have experience working with SQL <MS SQL> and NoSQL <Cassandra> databases

7. I look forward to continuing building my frontend skills and also interested in learning Python so I’m really excited about this opportunity as it aligns really well with my professional goals.

8. Please let me know if you want me to describe some specific expertise in more details or tell more about my previous projects.

### Python Full stack example

1. I am a full stack Python developer with more than 7 years of dev experience.

2. On Front End, I have solid experience with <ReactJS (MobX, Redux, Redux-Saga)> (focus on required technologies and languages). I also can do Angular, Vue and pure JavaScript.

3. I love TypeScript as I find it more productive and more readable in comparison to pure JavaScript.

4. As of Backend my primary languages and frameworks are Python, Flask, Thrift, Protobuf. I also have experience writing backend services using Java in the past. I was mostly responsible for creating REST endpoints. Currently, I’m really interested in getting hands-on experience with GraphQL.

5. I love micro-service and distributed architectures. Have experience working with Docker, Kubernetes, also worked with Kafka and RabbitMQ as a communication mechanism.

6. As for Data layer, I mostly worked with NoSQL databases such as MongoDB and Cassandra. Also, have experience with Memcached and Redis.

7. I’m really excited about this opportunity as I think I can apply my current skills here and learn a lot.

9. Please let me know if you want me to describe some specific expertise in more details or tell more about my previous projects.

### Frontend Lead example

I'm a senior frontend engineer with more than 8 years of experience. And more than a half of that time I worked with React, using Typescript as a main language. Also I did a lot of projects using Redux for data layer.

I love designing and developing modern front-end solutions. For example, previously I worked for Microsoft on AppCenter project where I was part of core UX team and responsible for developing reusable UI components following ARIA guidelines for the rest of the team.

Now I'm working on Reddit.com main app. So if you used new Reddit web app there is a high chance that you are familiar with some of my work :) I'm a part of SEO team working as a tech lead. I’m responsible for architecting and developing new customer faced UI features along with improving some existing backend functionality - for example migrating legacy REST based data layer to GraphQL API resulted in major performance boost. I also help leading and supervising my team: mentoring, planning, technical guidelines, ensure quality, best and modern practises.

I’m really passionate about building modern UI so really excited about this opportunity.

## Previous (current) project

Often there is a follow-up question after Introducing yourself. Recommendations:

* In a nutshell about the project (name, purpose). No need for business project details.
* Talk about architecture and technology **(structured talk about design/architecture, the system automatically adds points to your seniority)**. Focus on the right technologies and languages.
* Tell about your responsibilities and **what exactly you did.** It is good to use the word responsible, for example, “I was responsible for both frontend and backend components on this project”.
* You can talk about the team, for example, “I was part of a distributed team of 6 engineers“.
* You can talk about testing, CI / CD, telemetry, and logging - everything that is interesting and may be useful in the current position.
* You can talk about the process/methodologies/tools (code reviews, source control system, bug tracking system, and continuous integration).
* Use the following checklist to evaluate your answer
  + What specifically candidate did on the project is clear
  + Results are clear
  + Are actions and result correspond to the level of impact of what I expect of a senior engineer?
  + Does candidate understand both business value/why as well as engineering?

### Sample Project 1

1. My last project was about <two words about subject area>

2. It was based on a <microservice architecture running on Kubernetes> ...

3. For API layer we used REST with <Redis, Memcached> for caching and <oauth/jwt> for authentication

4. As database we used <Cassandra>

5. I was responsible for Front End where we used <frameworks, etc.>. My typical tasks were adding new functionality ...

6. For testing we used <Mocha unit tests, Runscope for API, etc>

7. We used <Agile with 2 weeks sprints> as a development methodology, also used pair programming.

### Sample Project 2

1. My last project was about <two words about subject area>

2. It has a <service-oriented, three-layer, etc.> architecture

3. As IOC containers we used <Castle Windsor, StructureMap, Autofac, etc.>

4. Data access layer was implemented using <EF, NHibernate, ADO.NET>

5. Business logic includes such functionality like <payment module, reporting, etc.>

6. Presentation layer was implemented using <3d party controls, frameworks, etc.>

7. The project was tested using automation testing <NUnit, MSTest, Selenium, Jmeter etc.>

8. We use a <SCRUM, Kanban> as a development methodology.

9. I was responsible for implementing new functionality (end-to-end), so I had a good chance to work with all components and layers described above.

Sample Project 3 (Lead level)

I currently work for Reddit as part of SEO team. Our current main goal is to increase user experience for first time visitors but we also constantly looking how to improve regular users experience as well.

For example we recently designed and shipped new survey flow for new users for Reddit.com in which I was deeply involved starting from high level proposal to final integrations and final evaluations like A/B testing.

So majority of my work here is related to developing different front end components and UI using React.

But I also work and help with other areas time to time, for example as I already mentioned we recently did a major rewrite of legacy data layer in Reddit into GraphQL resulted in better performance and more smooth experience for our users. It was a fantastic experience since that work added a lot of customer value and also because I'm mostly working with REST based APIs, so it was kinda new experience to me which is great as I always try to learn and then incorporate new modern practices.

Another my responsibilities here include mentoring, writing specifications, technical guidelines.

## Do you have any questions for me?

<https://www.thebalancecareers.com/job-interview-do-you-have-questions-4138097>

At the end of the interview, most often you will be asked to ask questions you have, here are some examples of what you can ask.

1. Questions related to the product and technologies on it, showing that you have looked at/studied the product.
2. Questions focused on business value team/project creates
3. Questions focused on how the team operates (but not too low level like what are my daily responsibilities). In the more senior role, less hand-holding is expected. The candidate needs to use questions to communicate confidence he’s done that before and can be diligent about how THIS team is doing it. Instead of “do you use scrum” - ask “I’ve used different ways of doing scrum in the past. Curious how is workflow on this team?”
4. Another good strategy is to mention that used product in the past, and ask about a feature/plan if the product allows. E.g., For Airbnb was quite delighted when the candidate mentioned he has been renting his place out as well as staying over in an Airbnb.

Sample Questions

1. If I were hired for this role, what would you want me to achieve in my first months in the position?
2. Can you tell me more about team size and how work is organized?
3. Can you talk about the company culture a bit?

[Senior positions]

1. I’ve used different ways of doing scrum in past. Curious how workflow is organized in your team?
2. How do you test and ensure the code is healthy? I’ve seen many different ways in the past so curious what workflow and type of automated testing do you use?
3. What telemetry information do you collect and what are the key indicators you use to make sure some functionality is successful?
4. How do you prioritize some work against another one?
5. ??? In your opinion, what is the single most important indicator for success in this job?
6. ??? What are you looking for in a candidate - top strengths?

[TPM]

1. What in your opinion what are the most challenging aspects of this role or team?
2. How big is the team what are their backgrounds and experience?
3. What are your short term and long term goals for this team/role?
4. What are the primary skills you are looking for?

**NEVER ask for an impression of yourself or the results of an interview** as this is considered unethical.

## Example answers to other popular questions

### What are your strong and weak sides?

Strengths are things like energy, passion, communication, constantly looking on how to improve existing things, team player, working with ambiguities, cross-team collaboration, creativity, motivating others and so on. When you state each one of these, try to cite a specific example.

In your weaknesses, be sure to minimize them by showing how you’re working on improving them.

### If you have a very tight deadline, how do you make sure that the deadline is met and the work is done?

First of all if there is a risk that I won’t be able to complete my work in time I would communicate this to my manager in advance to see what will be the damage. We might want to select specific work items I should focus on, see if we want to simplify functionality to deliver, for example, implement critical/important functionality only. We can also see if we want to load-balance current work items so that someone else will help with important tasks.

I’m totally fine with working hard and extra hours to meet an important deadline but I think that proper planning is crucial to avoid such situations.

Also, to ensure I meet the deadline I always try to eliminate implementation risks, blockers and unclear points before I start coding and always try to split complex work items on small well defined chunks which leads to accurate estimation and good understanding of what should be done and how.

Plus I always count time for testing, adding tests, deployment, addressing code review notes, etc when planning since development is not just writing code, but adding some incremental value so until your code is merged, deployed, verified and adds value the work is not done.

### How do you mitigate the risk of underestimation?

I always try to split complex work items on small well defined chunks which lead to accurate estimation and good understanding of what should be done and how.

I also always try to eliminate implementation risks, blockers and unclears points before I start coding. For example, if something is very unclear, I would always recommend to perform some quick spike or research before starting actual development.

### How do you deal with ambiguity?

Ambiguity is a part of work and if it affects planning or estimation process I usually do a quick research or spike to clarify it. For example, I can split feature implementation on two phases:

* spend few hours to eliminate all implementation/architecture/other unclear points as part of current sprint and then
* include actual development to the next sprint based on good understanding of how much time it will require.

If I see that there are several ways of how functionality or architecture could be done I write pros and cons for each alternative and then discuss with my colleagues or lead.

### How would you deal with the situation if your colleague does not fit in a timeline (bad performance)?

I would definitely see how I can help them. I’ll start from spending minimal time to understand what happened - bad planning, missing expertise in some particular field, personal/family issue, lack of motivation.

Then based on the situation I’ll see how I can help, for example, discuss the situation with lead, help them myself, re-balance their and my work items, etc.

### Quite often people in one team have different opinions. How do you deal with that? How do you deal with disagreement?

I think this is a very common situation. I’m totally fine if we proceed with another person’s idea as the right decision is more important than who proposed it. I usually list existing alternatives and try to evaluate them based on the criteria and then discuss them with the team. Discussion is a good way to exchange personal experience in the field, maybe someone has extra details or context, discuss different cases and come up with a decision.

### Why do you want to work in <Company name>?

I believe that <Company name> is a very good company to apply my current skills to and to continue growing professionally.

### What is your visa status? (mostly applicable to onsite roles)

If you have an open B1 visa - just say so. If not, your answer should be “I don’t have an open visa now, but I can surely apply for business visa if needed.”

### What is your time zone?

(the answer depends on the client you’re interviewing with - please verify it with HR before the preps)

If this has not been discussed with HR then just tell your time zone, for example “I’m in Moscow time zone which is GMT+3”

### When are you available to start? (below are some suitable options)

“This should be clarified with my management but my understanding is that I can start <>“

* ASAP
* In 2 weeks
* I can start offshore ASAP (or in 2 weeks) and onsite as soon as I get the visa, which is normally around 6-7 weeks

## Other popular questions

1. How are you?
2. Is it just both of us on the call?
3. Do you have any questions upfront?
4. What is your visa status?
5. What is your time zone?
6. When are you available to start?
7. Tell me about your current / previous project. When did you start working on the project, on what stage of it? What were your responsibilities (or role) on the project? What was it that you particularly did?
8. Describe your typical work day.
9. What was the most difficult part or task on the project? What is your recent technical challenge you experienced and how did you solve it? What was the hardest bug you've faced?
10. What did you like / dislike about the project?
11. What was the most interesting thing you did on the project?
12. What bored you the most on your last projects?
13. How would your colleagues describe you?
14. What are your strong and weak sides?
15. What do you like in programming?
16. What does clean/quality code mean to you?
17. Do you use Agile methodologies? What meetings do you have during the week? How long is your sprint?
18. How do you make sure the deadline is met? If you have a very tight deadline, how do you make sure you meet it?
19. Tell me about a time when you were unable to meet a project deadline. How did you react?
20. How do you deal with ambiguity?
21. How do you mitigate risk of underestimation?
22. What would be your actions if you had to start a project from scratch?
23. What is your primary programming language
24. Estimate your JavaScript / Python / etc. skills on the scale from 1 to 10. What frameworks do you have experience with?
25. What other languages do you consider your strengths? (your answer should have both technologies and languages + make sure to mention the ones required for the position)
26. How do you feel about diving in new technologies and learning things as you go?
27. If you could master one technology this year, what would it be?
28. What is the biggest failure in your career? What can you do to avoid repeating it?
29. Do you have any experience in pair programming?
30. Is that ok for you to check your email and incoming messages in the evening?
31. Have you ever worked with foreign colleagues?
32. Is that ok for you to have late meetings, saying 9PM your time or so?
33. What do you like most in remote work? What things you don’t like while working remotely?
34. What devices do you have at your disposal? (Test devices such as latest iPhone and Android you can use for development and testing; Akvelon provides such devices so your answer should be “We have different devices available for test and dev so I don’t think there will be any problem”)
35. What are your motivations in working with Reddit / Microsoft / OpenTable?
36. Name your favorite tech company. Why are they your favorite?
37. What is the proudest achievement on your resume?
38. How do you keep up-to-date on new technologies and trends in this industry?
39. At <Company> we seek to hire leaders. Tell me about a time when you took the initiative to improve work related procedures.
40. We consider research to be the backbone of what we do at <Company>. Do you think it's important to conduct research before beginning a project or would you rather jump right in?
41. At <Company> we take privacy and confidentiality very seriously. Are you willing to sign a non-disclosure agreement, if hired?
42. In your opinion, what has been the biggest advancement in technology this past year?
43. Quite often people in one team have different opinions. How do you deal with that? How do you deal with disagreement?
44. Have you worked with someone who is falling behind in the project? What would you do with this type of person?
45. [TPM] What is your workflow with the team? What would be your interaction with them and your interaction with me?
46. In your opinion, what is the most exciting startup at the moment?

## Bonus / Extra materials

[16 Things to Say in a Job Interview](https://www.thebalancecareers.com/what-to-say-in-a-job-interview-4158527)

[How to Approach Behavioral Questions (video from HackerRank)](https://www.youtube.com/watch?time_continue=0&v=tZxNNKqxXnw)

[How to Answer the Most Frequently Asked Interview Questions](https://www.thebalancecareers.com/job-interview-questions-and-answers-2061204)

# Technical Questions

In this block, you need to answer questions about the technologies that are stated in your resume or are required for the project in order for you to confirm your expertise and determine how deep you know this subject area.

Recommendations

1. Browse the Internet for “Top <Some-Technology> interview questions” on the key technologies required for the position
2. Rehearse answers out loud in English
3. When answering technical questions, if possible, tell where and how you used it, for example
4. “ReactJS Hooks, yes, I used them for a project a month ago” (or I actively use them on the current project)

An example of a well-structured answer to a technical question.

What SQL replication types do you know?

1. Explain types
2. Explain use cases for each type - when best to use
3. Give example where you used a SQL replication, why selected particular type
4. Ask if interviewer wants to go into more details?

# Coding Task

## Recommendations on how to behave and solve the tasks

[Tips and Guidelines from HackerRank](https://www.hackerrank.com/interview/interview-preparation-kit/tips-and-guidelines/videos)

[How to Crack a Google Coding Interview - An Ex-Googler’s Guide](https://www.youtube.com/watch?v=uQdy914JRKQ)

Basic moments:

1. Ask questions. In any company, there is a factor of the incomprehensibility of tasks, and the better the candidate copes with such tasks (clarifies details, asks the right questions), the more valuable the person is. The worst thing that a candidate usually does is immediately rush to write code without any discussion and without finding out whether he correctly understood the task and how the task merges into a higher-level part - what are the restrictions and so on (memory, runtime, etc). It is important to ask clarifying questions about what the interviewer wants. This makes it clear the level of the candidate - a senior developer cannot start solving problems without clarification. It's good to start like this - "Have I understood you correctly that..?" Then clarify your assumptions and ask questions.
2. Function signature. Before writing code, you should propose a function signature (a name that accepts and returns) and agree with the interviewer
3. Sample data. It’s good if you immediately offer and write down several cases for testing. Be sure to test for null or invalid data. You can just write all the input options at once that you would like to test and what you would expect to receive, then go through them and check, for example:

ReverseString(null) -> exception

ReverseString("") -> ""

ReverseString("a") -> "a"

ReverseString("ccFvzzz-123") -> "321 -zzzvFcc"

ReverseString(someVeryLongString) ->

When it comes to how to handle invalid values ​​or empty strings, it's easiest to focus on how it is customary in an existing application/library or how such cases are handled by standard classes/libraries in your language.

1. Brute force solution. You should not immediately rush and write the most optimal, but difficult to implement and understand the algorithm. It is better to say that there are several alternatives, compare their pros and cons, and offer to implement the easiest one to implement and further support. No need to do premature optimization.
2. Do not be silent while writing the code, if you are thinking about something, then voice it.
3. The code is written real, with comments and mandatory checking of input arguments (in order not to waste time, you can add a comment // TODO args check).
4. Take Your Time and Pay Due Attention to Variable Names and Style
5. Test solution. Be sure to check how your code works on test data
6. Optimize the solution. Tell us how the solution could be improved (complexity, thread safety, etc) or refactored.
7. Be prepared to evaluate the algorithm for speed and memory using the “Big O” notation, such as the O(N) complexity of a linear algorithm.
8. Most often, coding happens in a regular text editor or online service without any support for IntelliSense and other things, so it is highly recommended to practice writing code on your own in a standard text editor.
9. Sometimes the candidate is asked to open their favorite IDE and write code in it. Think in advance about what type of application you will use for this (for example, console / Console app).
10. Situation: the candidate was given a task, but he knows the solution - it is better to pronounce this solution (rather than write it right away - the client may suspect why he wrote it so quickly, does he know the answer?) Or even say that the task is already known.
11. Situation: a problem with solving a problem - you can defuse the situation by saying something along the lines of “I’m so nervous and dumb about something, I always solve such problems, but something doesn’t work here :)”

## Mock Coding Interview and how to evaluate it

In practice, when assessing the level of a candidate, a big role is played not only by whether you actually solve the problem or not but by how much you can show your seniority and also how good problem-solving skills you have (and can show it).

An excellent option to significantly improve your skills in this direction would be to conduct training coding sessions.

Things to think about/pay attention to before the coding session:

1. Do you have an understanding of what people pay attention to when solving a problem, and how, and by what criteria your code is evaluated?
2. Conduct several mock interviews and “consciously” apply the criteria from paragraph 1 when performing a coding task, mentally give yourself points in terms of how much you adhere to the advice from paragraph 1 (ideally, if someone else evaluates you, for example, Your colleague, who can then share his feedback with you)
3. Remember that your main goal is to show your seniority when solving a problem since the client has an expectation that you will write and reason on the code while working in the same way as you do now when performing a coding task.

Mock interviews are conducted not so much to improve problem-solving skills (for this you can train individually, on leetcode/codewars/hr, etc), but to know what they pay attention to when conducting an interview and how you can present your skills in a favorable light.

## Practical tasks

1. Solve Easy and Medium tasks from HackerRank's selection of the most popular types of interview problems (strings, arrays, etc.): https://www.hackerrank.com/interview/interview-preparation-kit
2. Additional materials

| [CodeWars Akvelon Warm Up Preset](https://www.codewars.com/collections/akvelon-basic-test) | Akvelon internal set of most popular coding tasks/algos/structures |
| --- | --- |
| Python Intro to Data Structures and Algorithms  <https://www.udacity.com/course/data-structures-and-algorithms-in-python--ud513> | Technical interviews follow a pattern. If you know the pattern, you’ll be a step ahead of the competition. This course will introduce you to common data structures and algorithms in Python. You'll review frequently-asked technical interview questions and learn how to structure your responses.  You will answer practice problems and quizzes to test your abilities. Then you'll practice mock interviews to get specific recommendations for improvement. Be ready for anything the technical interviewer throws at you. |