Recommended Resources: Python Development

On this page, you'll find a few tips on how to prepare for an interview with the Python Development team at Qminers. Our work is diverse: some team members lean more toward data analytics, others toward DevOps infrastructure, and others focus on application or computational code. What's essential for one may be irrelevant for another.

But we all have a few things in common – we write excellent Python, communicate clearly with analysts, and think several steps ahead.

Our interview structure reflects this. We'll talk about Python, tackle an algorithmic problem together, work through some logical puzzles, and possibly explore a bit of mathematical reasoning. The goal is to understand what you're good at and where your boundaries lie. We don't expect you to know everything – but we do hope you'll demonstrate broad awareness and real strength in at least one area.

Want to prepare? Here are a few resources that may come in handy.

Jakub Tesárek
Python Developer, Team Leader

1 Fluent Python by Luciana Ramalho

Python is our daily language of choice, and ***Fluent Python*** is an excellent resource for going beyond the basics.

We recommend reading it from cover to cover — but if it feels too long, focus on the following chapters, which are especially relevant for us:

II. Data Structures
2. An Array of Sequences
3. Dictionaries and Sets

III. Functions as Objects
5. First-Class Functions
7. Function Decorators and Closures

IV. Object-Oriented Idioms
8. Object References, Mutability, and Recycling
9. A Pythonic Object
10. Sequence Hacking, Hashing, and Slicing

V. Control Flow
14. Iterables, Iterators, and Generators
15. Context Managers and else Blocks
16. Coroutines *
18. Concurrency with asyncio *

2 Pandas and Numpy

We use ***Pandas*** and ***Numpy*** for data processing on a daily basis. While we won't test them directly during the interview, knowing the basics will give you a head start.

3 100 days of algorithms by Tomáš Bouda

Understanding algorithms and being able to apply them to real-world problems is fundamental to our work. We don't ask you to write a red-black tree balancer you'll never use again – we use algorithms all the time. One of the best ways to get up to speed is ***100 Days of Algorithms*** by Tomáš Bouda, who previously led our PyDev team.

4 Průvodce labyrintem algoritmů

For a more structured and academic resource, our colleague Viki recommends ***Guide Through the Algorithmic Labyrinth*** by Martin Mareš and Tomáš Valla. (Czech only.)

5 Project Euler / LeetCode

If you want to practice algorithmic problems hands-on, ***Project Euler*** is highly recommended – many of the challenges mirror the kind of tasks we solve, especially when working with analysts on mathematical models. ***LeetCode*** is also useful, especially if you're new to algorithmic problem-solving or just want to brush up on it.

6 Puzzling on StackExchange

Expect at least one logic puzzle during the interview. We're not just interested in whether you solve it, but how you approach it, explain your reasoning, and defend your solution.
If you'd like to train with similar problems, check out the logical-deduction category on ***StackExchange***.