

Q1. What are the core design goals and primary use cases for C?

Answer:

Q2. Explain the type system in C: static vs dynamic typing, type inference, and type safety.

Answer:

Q3. How does C handle memory management: manual, garbage collection, or reference counting?

Answer:

Q4. Describe the concurrency model in C: threads, async/await, coroutines, or actors.

Answer:

Q5. What are the key data structures in C's standard library and when do you use each?

Answer:

Q6. How does C implement object-oriented programming, functional programming, or both?

Answer:

Q7. Explain error handling in C: exceptions, Result/Either types, or error codes.

Answer:

Q8. How does C's package/module system work and what is the standard build tool?

Answer:

Q9. What are common performance optimization techniques specific to C?

Answer:

Q10. How do you write unit tests and measure code coverage in a C project?

Answer: