🌍 International C Programming Syllabus (6 Levels)

Target: Google • Microsoft • Germany • EU • Remote Roles
This syllabus **merges fundamentals + advanced system-level C** to cover **95%+ international interview questions** while remaining practical and teachable.

📘 LEVEL 1: Core Foundations (Beginner → Interview Safe)

Goal: Syntax mastery, clean code, confidence

- C language overview & compilation pipeline (preprocess → compile → link)
- Data types, variables, constants
- Operators & expressions
- Input / Output (printf, scanf)
- Control flow (if, switch, loops)
- Functions (declaration, definition, return values)
- Scope & lifetime
- Coding style (Google readability principles)

🎯 **Interview focus:** - Difference between declaration & definition - Scope-based bugs - Operator precedence

---

🧠 LEVEL 2: Logic, Arrays & Strings (Problem-Solving Core)

Goal: Strong logic & data handling

- Arrays (1D, 2D)
- Array algorithms (search, sort basics)
- Strings & string.h
- Character arrays vs string literals
- Nested loops & pattern problems
- Time complexity basics (Big-O)

🎯 **Interview focus:** - String manipulation without library - Array vs pointer questions

---

🧩 LEVEL 3: Pointers & Memory Fundamentals (Most Critical)

Goal: Master C's hardest concepts

- Pointer basics & arithmetic
- Pointers with arrays & strings
- Call by value vs reference
- Recursion
- Storage classes (static, extern)
- Stack vs heap basics

🎯 **Interview focus:** - Segmentation fault causes - Dangling & wild pointers - Recursion stack behavior

---

⚙️ LEVEL 4: Structures, Files & Dynamic Memory (Job-Ready)

Goal: Real-world program design

- Structures & unions

- Array of structures
- typedef
- Dynamic memory (malloc, calloc, realloc, free)
- Memory leaks & prevention
- File handling (text & binary)

🎯 **Interview focus:** - malloc vs calloc - Structure padding & alignment - File pointer behavior

---

💻 LEVEL 5: System Programming & OS Concepts (Big Tech Core)

Goal: Google/Microsoft depth

- Process memory layout
- Stack vs heap growth
- Command-line arguments
- Low-level file I/O
- System calls (POSIX basics)
- Process creation (fork, exec)
- IPC basics (pipes, signals)
- Error handling (errno, perror)

🎯 **Interview focus:** - How fork() works - Stack overflow vs heap overflow - Zombie processes

---

🛡️ LEVEL 6: Advanced C, Security & Interview Excellence

Goal: International senior-level readiness

- C standards (C99 → C23)
- Undefined vs unspecified behavior
- Bitwise operations
- Preprocessor & macros
- Function pointers & callbacks
- Secure C coding (buffer overflow, CWE)
- Debugging (GDB basics)
- Static & dynamic analysis (Valgrind, ASan)
- Testing fundamentals
- Build systems (Make, CMake)

🎯 **Interview focus:** - Undefined behavior examples - Why gets() is dangerous - Debugging memory corruption

---

🏆 CAPSTONE & INTERVIEW PREP

- Mini system utility project
- Code review simulation
- Whiteboard-style C problems
- Algorithm implementation in C
- Behavioral + technical interview prep

☑ WHAT THIS SYLLABUS GUARANTEES

✓ Covers **Google / Microsoft / EU interview questions** ✓ Suitable for **Germany & remote jobs** ✓ Balanced: fundamentals + systems + security ✓ Practical, not academic-only

---

📌 Recommended Outcome

After completion, a learner can: - Crack C-based interviews - Debug real production issues - Write secure, portable C code - Transition to C++ / OS / Embedded / Backend systems

---

🚀 *This is a complete, international-standard C roadmap.*

🎯 **What this syllabus achieves**

- ✅ Covers **Google & Microsoft interview depth**
- ✅ Suitable for **Germany & EU system/software roles**
- ✅ Strong enough for **remote international jobs**
- ✅ Balanced: **beginner-friendly → system-level mastery**
- ✅ Includes **security, debugging, tooling, OS concepts**

This is **not a normal C course** — it's a **career-grade syllabus**.

📌 **How to use it (important advice)**

- **Levels 1–3** → Focus on *clarity + speed*
- **Levels 4–6** → Focus on *depth + interviews*
- Practice **writing code on paper** (whiteboard style)
- Expect **"why" questions**, not just syntax