C++后端/HPC/算子开发 一年学习&实习求职规划

规划总目标

用12个月完成**通用基础→方向专属→项目实战→求职冲刺**的全流程学习，掌握C++后端、HPC、算子开发三大方向核心技能，具备对应实习岗位的技术能力，成功拿到目标实习offer。

核心原则

1. **先通用后专属**：前4个月夯实通用知识（C++/Linux/计算机基础），后续分方向突破，避免基础不牢导致项目落地困难；
2. **学练结合**：每周既有知识点学习，也有编码实战，每月完成1个小项目/模块，逐步积累完整项目经验；
3. **梯度进阶**：从“理解原理→手写代码→项目落地→性能优化”逐步提升，符合新手学习规律；
4. **求职前置**：第9个月启动求职准备，同步学习+投递，避免临近求职手忙脚乱。

第一阶段：通用核心知识夯实（1-4月）

月度目标

掌握C++11+核心、Linux系统编程、计算机基础（数据结构/体系结构）、工程工具使用，能独立编写基础C++程序，排查简单内存/性能问题。

第1个月：C++基础与工程工具入门

周数	核心学习任务	编码实战任务	验收标准
第1周	1. C++基础语法：变量/类型/流程控制/函数； 2. C++11核心特性：智能指针（unique_ptr/shared_ptr）、nullptr、范围for； 3. Git基础：克隆/提交/分支/PR	1. 手写智能指针简易实现（防拷贝、资源释放）； 2. 用Git管理本地代码仓库	1. 能解释智能指针的原理，手写无BUG的unique_ptr； 2. 熟练执行Git常用命令（git add/commit/push/branch）
第2周	1. STL容器：vector/list/map/unordered_map（底层原理+使用场景）； 2. STL算法：sort/find/transform； 3. CMake基础：单文件/多文件编译、静态库/动态库	1. 实现“基于STL的学生成绩管理系统”（增删改查）； 2. 用CMake配置上述项目编译	1. 能区分map和unordered_map的适用场景，手写排序/查找逻辑； 2. 独立编写CMakeLists.txt，完成项目编译
第3周	1. 内存管理：堆/栈/内存泄漏、new/delete原理； 2. 调试工具：gdb基础（断点/查看变量/堆栈）； 3. 编码规范：Google C++ Style	1. 编写含内存泄漏的代码，用gdb定位并修复； 2. 按规范重构第2周的成绩管理系统	1. 能通过gdb排查段错误、内存越界问题； 2. 代码符合命名/注释规范，无内存泄漏

周数	核心学习任务	编码实战任务	验收标准
第4周	1. C++11进阶：Lambda表达式、函数对象、bind绑定器； 2. 复习本月知识点，整理笔记； 3. valgrind入门：检测内存泄漏	1. 用Lambda表达式简化STL算法调用； 2. 用valgrind检测自己编写的程序，确保无内存问题	1. 能熟练使用Lambda表达式，理解捕获规则； 2. 独立完成本月知识点思维导图，无遗漏

第2个月：C++进阶与Linux系统编程基础

周数	核心学习任务	编码实战任务	验收标准
第1周	1. C++11/14进阶：右值引用/移动语义、emplace_back； 2. RAII设计模式； 3. Linux基础命令：ls/cd/grep/ps/top	1. 实现“基于RAII的文件读写类”（自动关闭文件）； 2. 用Linux命令查看进程/内存使用情况	1. 能解释右值引用和移动语义的作用，手写移动构造函数； 2. 熟练使用10+常用Linux命令
第2周	1. Linux进程管理：fork/exec/waitpid、进程状态； 2. 进程间通信（IPC）：管道（匿名/命名）； 3. perf基础：简单性能分析	1. 实现“父进程通过管道给子进程传数据”； 2. 用perf分析程序运行时间	1. 理解进程创建流程，独立编写进程通信程序； 2. 能通过perf定位程序耗时瓶颈
第3周	1. Linux线程管理：pthread库（创建/退出/等待）； 2. 线程同步：互斥锁/条件变量/原子操作； 3. 死锁排查与避免	1. 实现“多线程售票系统”（避免超卖/死锁）； 2. 模拟死锁场景，定位并修复	1. 能区分进程和线程的区别，手写线程同步代码； 2. 掌握死锁的4个条件，能排查死锁问题
第4周	1. Linux IPC进阶：消息队列/共享内存； 2. 复习本月知识点，整理线程/进程通信笔记； 3. 编写“多线程日志模块”（基础版）	1. 实现“共享内存实现进程间大数据传输”； 2. 完成多线程日志模块（支持多线程写入）	1. 能对比不同IPC方式的优缺点，选择合适方案； 2. 日志模块无线程安全问题，能正常写入文件

第3个月：计算机基础与网络编程入门

周数	核心学习任务	编码实战任务	验收标准
第1周	1. 数据结构：链表（单/双）、栈/队列、哈希表； 2. 算法：时间/空间复杂度分析、冒泡/快排/二分查找； LeetCode每日1题（简单）	1. 手写链表增删改查（含环检测）； 2. 实现快排+二分查找，分析时间复杂度	1. 能手撕链表经典题（反转/环检测）； 2. 快排时间复杂度分析准确，代码无BUG

周数	核心学习任务	编码实战任务	验收标准
第2周	1. 计算机体系结构：CPU缓存（缓存行/预取）、NUMA架构； 2. 内存模型：虚拟内存、分页机制； 3. SIMD指令集入门	1. 编写“缓存友好的数组遍历程序”（优化缓存命中率）； 2. 分析不同遍历方式的性能差异	1. 理解缓存行对性能的影响，能优化简单程序； 2. 能解释虚拟内存的作用
第3周	网络基础：TCP/IP协议栈、TCP三次握手/四次挥手、UDP特点； Socket编程：socket/bind/listen/accept/connect	1. 实现“简单TCP客户端-服务端”（Echo功能）； 2. 抓包查看TCP握手过程	1. 能画出TCP握手流程图，解释各字段含义； 2. 独立编写TCP通信程序，实现消息互发
第4周	1. IO多路复用：select/poll/epoll（原理对比）； 2. epoll LT/ET模式； 3. 复习本月知识点，刷LeetCode中等题5道	1. 用epoll重构TCP服务端（支持多客户端连接）； 2. 对比LT/ET模式的差异	1. 能解释epoll的优势，手写epoll服务端； 2. 区分LT/ET模式的使用场景，代码无BUG

第4个月：通用知识强化+小项目实战

周数	核心学习任务	编码实战任务	验收标准
第1周	1. 模板元编程基础：模板特化、SFINAE； 2. 函数对象与泛型编程； 3. 数据库基础：MySQL安装/基础SQL	1. 实现“泛型数组排序函数”（支持不同类型）； 2. 编写SQL语句（增删改查/联表查询）	1. 能编写简单模板函数，理解泛型编程思想； 2. 熟练执行常用SQL语句，理解索引基础
第2周	1. 线程池设计原理：任务队列、线程管理、任务调度； 2. 无锁编程基础：CAS操作、原子变量； 3. valgrind进阶：内存泄漏/越界检测	1. 实现“简单线程池”（支持任务提交/线程复用）； 2. 用valgrind检测线程池内存问题	1. 理解线程池的核心模块，手写无BUG的线程池； 2. 能通过valgrind定位复杂内存问题
第3周	1. 异步日志模块：日志级别、异步落盘、缓冲区设计； 2. TCP粘包/拆包问题：原因+解决方案（固定长度/分隔符/协议头）	1. 给TCP服务端添加异步日志模块； 2. 解决TCP粘包问题，实现可靠消息传输	1. 日志模块支持异步落盘，不阻塞主线程； 2. 能处理粘包，确保消息完整解析
第4周	1. 通用知识总复习：梳理C++/Linux/网络/数据结构知识体系； 2. 第一个小项目：“高性能TCP Echo服务器”（整合线程池+epoll+日志）	1. 完成TCP服务器项目，编写README； 2. 压测：支持1000+并发连接，无崩溃	1. 项目代码规范，无内存泄漏/线程安全问题； 2. 能解释项目各模块设计思路，完成压测

第二阶段：方向专属知识学习+项目拆分实战（5-8月）

月度目标

分方向学习C++后端、HPC、算子开发的专属知识，完成每个方向的核心模块开发，为完整项目打基础。

第5个月：C++后端专属知识（网络/数据库/分布式）

周数	核心学习任务	编码实战任务	验收标准
第1周	1. 网络编程进阶：Reactor/Proactor模式； 2. epoll反应堆模型； 3. TCP心跳机制/超时重连	1. 用Reactor模式重构TCP服务器； 2. 添加心跳机制，检测客户端离线	1. 理解Reactor模式原理，重构后的服务器支持高并发； 2. 能检测客户端异常离线，自动清理连接
第2周	1. MySQL进阶：索引原理（B+树）、事务ACID、隔离级别； 2. 数据库连接池设计：复用连接、超时回收	1. 实现“简单MySQL连接池”（支持连接复用）； 2. 优化SQL语句（添加索引提升查询速度）	1. 能解释B+树索引的优势，手写连接池； 2. 索引优化后查询耗时降低50%+
第3周	1. 分布式基础：一致性哈希、CAP理论、BASE理论； 2. RPC框架原理：序列化/反序列化、远程调用流程； 3. Protobuf入门：定义接口、编译使用	1. 实现“一致性哈希算法”（支持节点增删）； 2. 用Protobuf定义RPC服务接口	1. 能解释一致性哈希的作用，手写算法； 2. 熟练定义Protobuf接口，编译生成代码
第4周	1. HTTP协议：请求/响应格式、状态码、RESTful API； 2. 把TCP服务器改造为“简易HTTP服务器”； 3. 复习本月后端知识	1. 实现HTTP服务器，支持GET/POST请求； 2. 解析HTTP请求参数，返回JSON响应	1. 能解析HTTP请求，返回正确响应； 2. 理解RESTful API设计规范，接口符合标准

第6个月：HPC专属知识（并行计算/异构编程）

周数	核心学习任务	编码实战任务	验收标准
第1周	1. 并行计算基础：数据并行/任务并行、阿姆达尔定律； 2. OpenMP入门：编译指令、并行区域、数据共享； 3. Eigen库：矩阵运算基础	1. 用OpenMP实现“并行求和/矩阵乘法”； 2. 对比串行/并行版本性能差异	1. 理解OpenMP核心指令，手写并行程序； 2. 8核CPU下并行矩阵乘法提速7倍+
第2周	1. MPI入门：点对点通信、集体通信（广播/归约）； 2. 分布式并行：数据分片、节点通信； 3. MPI环境搭建（单机多进程模拟）	1. 用MPI实现“分布式矩阵乘法”（4进程）； 2. 分析通信开销，优化数据分片	1. 能编写MPI并行程序，实现进程间通信； 2. 分布式矩阵乘法正确运行，结果与串行一致

周数	核心学习任务	编码实战任务	验收标准
第3周	1. CUDA编程基础：CUDA架构（SM/线程束）、内存模型（全局/共享/常量内存）； 2. 核函数编写、主机/设备数据传输	1. 用CUDA实现“向量加法”； 2. 对比CPU/CUDA版本性能	1. 理解CUDA内存模型，手写核函数； 2. CUDA版本向量加法提速10倍+
第4周	1. CUDA进阶：核函数优化、共享内存使用、线程块划分； 2. NVIDIA Nsight：性能分析、瓶颈定位； 3. 复习本月HPC知识	1. 用CUDA实现“矩阵乘法”，优化共享内存； 2. 用Nsight分析并优化性能	1. 共享内存优化后GPU利用率提升至80%+； 2. 能定位CUDA程序的内存/计算瓶颈

第7个月：算子开发专属知识（深度学习框架/算子优化）

周数	核心学习任务	编码实战任务	验收标准
第1周	1. 深度学习基础：张量、计算图、自动微分； 2. PyTorch入门：张量操作、模型定义、前向传播； 3. PyTorch C++扩展入门	1. 用PyTorch实现简单神经网络（线性回归）； 2. 编写PyTorch C++扩展（简单函数）	1. 理解张量和计算图原理，能训练简单模型； 2. 成功编译并调用C++扩展
第2周	1. 算子开发基础：算子注册机制、前向/反向传播； 2. 卷积算子数学原理：卷积核、步长、填充； 3. CUDA卷积算子入门	1. 推导卷积算子计算公式； 2. 用CUDA实现“简单卷积算子”（3×3核）	1. 理解卷积的数学逻辑，手写计算公式； 2. CUDA卷积算子能正确计算结果
第3周	1. 算子优化：算子融合、向量化计算、内存访问优化； 2. TensorRT入门：模型转换、推理引擎构建； 3. ONNX格式：模型导出/解析	1. 优化CUDA卷积算子（共享内存/向量化）； 2. 导出PyTorch模型为ONNX格式	1. 优化后卷积算子性能提升20%+； 2. 成功导出ONNX模型，能解析模型结构
第4周	1. 异构硬件适配：GPU/NPU架构差异、算子适配思路； 2. 精度校准：FP32/FP16/INT8精度对比； 3. 复习本月算子开发知识	1. 实现卷积算子的FP16精度版本； 2. 对比不同精度的性能/精度差异	1. 理解不同精度的优缺点，实现精度转换； 2. FP16版本提速30%，精度损失<0.5%

第8个月：方向项目模块整合（完成3个核心项目初稿）

周数	核心学习任务	编码实战任务	验收标准
第1周	1. C++后端项目：轻量级RPC框架（通信层+协议层）； 2. 学习RPC服务注册与发现原理	1. 实现RPC客户端/服务端通信； 2. 支持Protobuf序列化/反序列化	1. 能完成远程方法调用，结果正确； 2. 代码模块化，易扩展
第2周	1. HPC项目：CUDA图像滤波加速（Sobel/高斯模糊）； 2. 学习OpenCV图像读写/处理	1. 实现CPU/GPU版本图像滤波； 2. 对比处理速度，优化GPU版本	1. 图像滤波结果正确，GPU版本提速50+倍； 2. 能解释优化思路（共享内存/线程块）
第3周	1. 算子开发项目：自定义卷积算子（集成到PyTorch）； 2. 实现自动微分功能	1. 卷积算子接入PyTorch，支持反向传播； 2. 用算子训练简单CNN模型	1. 算子能参与模型训练，梯度更新正确； 2. 模型收敛，精度达标
第4周	1. 三个项目功能完善+BUG修复； 2. 编写项目文档（设计思路/使用说明/性能指标）； 3. 代码开源到GitHub（初始化仓库）	1. 完善项目README，补充注释； 2. 修复项目中所有BUG，确保稳定运行	1. 三个项目无核心BUG，能独立运行； 2. GitHub仓库结构清晰，文档完整

第三阶段：项目优化+求职冲刺（9-12月）

月度目标

完成项目性能优化、简历制作、笔面试准备，批量投递实习岗位，最终拿到C++后端/HPC/算子开发方向的实习offer。

第9个月：项目性能优化+求职准备

周数	核心学习任务	编码实战/求职任务	验收标准
第1周	1. C++后端项目优化：添加负载均衡（轮询）、超时重试； 2. 学习高并发系统性能调优技巧	1. 优化RPC框架，支持负载均衡； 2. 压测：单机支持10k+并发调用	1. 负载均衡能降低单节点负载30%； 2. 压测无崩溃，响应延迟<10ms
第2周	1. HPC/算子项目优化：HPC矩阵乘法MPI优化、算子TensorRT集成； 2. 学习性能分析工具进阶用法	1. MPI矩阵乘法4节点提速3倍； 2. 卷积算子接入TensorRT，推理提速40%	1. 性能指标达到简历量化标准； 2. 能解释所有优化点的原理
第3周	1. 简历制作：梳理技能/项目/成果，量化项目亮点； 2. 学习简历优化技巧（突出匹配度）； 3. 整理目标岗位列表（按方向分类）	1. 完成3版简历（后端/HPC/算子）； 2. 筛选50+目标实习岗位（大厂/中厂/科研机构）	1. 简历突出核心技能+量化成果； 2. 岗位列表覆盖三个方向，匹配自身技能

周数	核心学习任务	编码实战/求职任务	验收标准
第4周	1. 笔试准备：刷LeetCode Hot 100（中等题）； 2. 整理C++/Linux/网络/并行计算笔试考点； 3. 注册实习僧/BOSS直聘/牛客网账号	1. 完成50道算法题，整理错题本； 2. 梳理笔试考点思维导图	1. 算法题正确率≥80%； 2. 考点覆盖90%+高频内容

第10个月：批量投递+笔试冲刺

周数	核心学习任务	求职/实战任务	验收标准
第1周	1. 投递策略：日常实习批量投递（3-5家/天）； 2. 笔试高频题复盘：C++内存管理/网络编程/HPC基础	1. 投递20+日常实习岗位； 2. 刷10套大厂笔试真题（后端/HPC）	1. 投递岗位匹配度≥80%； 2. 笔试真题正确率≥70%
第2周	1. 算子开发笔试准备：深度学习框架/算子优化真题； 2. 项目复盘：梳理每个项目的难点/解决方案/优化点	1. 刷5套算子开发相关笔试真题； 2. 编写项目复盘文档（每个项目3个难点+解决方案）	1. 算子笔试正确率≥65%； 2. 能流畅描述项目难点和解决思路
第3周	1. 面试准备：整理高频面试题（基础/项目/技术视野）； 2. 模拟面试：和同学复盘项目/回答问题； 3. 跟进已投递岗位，补充投递	1. 整理100+高频面试题及答案； 2. 完成2次模拟面试，复盘不足； 3. 累计投递50+岗位	1. 能流利回答80%+面试题； 2. 模拟面试反馈良好，调整回答思路
第4周	1. 暑期实习岗位关注（1-3月申请）； 2. 技术热点学习：云原生/大模型算力/国产芯片； 3. 开源贡献：给muduo/PyTorch提小PR	1. 筛选10+暑期实习目标岗位； 2. 提交1个开源PR（BUG修复/注释优化）； 3. 复习所有笔试考点	1. 暑期岗位列表明确，匹配自身技能； 2. PR通过审核或获得正面反馈

第11个月：面试冲刺+offer跟进

周数	核心学习任务	求职/实战任务	验收标准
第1周	1. 面试高频问题强化：项目深挖/技术原理/场景题； 2. 针对已收到的笔试/面试邀请，定制准备	1. 针对目标公司整理专属面试题； 2. 参加3-5场笔试/面试	1. 能针对性回答公司业务相关问题； 2. 笔试/面试后及时复盘，调整策略
第2周	1. 薪资/实习时长谈判技巧； 2. 补充投递高匹配岗位（科研机构HPC/芯片公司算子）； 3. 项目最终优化（补充技术博客）	1. 投递10+高匹配岗位； 2. 撰写2篇技术博客（项目难点解析）； 3. 跟进面试结果	1. 技术博客发布到CSDN/掘金，阅读量≥500； 2. 收到至少2个面试邀约

周数	核心学习任务	求职/实战任务	验收标准
第3周	1. 面试复盘：整理错题/回答不足，强化薄弱点； 2. 学习行业动态：大厂技术栈/岗位需求变化； 3. 内推渠道拓展：师兄师姐/牛客内推	1. 补充薄弱知识点（如CUDA高级优化）； 2. 获得5+内推机会； 3. 参加2-3场内推面试	1. 薄弱知识点掌握度提升至90%； 2. 内推面试通过率≥50%
第4周	1. offer筛选：对比薪资/转正率/技术方向； 2. 实习前准备：学习目标公司技术栈； 3. 拒绝/接受offer的沟通技巧	1. 收到至少1个实习offer； 2. 了解目标公司技术栈，补充学习； 3. 完成offer确认/拒绝沟通	1. 拿到符合预期的实习offer； 2. 掌握目标公司核心技术栈基础

第12个月：实习准备+知识查漏补缺

周数	核心学习任务	实战/求职任务	验收标准
第1周	1. 目标实习岗位技术栈强化：如大厂后端的brpc/算子开发的TensorRT； 2. 整理学习笔记，形成知识体系	1. 完成目标岗位技术栈入门学习； 2. 整理全年学习笔记，归档项目代码	1. 能使用目标技术栈编写简单程序； 2. 笔记/代码分类清晰，便于后续复习
第2周	1. 实习前准备：Linux环境配置/代码规范/团队协作流程； 2. 模拟实习场景：解决实际业务问题（如高并发bug）	1. 配置符合大厂规范的开发环境； 2. 完成2个模拟业务问题的解决	1. 开发环境适配目标公司要求； 2. 能独立解决简单业务问题
第3周	1. 三个方向知识查漏补缺：回顾全年知识点，补充遗漏； 2. 项目维护：GitHub项目更新/回复issues	1. 完成全年知识点复盘，补充薄弱点； 2. 维护开源项目，提升活跃度	1. 知识点无明显遗漏，能应对技术问答； 2. 开源项目获得少量star/fork
第4周	1. 实习入职准备：简历/作品集/学习总结； 2. 心态调整：实习预期/职场沟通技巧； 3. 长期规划：实习期间学习目标	1. 整理实习作品集（项目/博客/开源贡献）； 2. 制定实习期间的学习计划； 3. 做好入职准备	1. 作品集完整，能展示核心能力； 2. 实习计划明确，聚焦技术提升

关键配套建议

1. **学习资源推荐**：
    - C++：《C++ Primer》《Effective Modern C++》、侯捷C++视频；
    - Linux/网络：《UNIX环境高级编程》《Linux高性能服务器编程》；
    - HPC/CUDA：《CUDA编程指南》《并行程序设计导论》；
    - 算子开发：PyTorch官方文档、TensorRT开发者指南；
    - 算法：LeetCode Hot 100、《剑指Offer》。

2. **时间管理**：
    - 每周保证20+小时有效学习时间（工作日2-3小时/天，周末8-10小时）；
    - 每天记录学习进度，每周复盘完成情况，未完成项顺延至下周优先完成。
3. **避坑提醒**：
    - 不要只看视频不写代码：每个知识点必须手写代码验证，避免“眼会手不会”；
    - 不要忽视基础：通用知识阶段务必学扎实，否则后续项目会频繁卡壳；
    - 不要拖延求职：第9个月必须启动投递，不要等所有知识学完再投，边投边学效率更高。

总结

这份规划以“通用基础→方向专属→项目实战→求职冲刺”为核心逻辑，12个月覆盖C++后端、HPC、算子开发三大方向的全流程学习，每月/每周任务具体可落地，既保证知识体系的完整性，又兼顾实习求职的时间节点。执行过程中可根据自身学习速度微调进度，但核心节奏（前4个月打基础、9个月启动求职）不变，坚持完成即可具备对应实习岗位的核心竞争力。