概述：

1. 使用protobuf作为数据（参数）序列化和反序列化方式，
2. 使用protobuf的service功能并重写远程调用的服务类以实现远程调用
3. 通过muduo网络库完成调用方法请求和参数的网络传输，可实现高并发网络IO服务，基于此解耦网络和业务模块代码
4. 通过zookeeper存储各调用方法所在服务器的ip和port，客户端通过查询zookeeper找到所需方法所在的服务器
5. 维护一个日志队列，并通过RAII（Resource Acquisition Is Initialization）中的lock_gurad、unique_lock和条件变量进行日志的异步存取

环境： Ubuntu 22.04.3 LTS gcc version 11.4.0 (Ubuntu 11.4.0-1ubuntu1~22.04)

依赖： muduo网络库 protobuf

proto编译指令 protoc [--proto_path=IMPORT_PATH] --cpp_out=DST_DIR file.proto 其中： protoc：编译工具 –proto_path：指定 .proto 文件的检索路径，可以多次指定指定。不指定默认在当前文件夹下检索。可以简写为 -I. –cpp_out：指定编译后的文件类型为C++ DST_DIR：指定文件的生活生成路径 file.proto：指定要编译的 .proto 文件（–proto_path 路径下的）

使用方法： 首先，需要使用上述protobuf编写服务方法和需要传输的参数，在example文件夹中有示例，或者上网查询，编译方法如上 其次，按照example中的示例代码重写protobuf中的类方法，功能自定 完成后，代码中附有autobuild.sh一键编译脚本，可以编译出静态链接库和相应的server(provider)和client(consumer)可执行文件 最后，先启动服务端，再启动客户端： ./provider -i test.conf ./consumer -i test.conf 注：consumer只需要带有zookeeper的ip和端口就行，可另写一个conf； provider需要有自身的serverip和port，也需要zookeeper的ip和port