**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：董文龙**

**学 号：202222080411**

**指导教师：丁旭阳**

**实验地点：主楼B412**

**实验时间：2022年11月8日**

1. **实验室名称：**

Linux环境高级编程实验室

**二、实验项目名称：**

Linux编程环境搭建与使用

**三、实验学时：**

4学时

**四、实验目的：**

1. 因为Linux系统相比Windows系统资源占用更少，性能更好的原因，更适合于做服务器，同时常见的大数据平台都是在 Linux 系统运行的，比如 hadoop、hive、hbase、spark。因此学习Linux更能帮助我们更好地与服务器交互和掌握大数据相关技术有更好的帮助。

2. 熟悉并完成Linux（Ubuntu 20.04）的安装和基本环境的搭建，包含（网络环境配置，APT工具使用，XShell配置）

3. 熟悉并完成高级vi的使用。

4. 熟悉并完成Make命令的使用。

5. 熟悉并完成gdb命令的使用。

6. 熟悉并完成googletest的使用。

7. 熟悉并完成valgrind的使用。

**五、实验内容：**

1. 本次实验需要完成Linux（Ubuntu 20.04）的安装和基本环境的搭建，包含（网络环境配置，APT工具使用，XShell配置）。

2. 本次实验需要完成高级vi的使用。

3. 本次实验需要完成Make命令的使用。

4. 本次实验需要完成gdb命令的使用。

5. 本次实验需要完成googletest的使用。

6. 本次实验需要完成valgrind的使用。

**六、实验步骤：**

6.1. 虚拟机的安装

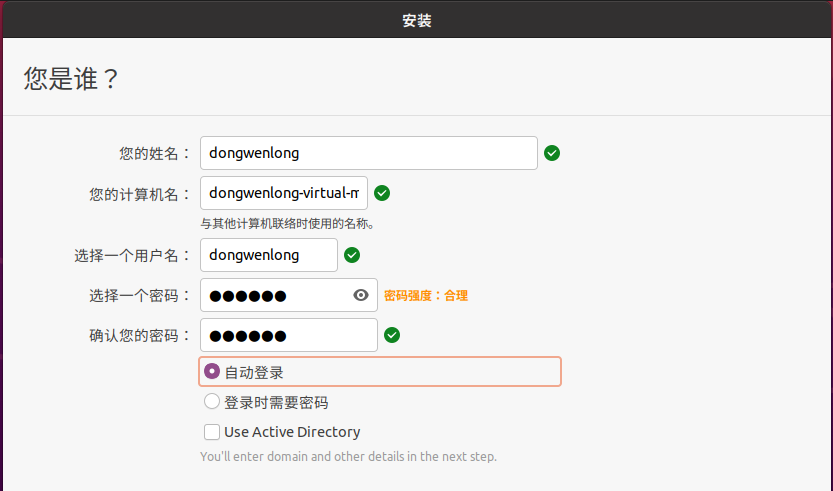
1. 虚拟机下载：[VMWare WorkStation Pro](https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html)

2．Ubuntu 20.04 ISO下载：[Ubuntu 20.04](https://releases.ubuntu.com/focal/ubuntu-20.04.5-desktop-amd64.iso)

3．创建新的虚拟机🡪自定义（高级）🡪稍后安装操作系统🡪Linux（Ubuntu）🡪自定义名称选择安装位置🡪选择处理器数量（2\*2）🡪4GB内存🡪使用网络地址转换（NAT模式）🡪SCSI控制器（LSI Logic）🡪虚拟磁盘类型（SCSI）🡪创建新虚拟机磁盘🡪磁盘大小（40GB，拆分为单个文件）🡪磁盘文件名（默认）🡪自定义硬件🡪新CD/DVD（使用ISO镜像文件，选择下载好的ISO文件）🡪关闭🡪完成🡪打开虚拟机即可

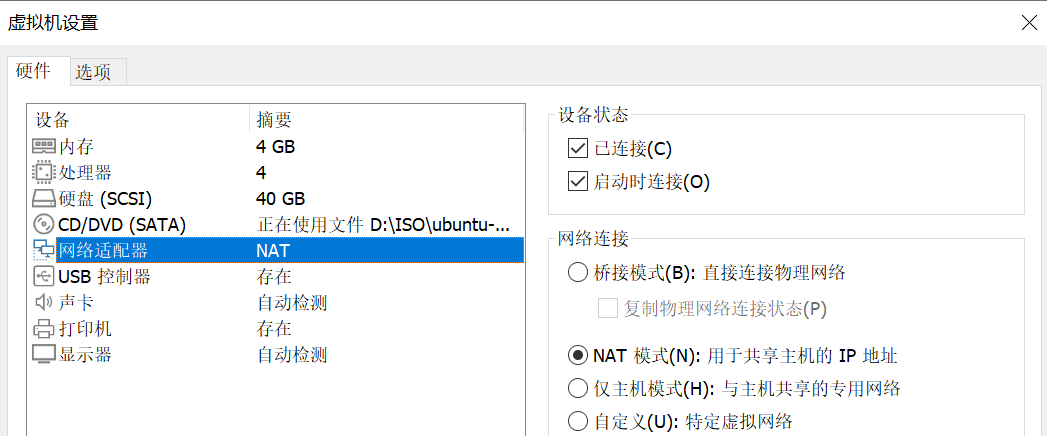


4. 打开虚拟机进行安装，中文简体🡪安装Ubuntu🡪输入用户名和密码，完成安装。



6.2. VMWare网络配置

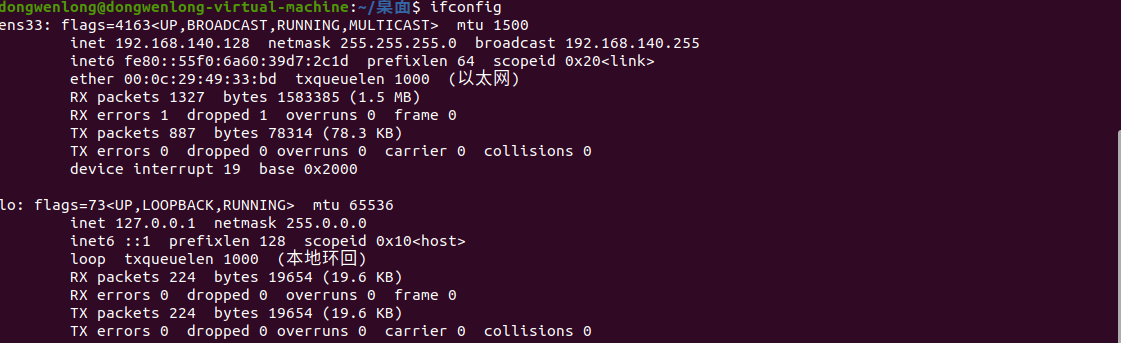
1. VMWare菜单栏虚拟机🡪设置🡪硬件🡪网络适配器🡪NAT🡪安装完成后需要重启



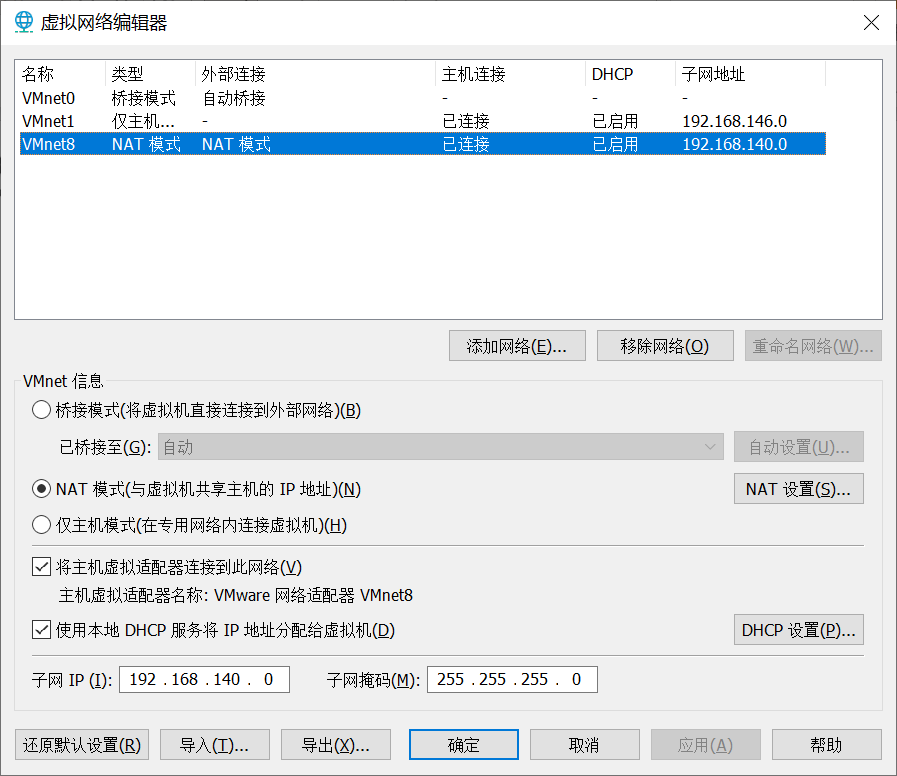
2.安装net-tools



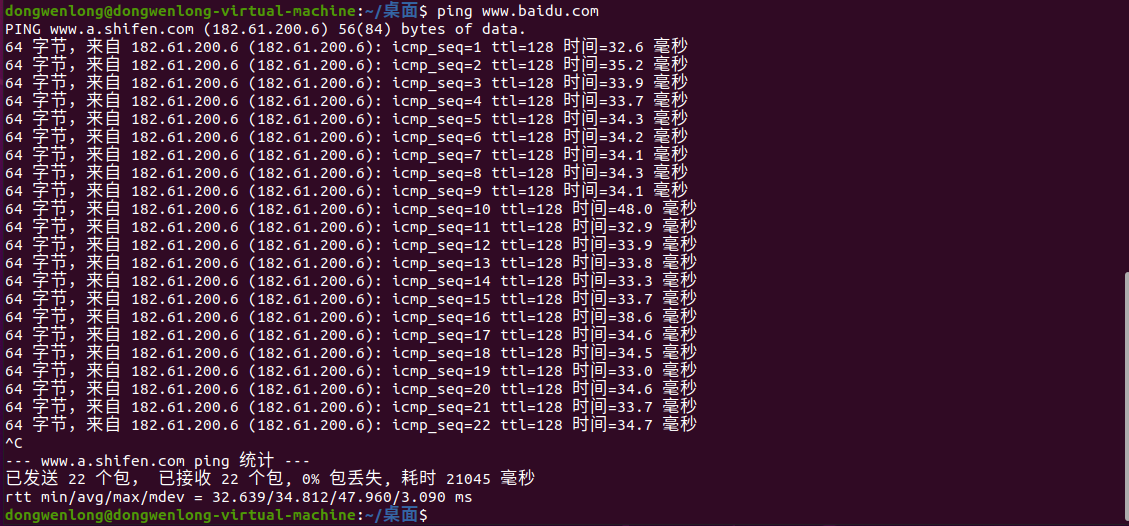
3. 查看当前网卡



4. 编辑🡪虚拟网络编辑器，确定网络当前配置

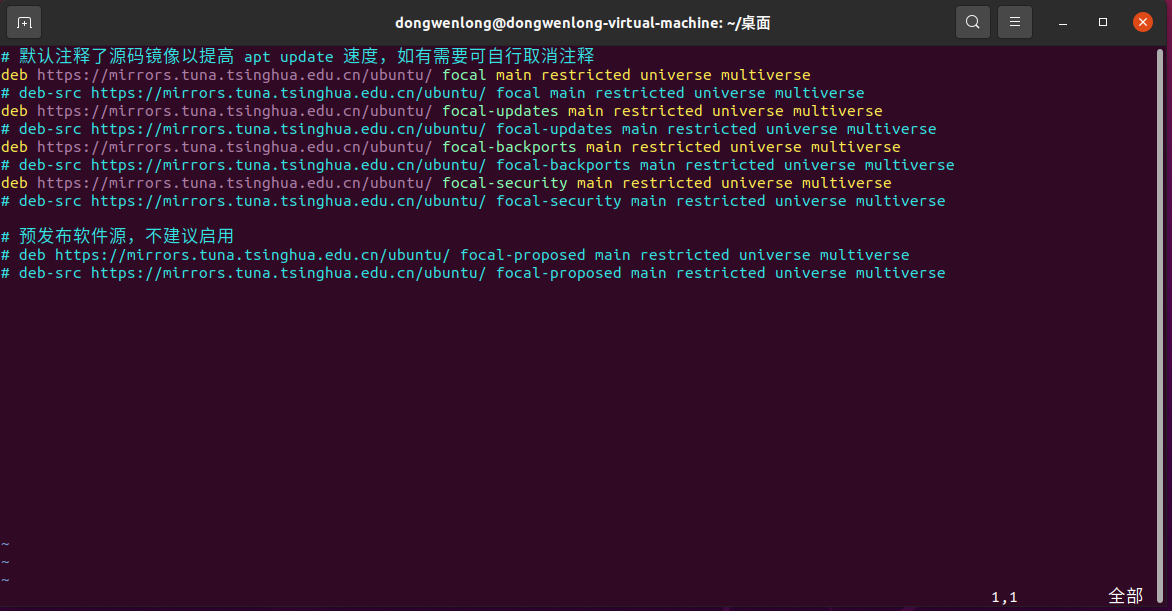


5. 检查网络是否正常，ping www.baidu.com

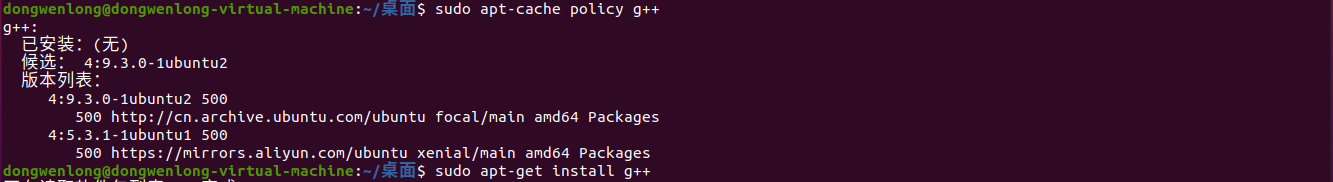


6.3. APT的使用

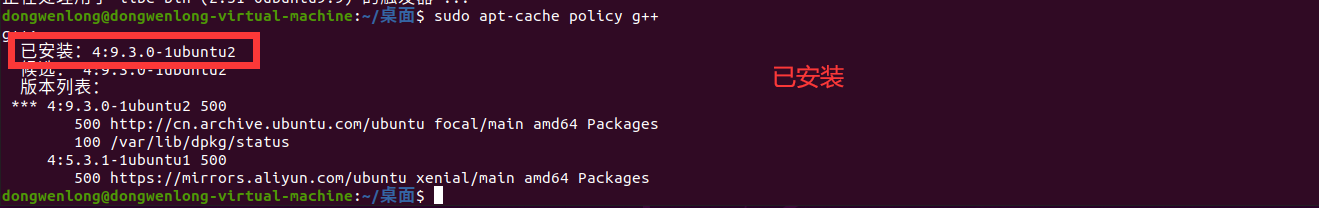
1. 加入APT源



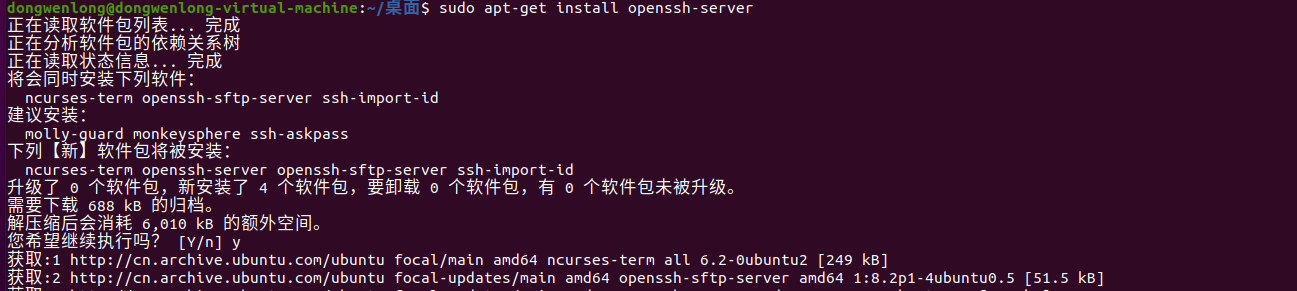
2. 查找是否安装了g++，如果没有进行安装

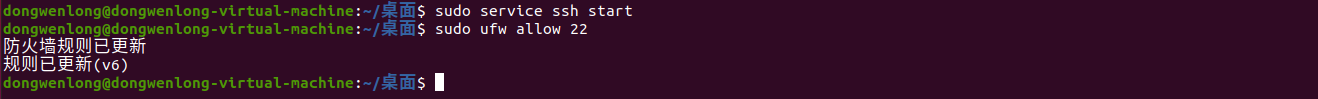






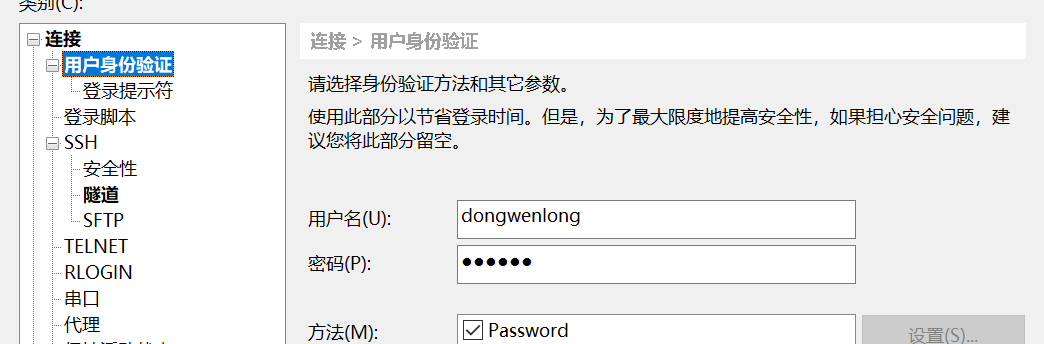
3. 安装OpenSSH，确保启动SSH服务，开发22端口

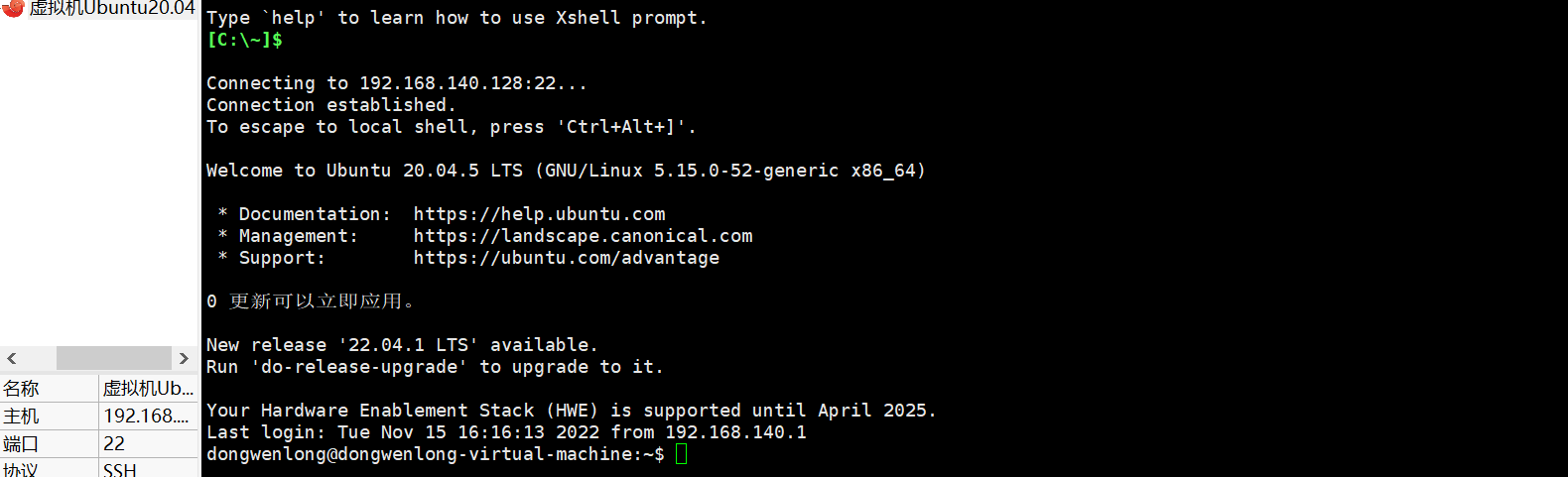




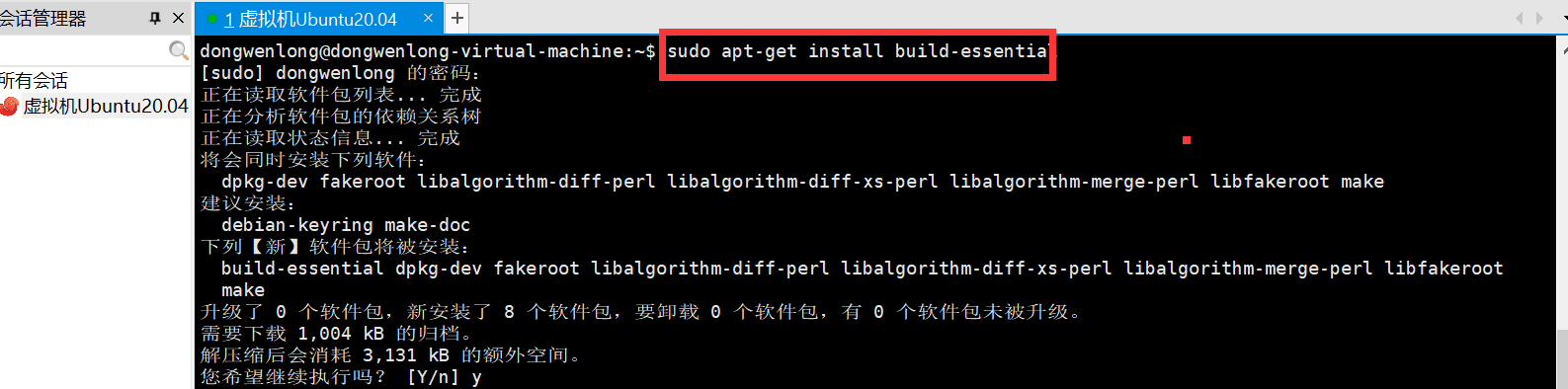
4. XShell连接，输入虚拟机的ip，用户名和密码，连接成功。





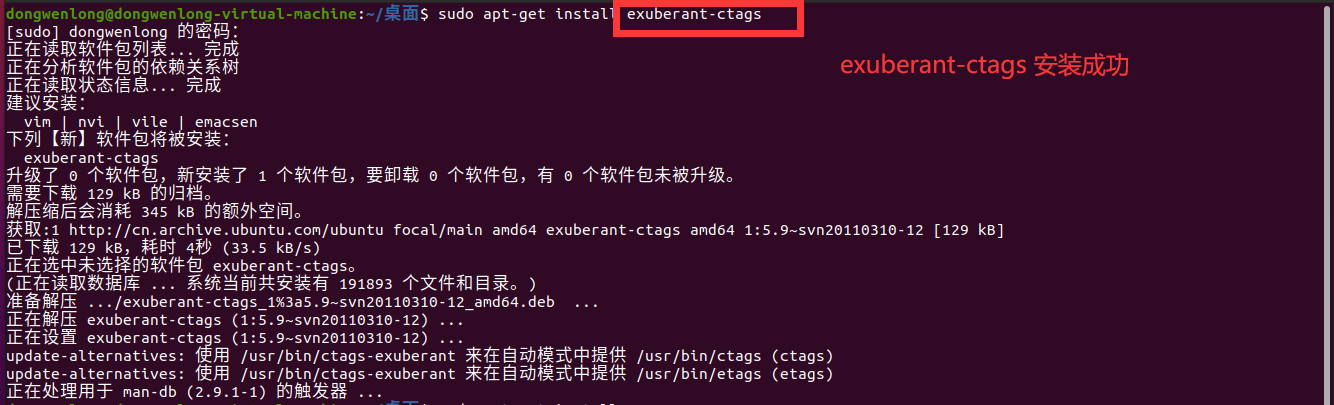


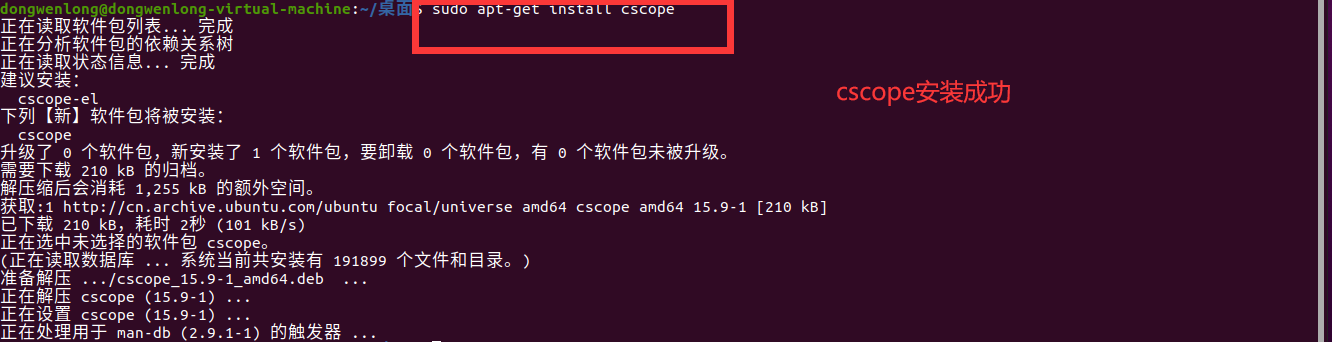
5. build-essential下载，通过XShell可以正常安装。



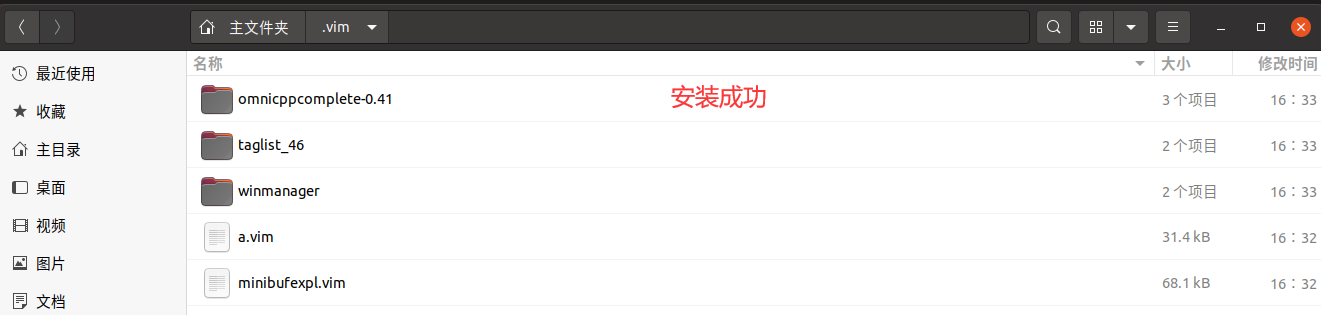
6.4．Vi的高级使用

1. vi插件安装成功







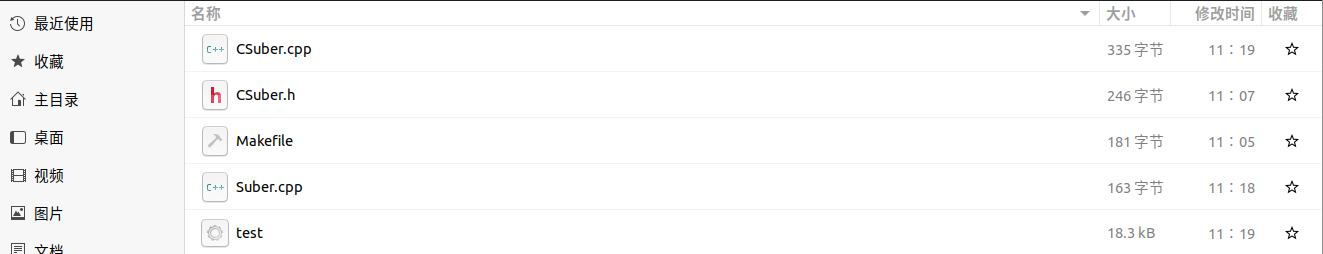


2. vimrc文件解释

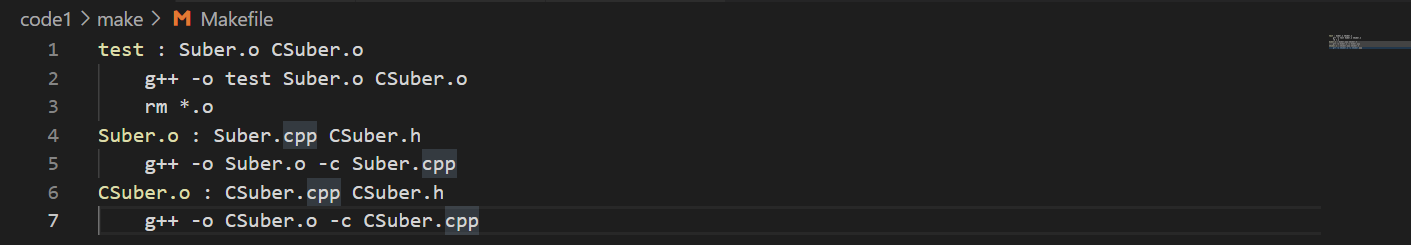


6.5．Make的使用（从这一步开始使用VSCode远程连接书写代码）

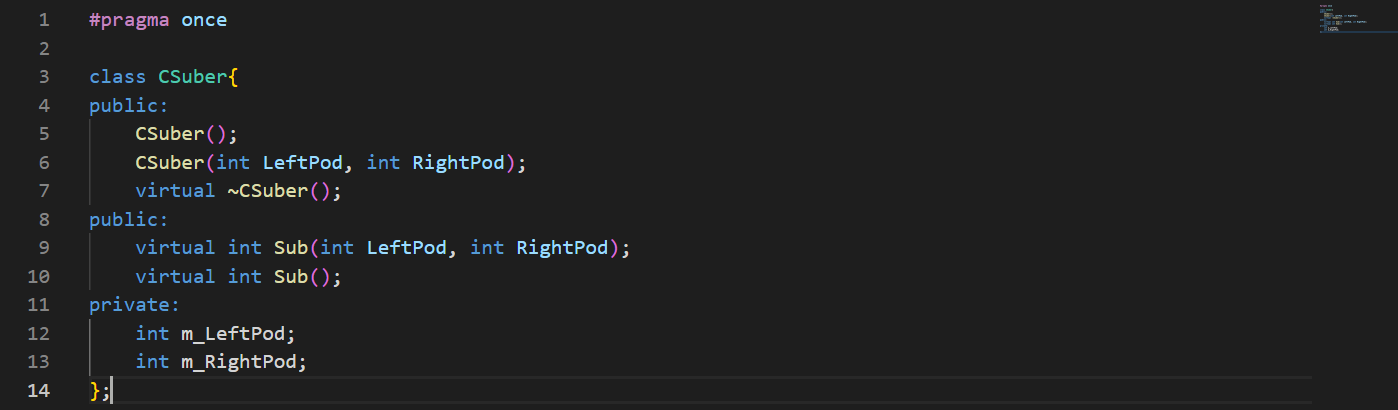
1. 文件目录结构如下：



2. Makefile编写如下：



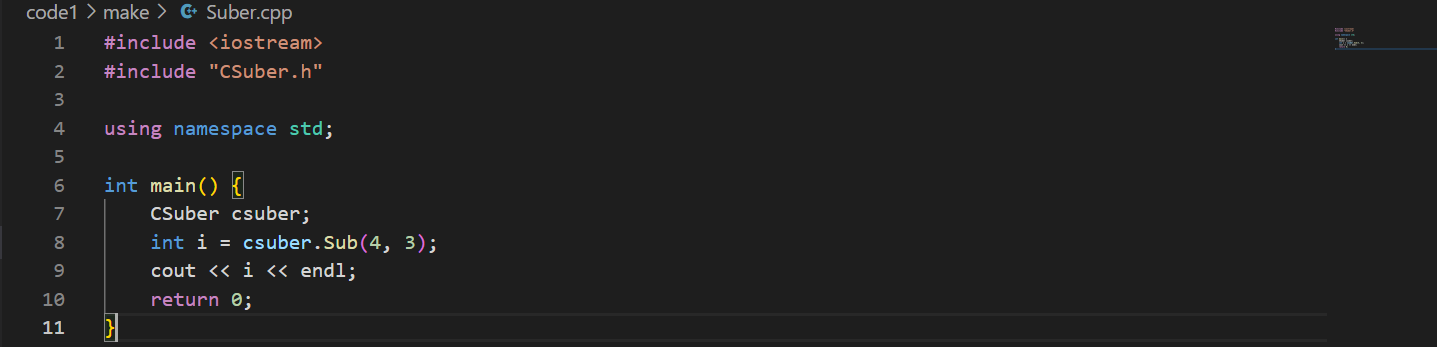
3. CSuber.h代码如下：



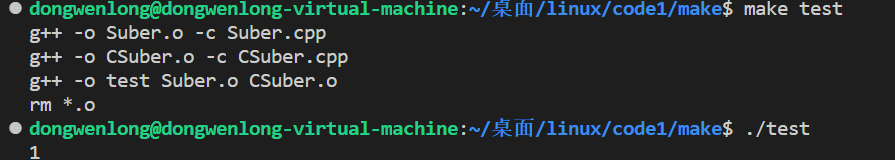
4. CSuber.cpp代码如下：



5. Suber.cpp代码如下：

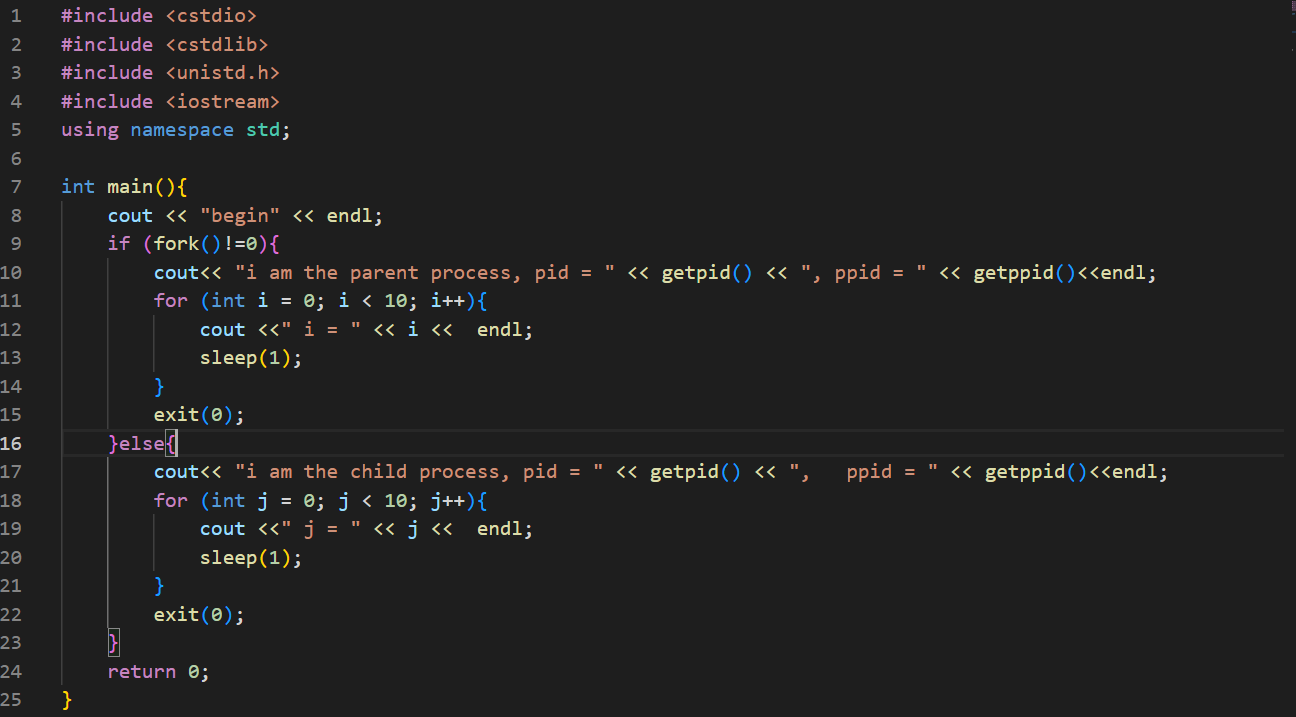


6. 执行make test 和 ./test结果为



6.6. GDB的使用

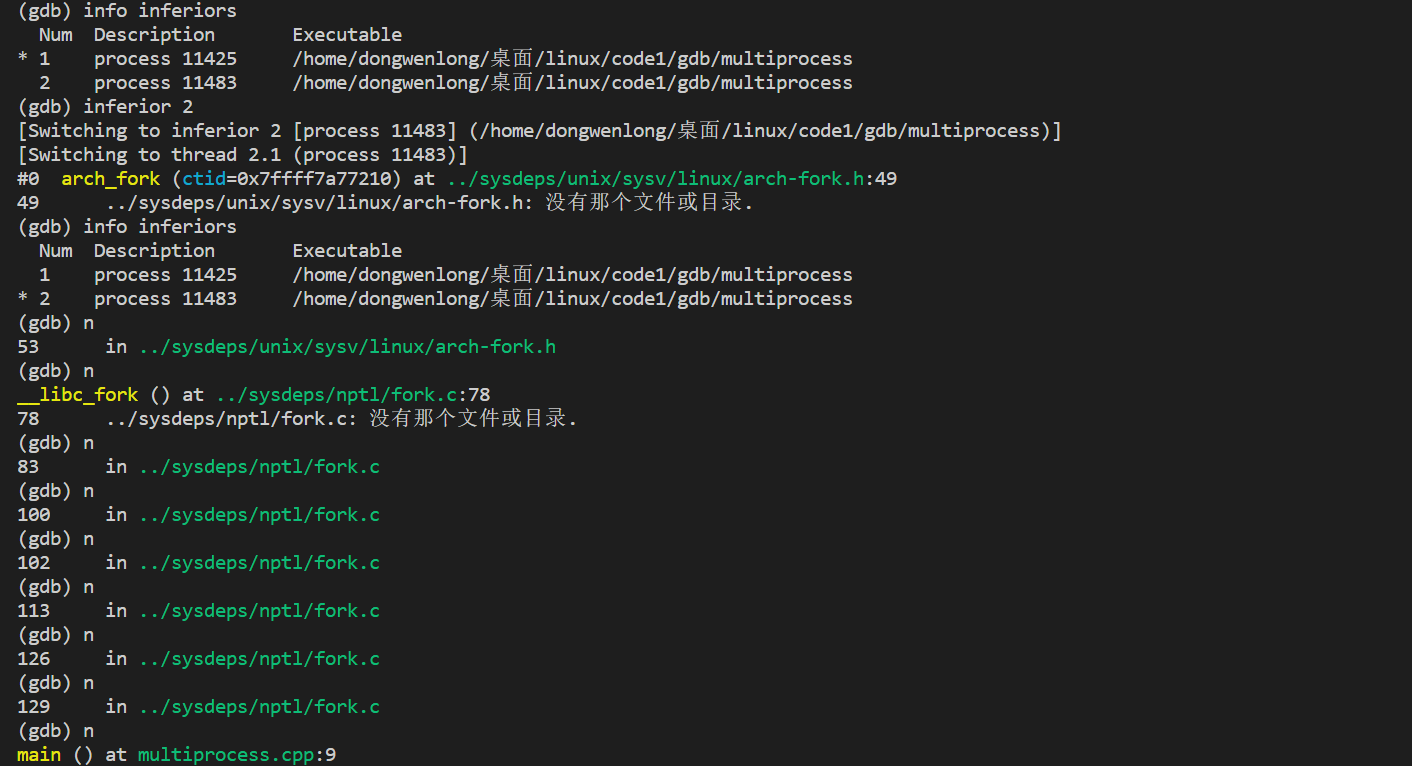
1. 源代码



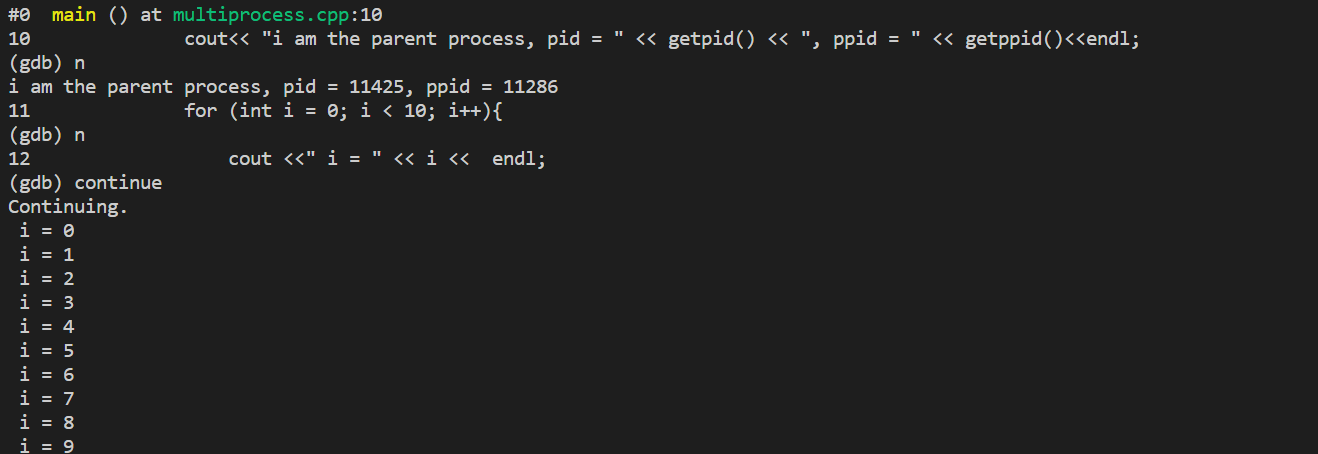
2. 先detach 父子进程，打上断点。



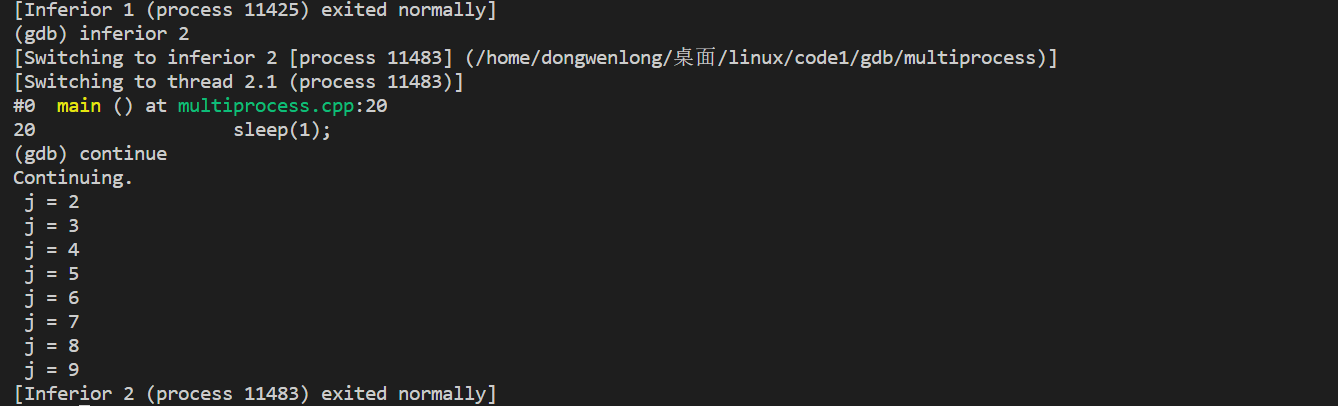
3. 查看进程信息，并切换到子进程



4. 切换到父进程



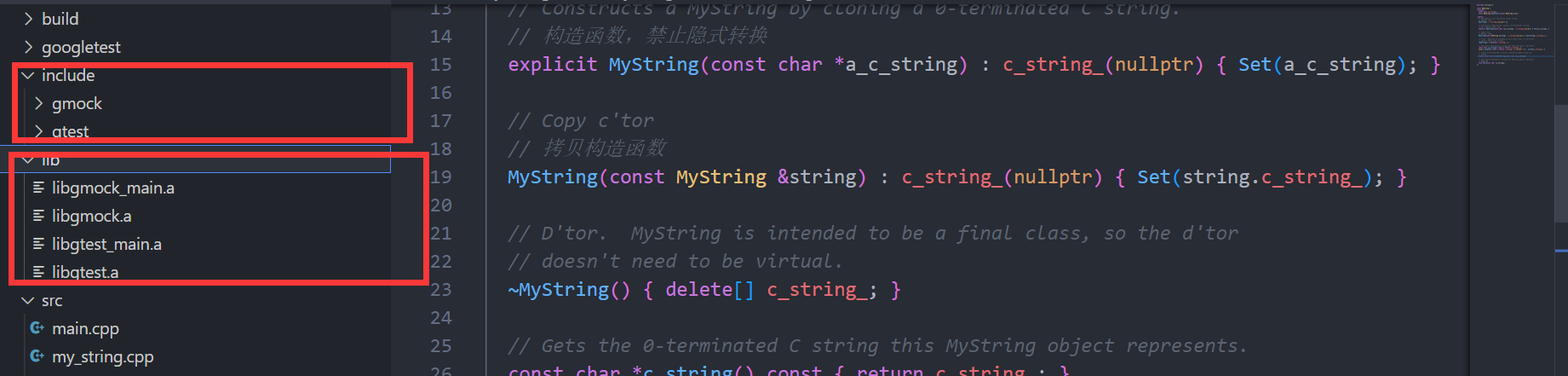
5. 切换到子进程



6.7. Googletest使用

1. google test 下载安装与使用。使用git进行克隆（将虚拟机配置代理，转发为本地ip和端口，之后更改git终端代理，最后克隆下来）

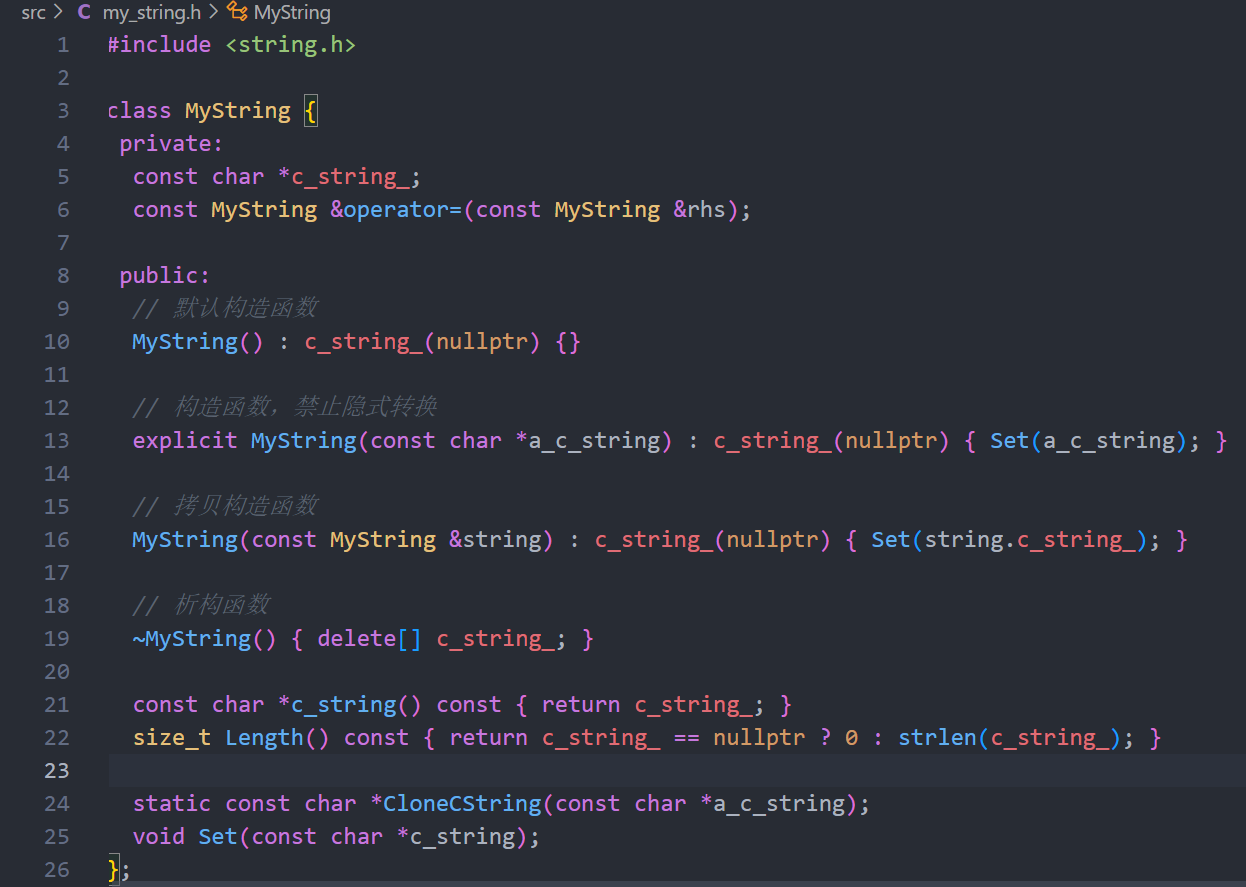
2. 下载之后进行编译，将其提取出来，结构如下（gtest和gmock我都提取出来了，本实验完成gtest即可）



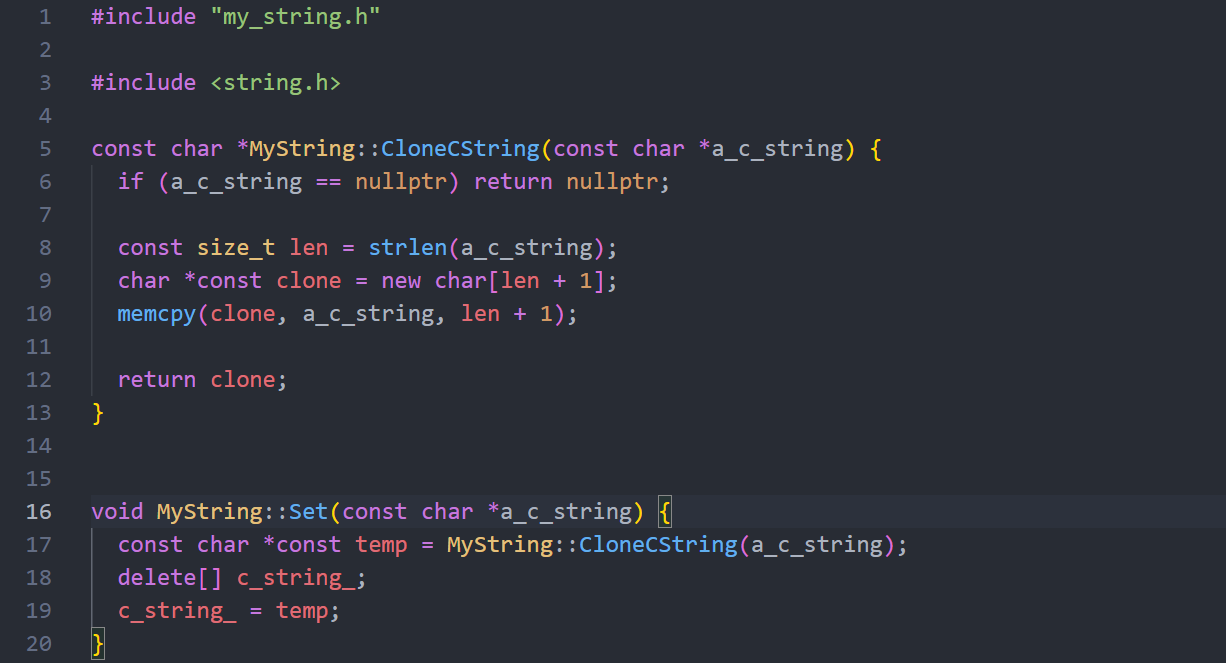
3. 编写CMakeLists文件如下



4. my\_string.h 实现如下，文件中有一个名为MyString的类，包括测试构造函数和成员函数。该类有如下特征：默认构造函数把成员变量c\_string指针初始化为nullptr，构造函数MyString接收一个char \*字符串然后通过Set函数拷贝给c\_string\_。

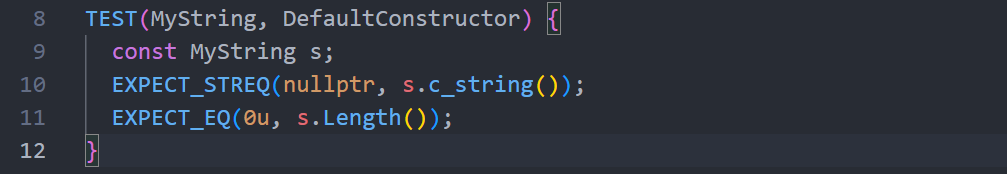


5. my\_string.cpp 实现如下，实现CloneCString和Set函数

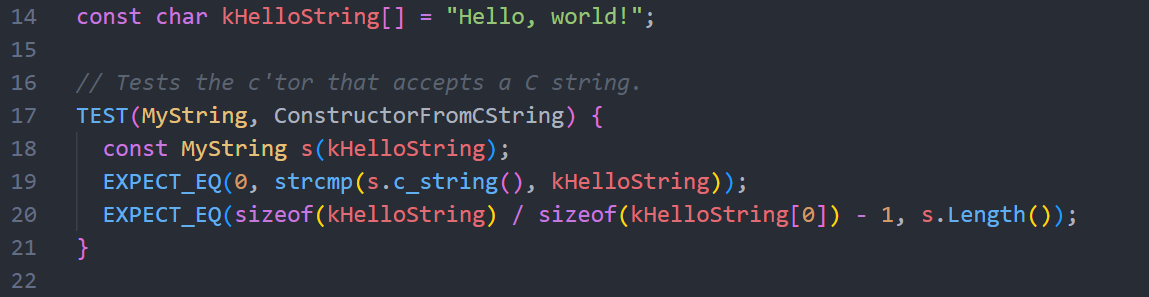


6. main.cpp 实现如下

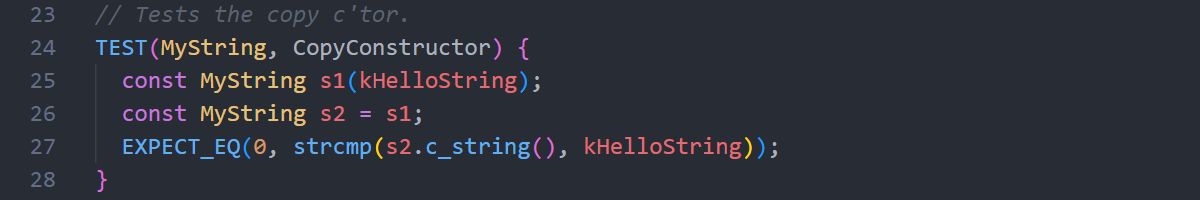
第一个用例：TEST(MyString, DefaultConstructor)，测试默认构造函数， MyString() : c\_string\_(nullptr) {}。



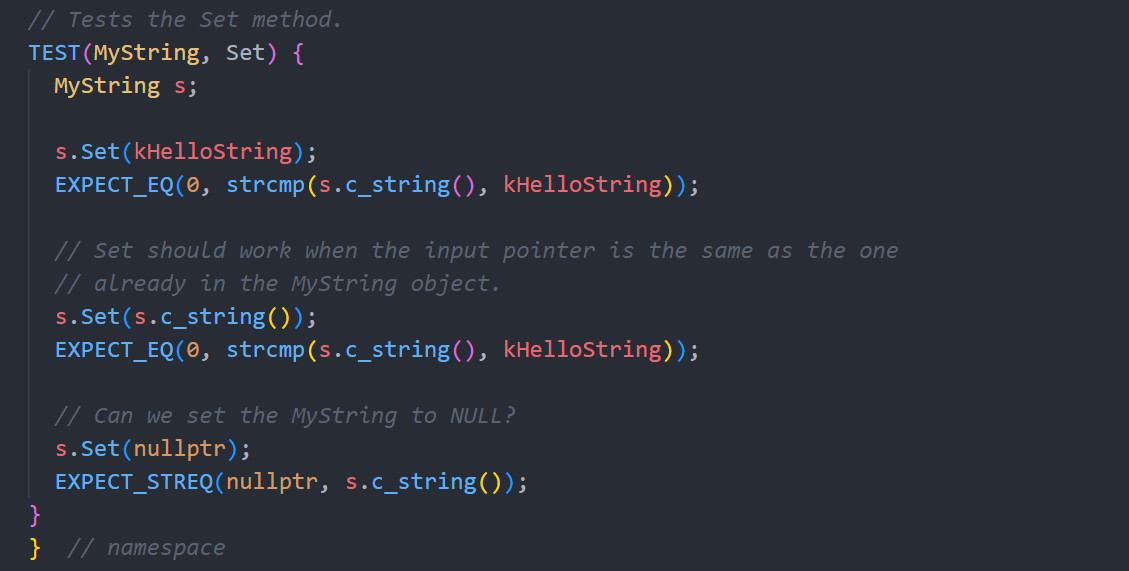
第二个用例：TEST(MyString, ConstructorFromCString)，测试 MyString(const char\* a\_c\_string) 构造函数，sizeof(kHelloString)-1和 s.Length()相等，是因为这是c类型的字符串，最后结尾是\0，sizeof计算的是分配给这个字符串的空间。



第三个用例：TEST(MyString, CopyConstructor)，测试拷贝构造函数。

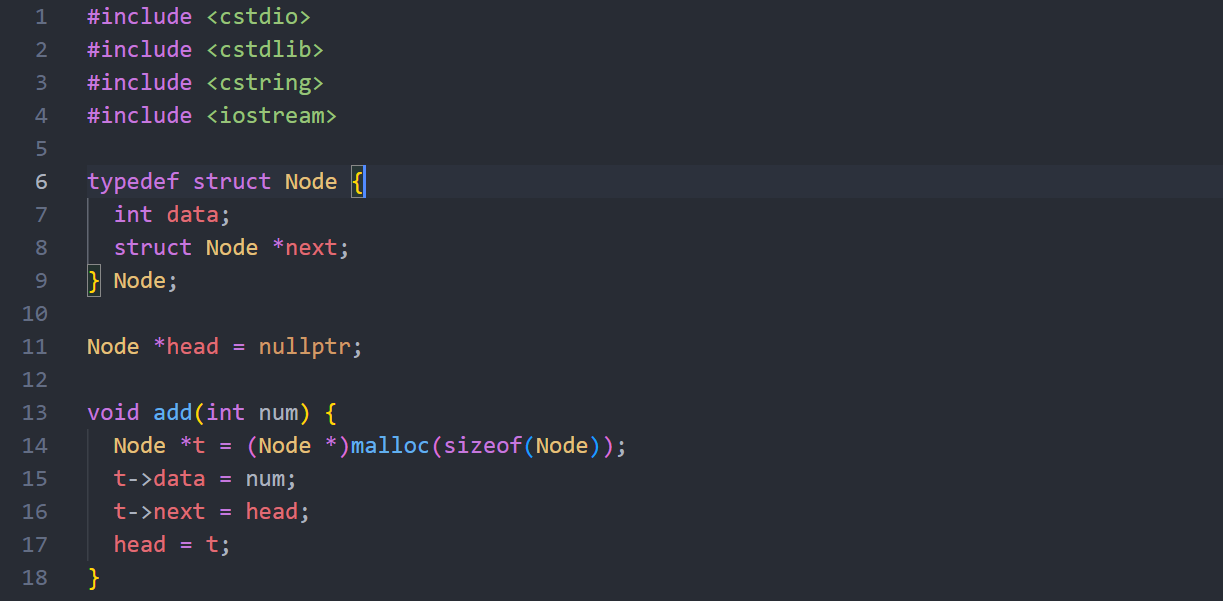


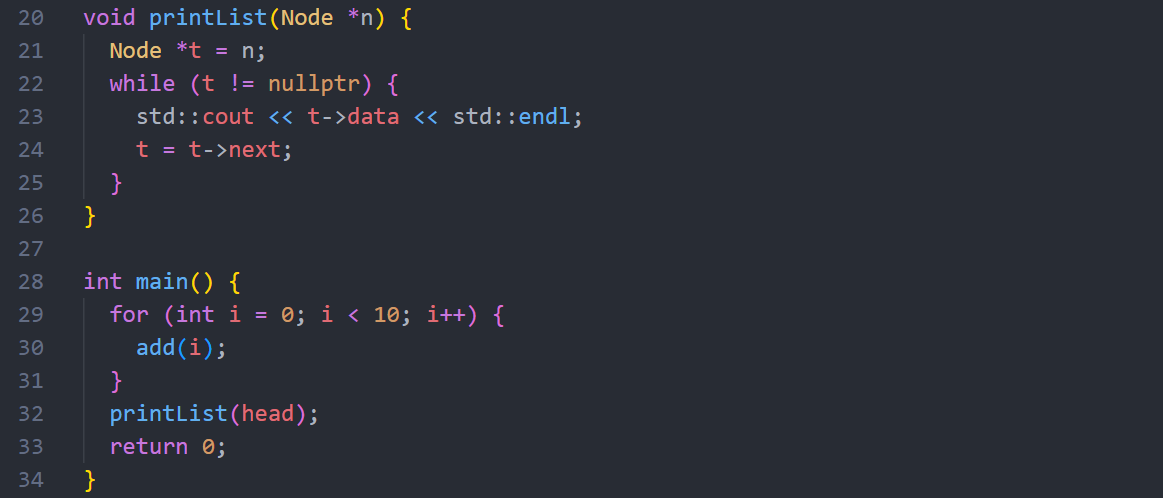
第四个用例：TEST(MyString, Set) ，测试Set成员函数。



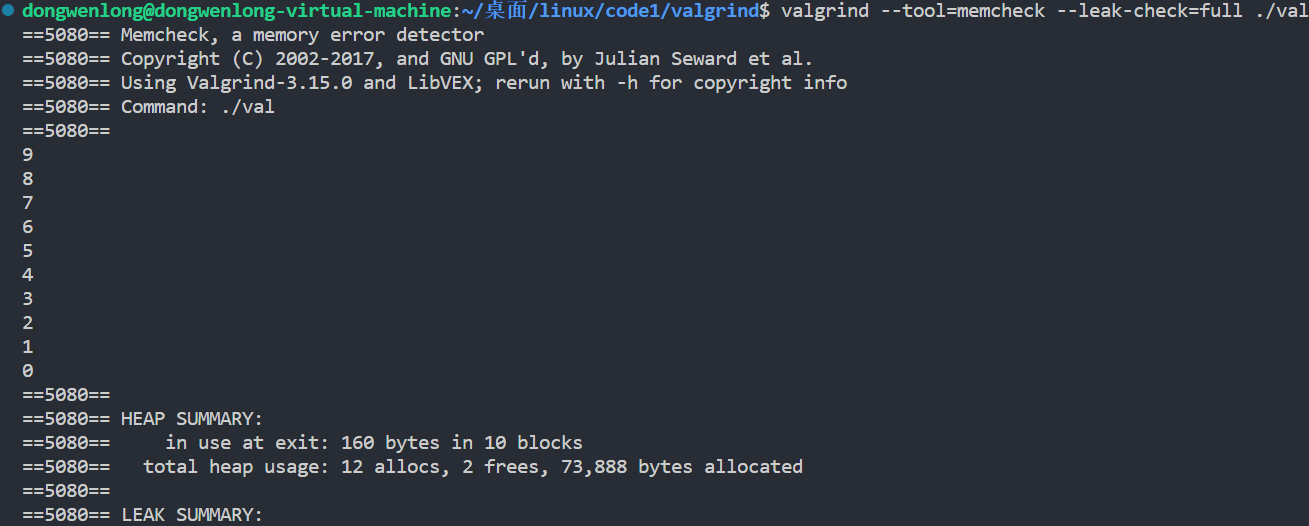
7．Valgrind使用

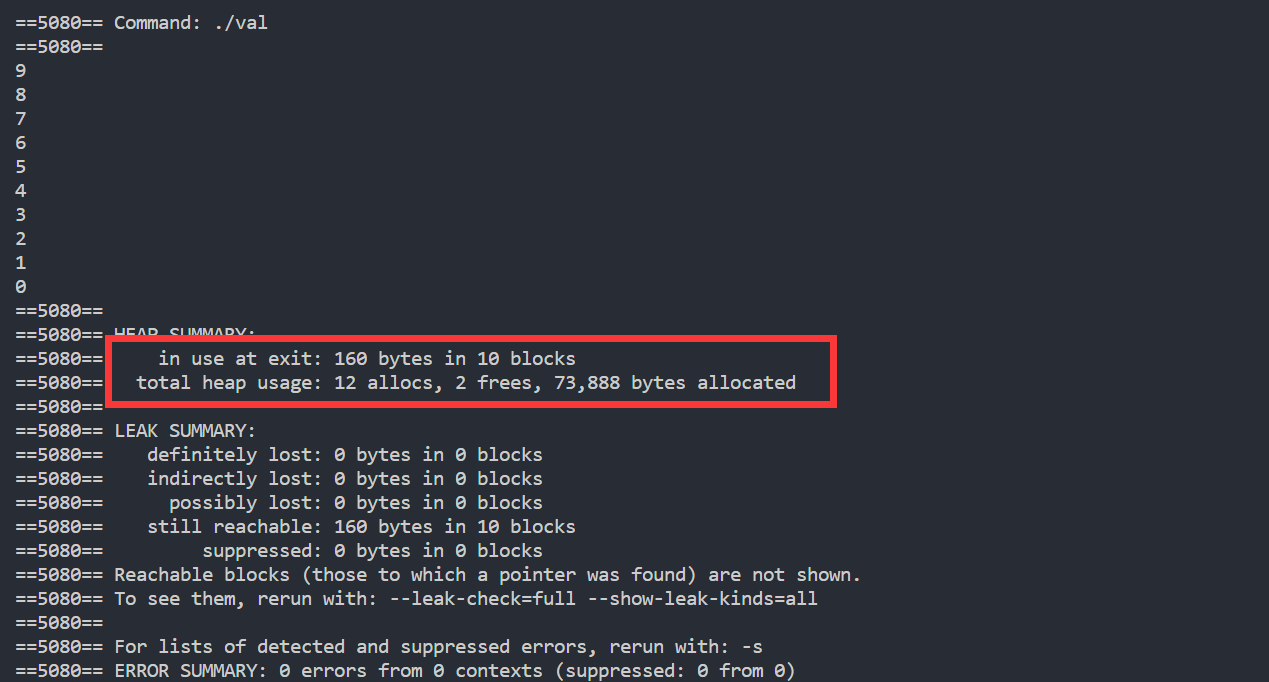
1. 源文件如下，add函数就是在链表中加一个节点，printList函数用来打印链表中的所有节点。





2. 采用valgrind工具如下：





默认的有两个调用，采用malloc有10次分配，但是每一次都不释放，从结果可以看出检测的完全正确。

**七、总结及心得体会：**

1. 完整的完成了上面工具的使用，对linux的一些工具有了更为清晰的认知。

2. linux配置环境是一个令人头大的事情。

3. 感觉自己查阅资料，上手新事物的能力有所上升。

**八、对本实验过程及方法、手段的改进建议：**

无

**报告评分：**

**指导教师签字：**