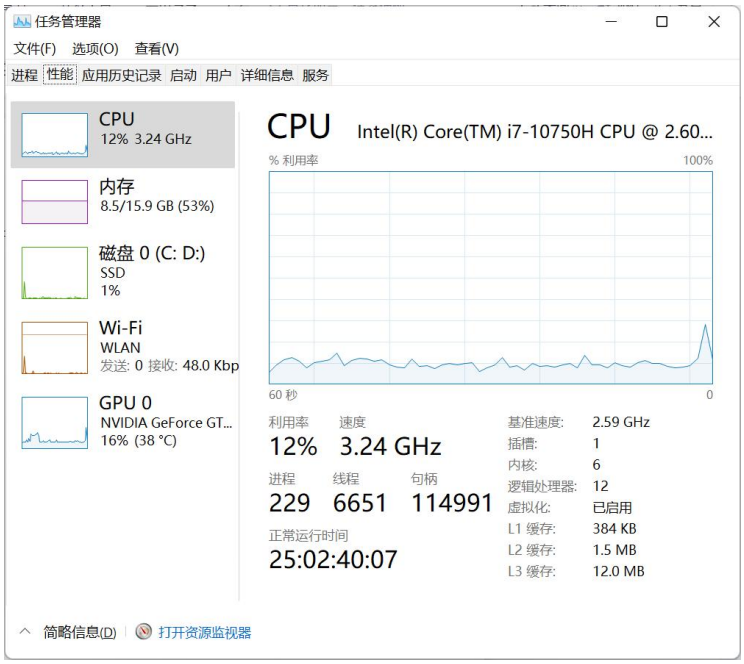
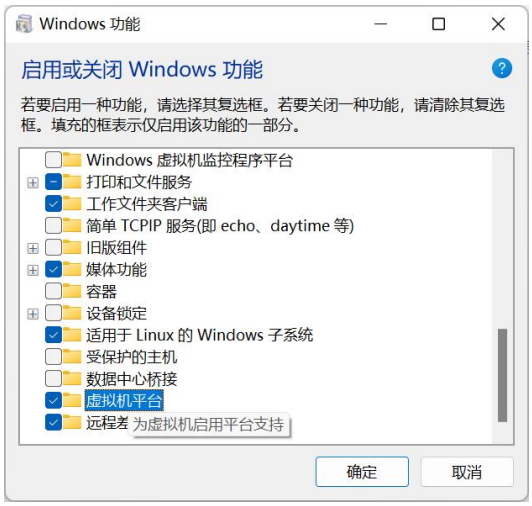
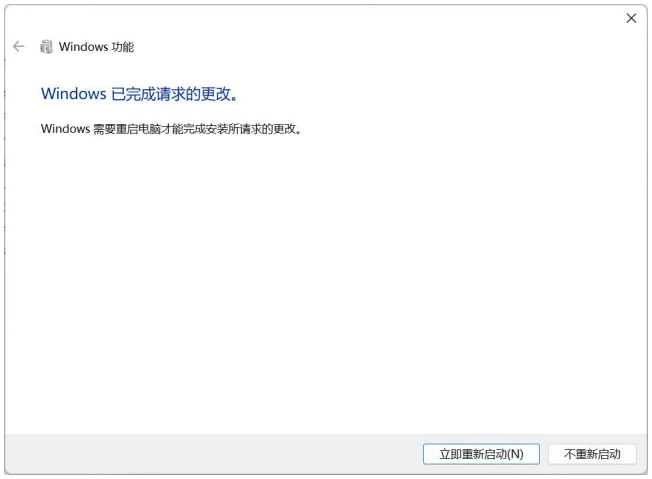
1. Ctrl+Shift+Esc 打开任务管理器确保 CPU 的虚拟化已启用



2. 控制面板->程序->程序和功能->启用或关闭 Windows 功能，打开“适用于 Linux 的 Windows 子系统”和“虚拟机平台”，点击“确定”后搜索完命令需要重启





3. Win+X 打开 windows 图标，用管理员权限打开 Windows PowerShell，输入 wsl --install

4. 输入 wsl --install -d Ubuntu 下载 Ubuntu，但是**巨慢...**

去 Microsoft Store 下载 Ubuntu 22.04.1 LTS 比较快，还可以省去 wsl 装 Ubuntu 后需要 20.04 转 22.04 的操作

5. 下载完 Ubuntu 直接打开会出现问题：

```
Something went wrong.
Please restart WSL with the following command and try again:
 wsl --shutdown
 wsl --unregister DISTRO_NAME
```

需要先下载内核：https://aka.ms/wsl2kernel ，下载完成后双击安装即可

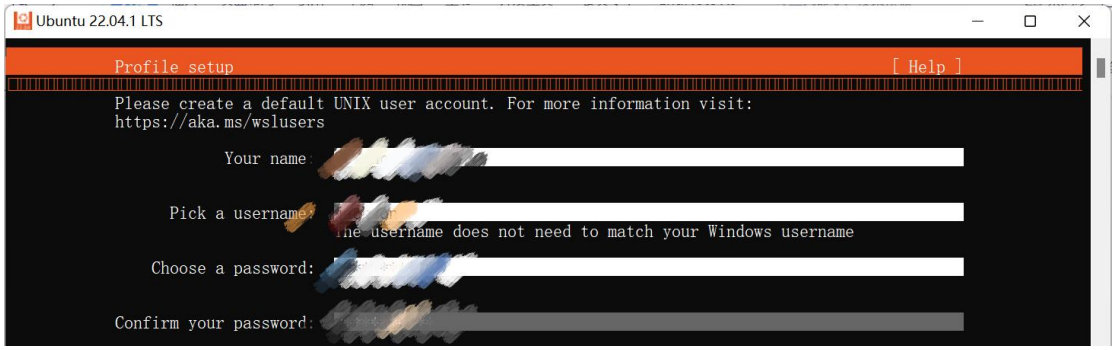
步骤 4 - 下载 Linux 内核更新包

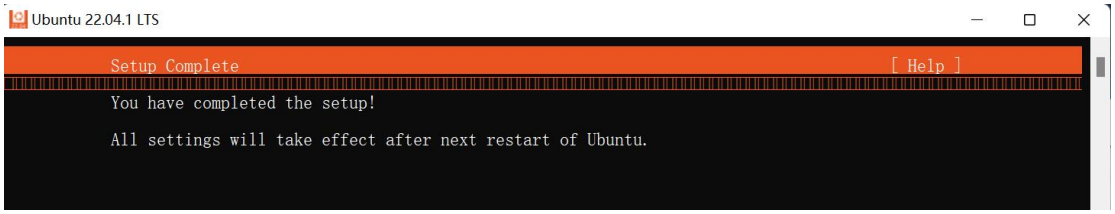
1. 下载最新包：
   - 适用于 x64 计算机的 WSL2 Linux 内核更新包

> ① **备注**
>
> 如果使用的是 ARM64 计算机，请下载 ARM64 包。 如果不确定自己计算机的类型，请打开命令提示符或 PowerShell，并输入： `systeminfo | find "System Type"`。 **Caveat:** 在非英文版 Windows 上，你可能必须修改搜索文本，对"System Type"字符串进行翻译。 你可能还需要对引号进行转义来用于 find 命令。 例如，在德语版中使用 `systeminfo | find '"Systemtyp"'`。

2. 运行上一步中下载的更新包。 （双击以运行 - 系统将提示你提供提升的权限，选择"是"以批准此安装。）

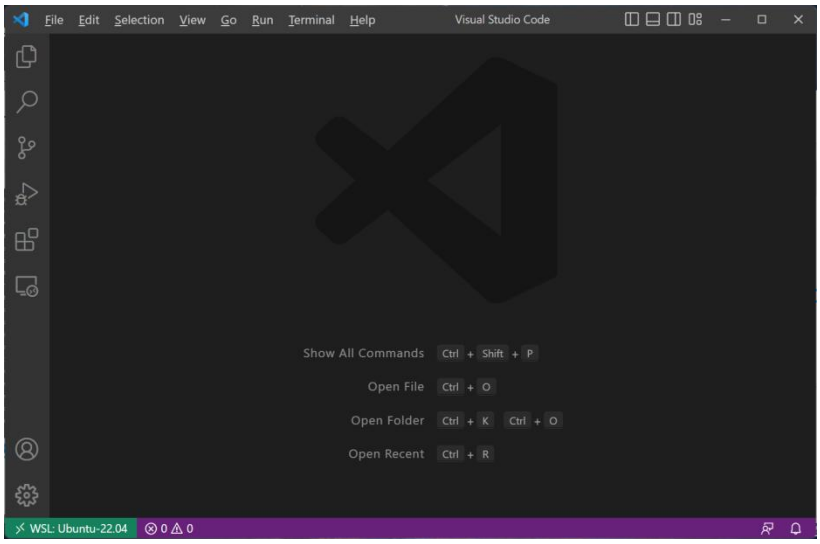
6. 安装好内核后再打开 Ubuntu 进行简单设置即可：





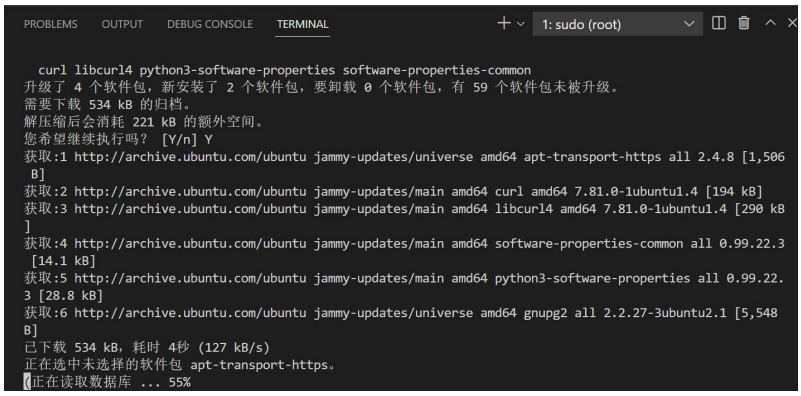
至此，已成功安装 Ubuntu，下面安装 docker

7. 打开 VS Code 在扩展中安装 WSL，安装成功后打开 remote，可以看到这是运行在 Ubuntu 上



8. 打开 terminal 执行

```
sudo apt-get install apt-transport-https ca-certificates curl gnupg2 software-properties-common
```



出现如下报错可以直接忽略：

```
Failed to retrieve available kernel versions.

Failed to check for processor microcode upgrades.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

9. 继续执行

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://mirrors.tuna.tsinghua.edu.cn/docker-ce/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
sudo apt-get update
```

```
sudo apt-get install docker-ce
```

```
Scanning processes...
Scanning processor microcode...
Scanning linux images...

Failed to retrieve available kernel versions.

Failed to check for processor microcode upgrades.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

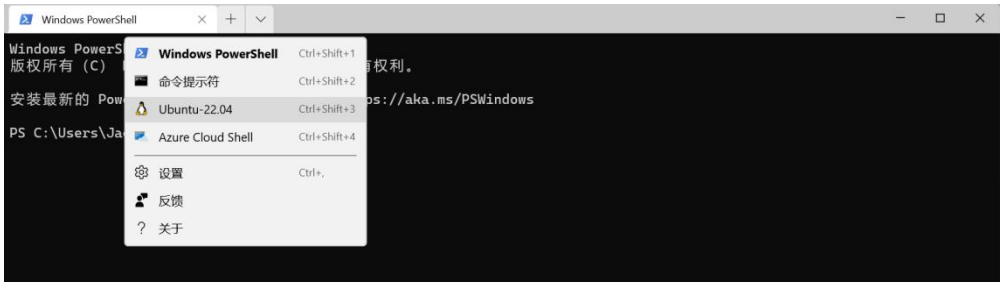
至此 docker 安装完成，但为了能够在 Linux 中不通过 root 用户（sudo）就能直接访问 Docker，以及在运行于用户权限的 VSCode 中直接使用 Docker，还需要配置 Docker 组的用户权限：

10. 配置 Docker 的用户组权

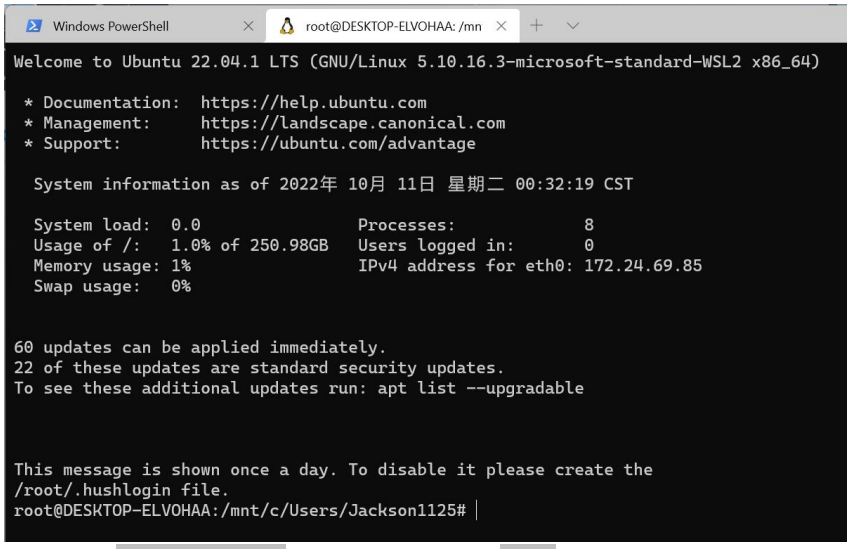
```
sudo usermod -aG docker $USER
```

```
newgrp docker
```

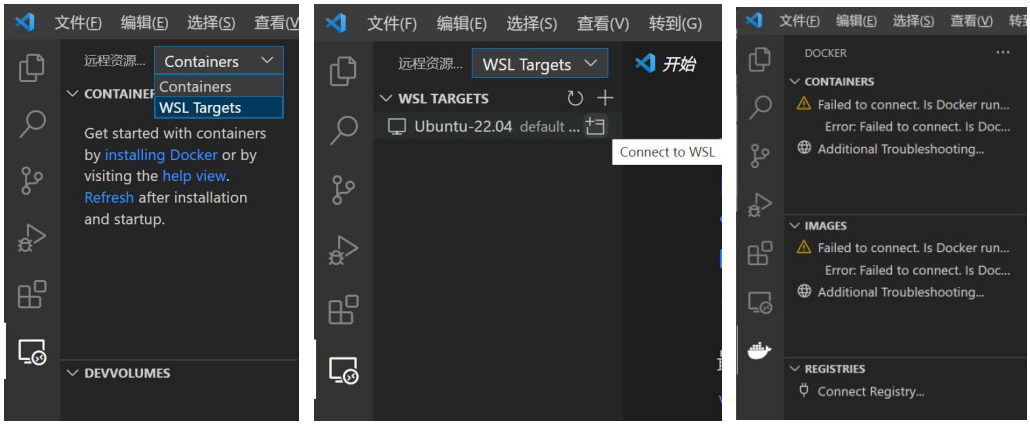
11. 在 VS Code 中安装 Docker 和 Dev Container 插件，然后关闭全部 VS Code
12. 搜索 terminal 打开的 Windows PowerShell 可以切换操作系统：



切换至 Ubuntu 的界面如下：



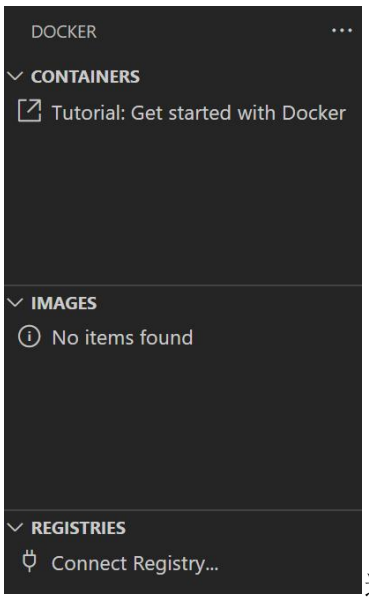
13. 输入 newgrp docker，Enter 然后输入 code，即可自行打开 VS Code
14. 连接 WSL，查看 Docker 状态



发现 Docker 没有自动打开，需要手动关闭后再打开：

15. Windows PowerShell 中输入 wsl --shutdown，然后关闭所有命令行，并重新加载 Ubuntu 上的 VS Code 窗口
16. 在 Ubuntu 上的 VS Code 终端输入 sudo service docker start， docker ps 后发现 Docker Daemon 没有启动，执行以下命令：

```
sudo apt install iptables
sudo update-alternatives --set iptables /usr/sbin/iptables-legacy
sudo update-alternatives --set ip6tables /usr/sbin/ip6tables-legacy
sudo service docker start
```



这样就是 Docker 启动了

17. 下载镜像，执行 docker pull chenyy/la32r-env，耗时较长

```
root@DESKTOP-ELVOHAA:~# docker pull chenyy/la32r-env
Using default tag: latest
latest: Pulling from chenyy/la32r-env
d19f32bd9e41: Pull complete
947bd14d4597: Pull complete
04014e0c378d: Pull complete
Digest: sha256:11ab22262972676faa4baeb21043c7a1d073d3fffc26a6ebe9a45391b
0f5da82
Status: Downloaded newer image for chenyy/la32r-env:latest
docker.io/chenyy/la32r-env:latest
```

耗时一晚上终于下载完成

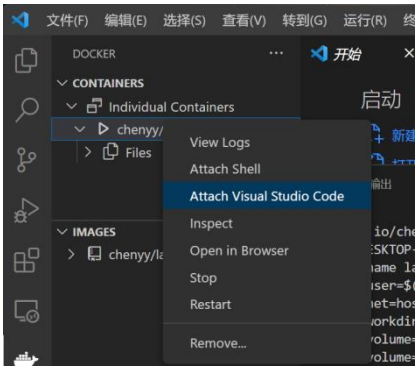
18. 创建容器，启动容器，进入 Shell

```
docker run -dit \
    --name la32r-docker \
    --user=$(id -u $USER):$(id -g $USER) \
    --net=host \
    --workdir="/home/$USER" \
    --volume="/home:/home" \
    --volume="/root:/root" \
    --volume="/mnt:/mnt" \
    --volume="/etc/group:/etc/group:ro" \
    --volume="/etc/passwd:/etc/passwd:ro" \
    --volume="/etc/shadow:/etc/shadow:ro" \
    --volume="/etc/sudoers.d:/etc/sudoers.d:ro" \
    -e LANG=en_US.UTF-8 \
    -e LANGUAGE=en_US.UTF-8 \
    -e LC_ALL=en_US.UTF-8 \
    chenyy/la32r-env

docker start la32r-docker
```

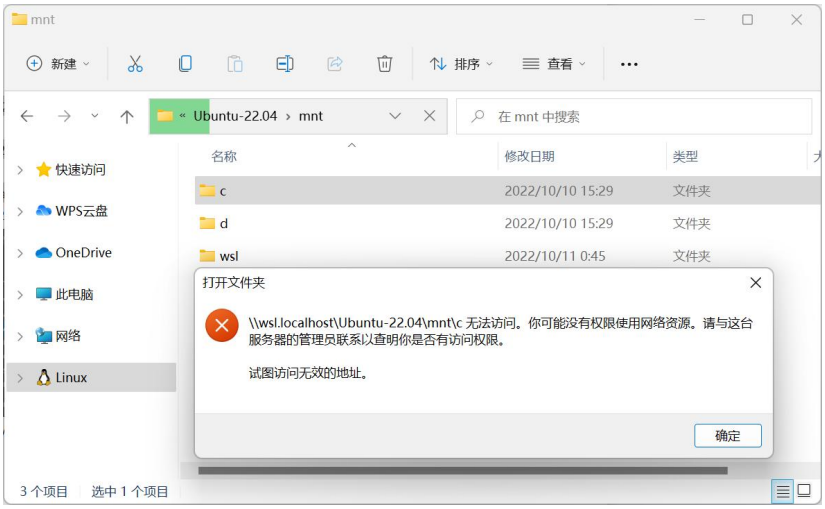
`docker exec -it la32r-docker /bin/zsh`

19. 将容器 la32r-docker 附加到 VS Code



失败的话尝试更新 VS Code

20. 打开 mnt/c/Users/用户名，创建文件夹 lab0（此处直接在文件夹里打开好像无权限）



至此环境已完全搭建好，下面编写裸机程序：

21. 按照指导书创建 main.c 和 start.S 文件，安装 LoongArch Assembly 和 C/C++插件
22. 创建 lab0.ld 和 Makefile 文件，注意将 Makefile 中空格换位 tab
23. 在终端输入 `make`，`make qemu` 出现以下提示则创建成功：

```
root@DESKTOP-ELVOHAA:/mnt/c/Users/Jackson1125/lab0# make
loongarch32r-linux-gnusf-gcc -nostdlib -T lab0.ld start.S main.c -O3 -o start.elf
root@DESKTOP-ELVOHAA:/mnt/c/Users/Jackson1125/lab0# make qemu
qemu-system-loongarch32 -M ls3a5k32 -m 32M -kernel start.elf -nographic
loongson32_init: num_nodes 1
loongson32_init: node 0 mem 0x2000000

Here is my first bare-metal machine program on LoongArch32!
```

有可能出现 Makefile 的报错，将其中 **4 个空格换成 tab** 即可

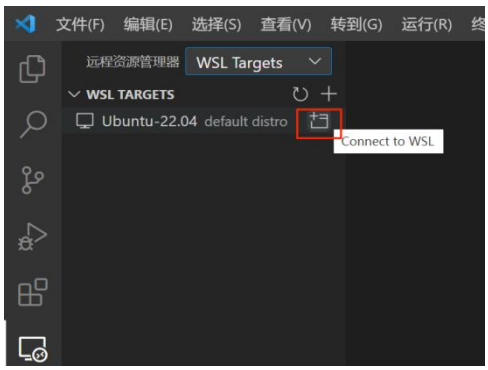
24. 先按 Ctrl+A，再按 X 可退出程序

```
QEMU: Terminated
root@DESKTOP-ELVOHAA:/mnt/c/Users/Jackson1125/lab0#
```

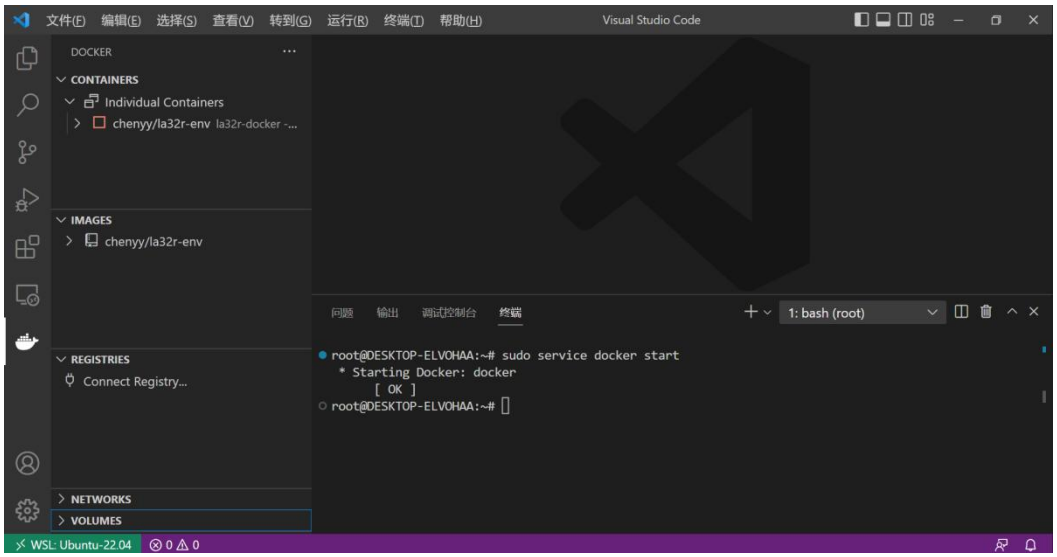
25. 复制 hello.c 文件运行即可

以后每次实验前进入虚拟机打开项目方式：

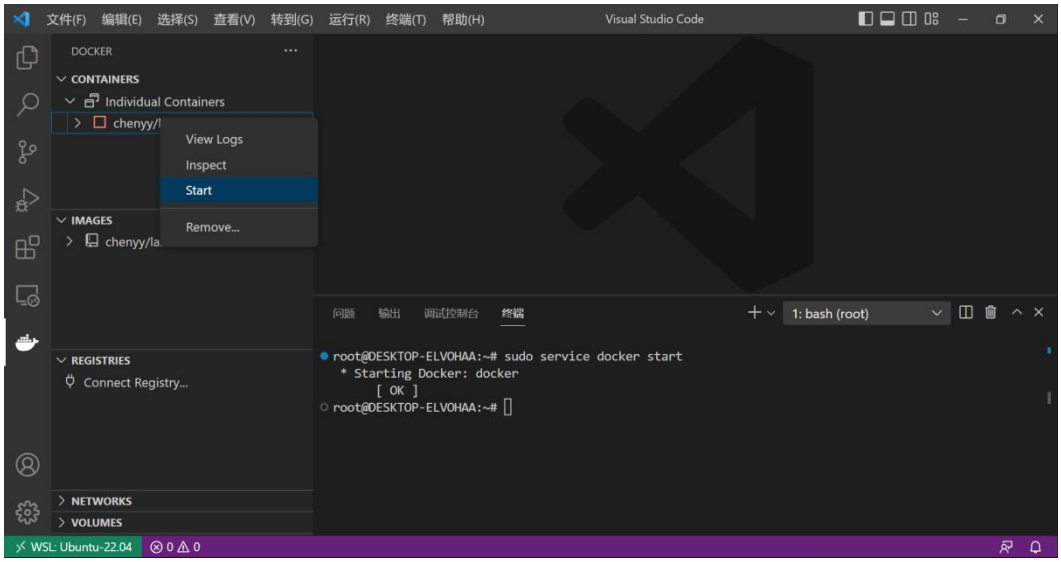
1. 打开 VS Code 远程资源管理器，选择 WSL Targest 下的 Ubuntu-22.04，点击右边图标，connect to WSL：



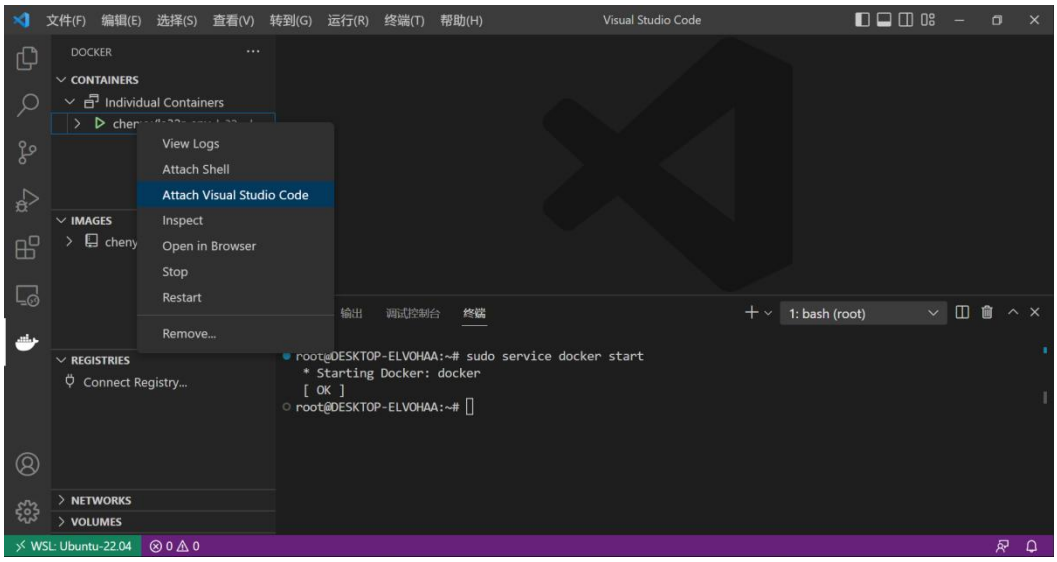
2. 这时会新弹出一个连接到 Ubuntu-22.04 上的 VS Code，此时查看 Docker 会发现未启动。在终端输入：sudo service docker start。再重新打开查看发现 Docker 已经在运行：



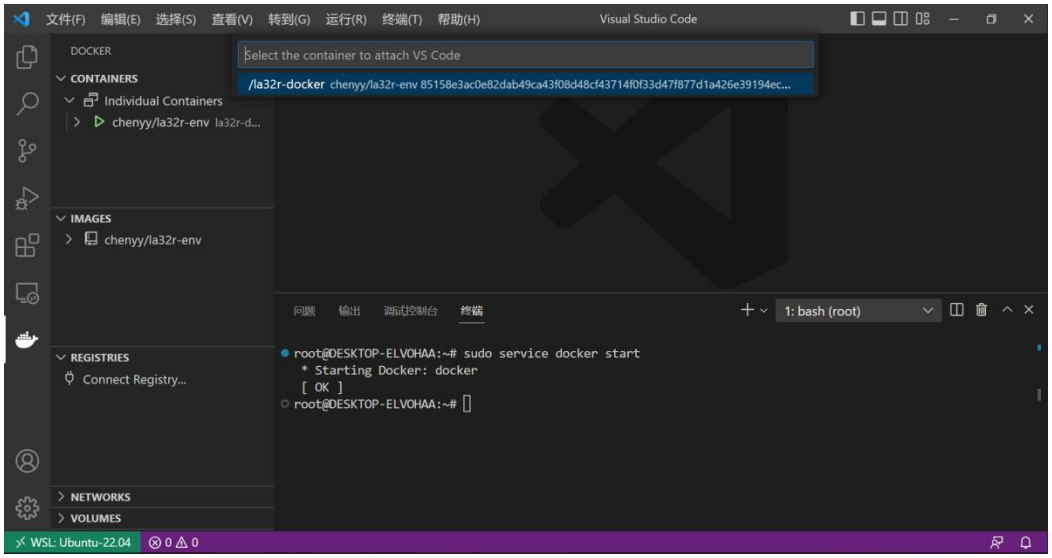
3. 右击容器选择启动：



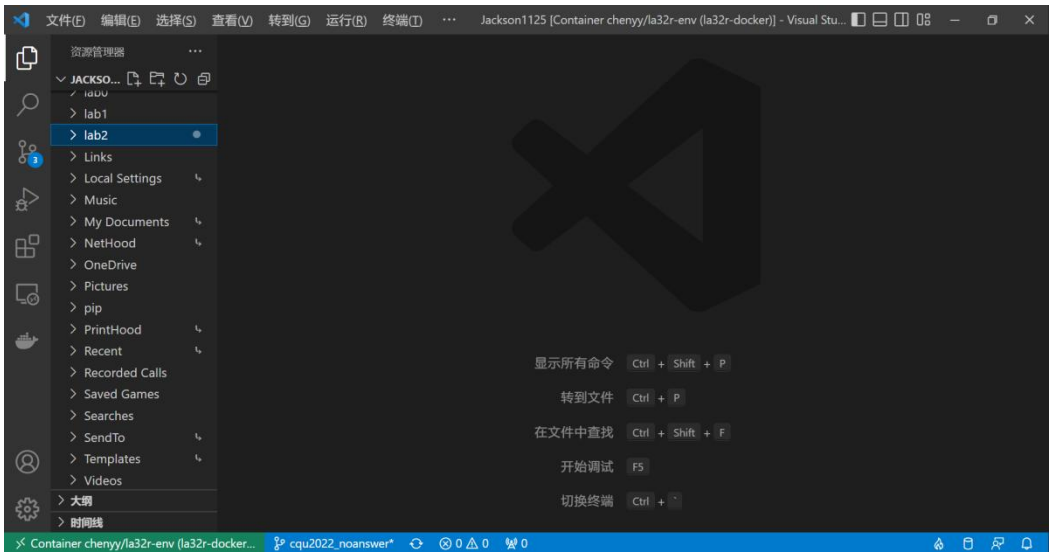
启动后容器图标变成绿色，此时再次右击可以 Attach VS Code：



选择容器后进入：



4. 此时就进入了容器内：



在终端使用命令 clone 实验包到容器中，我一般习惯新建文件夹在 clone 文件，即图中 lab1、lab2，但 clone 之前需要先进入对应的实验目录。clone 命令如下：

不含答案：git clone https://github.com/cyyself/ucore-loongarch32.git -b cqu2022_noanswer

想要含答案的再运行 cd ucore-loongarch32、git checkout cqu2022 即可。

也可以直接使用 git clone https://github.com/cyyself/ucore-loongarch32.git -b cqu2022 命令 clone 含答案版本。

```
root@DESKTOP-ELVOHAA:/mnt/c/Users/Jackson1125# cd lab3
root@DESKTOP-ELVOHAA:/mnt/c/Users/Jackson1125/lab3# git clone https://github.com/cyyself/ucore-loongarch32.git
-b cqu2022_noanswer
Cloning into 'ucore-loongarch32'...
remote: Enumerating objects: 1115, done.
remote: Counting objects: 100% (228/228), done.
remote: Compressing objects: 100% (148/148), done.
remote: Total 1115 (delta 129), reused 148 (delta 80), pack-reused 887
Receiving objects: 100% (1115/1115), 1.26 MiB | 1.21 MiB/s, done.
Resolving deltas: 100% (658/658), done.
```

5. 补全代码后，注意 Makefile 文件的状态修改，如进行实验 2 需要注释掉 LAB3、LAB4 两行：

```
LAB1    := -DLAB1_EX2 -DLAB1_EX3 #-D_SHOW_100_TICKS -D_SHOW_SERIAL_INPUT
LAB2    := -DLAB2_EX1 -DLAB2_EX2 -DLAB2_EX3
#LAB3   := -DLAB3_EX1 -DLAB3_EX2
#LAB4   := -DLAB4_EX1 -DLAB4_EX2
```

6. 在终端执行以下命令即可编译运行实验：

```
make clean    # 清空编译产物
make -j 16    # 编译
make qemu     # 运行 qemu
```

要想结束实验，按 Ctrl+A，再按 X 可退出程序