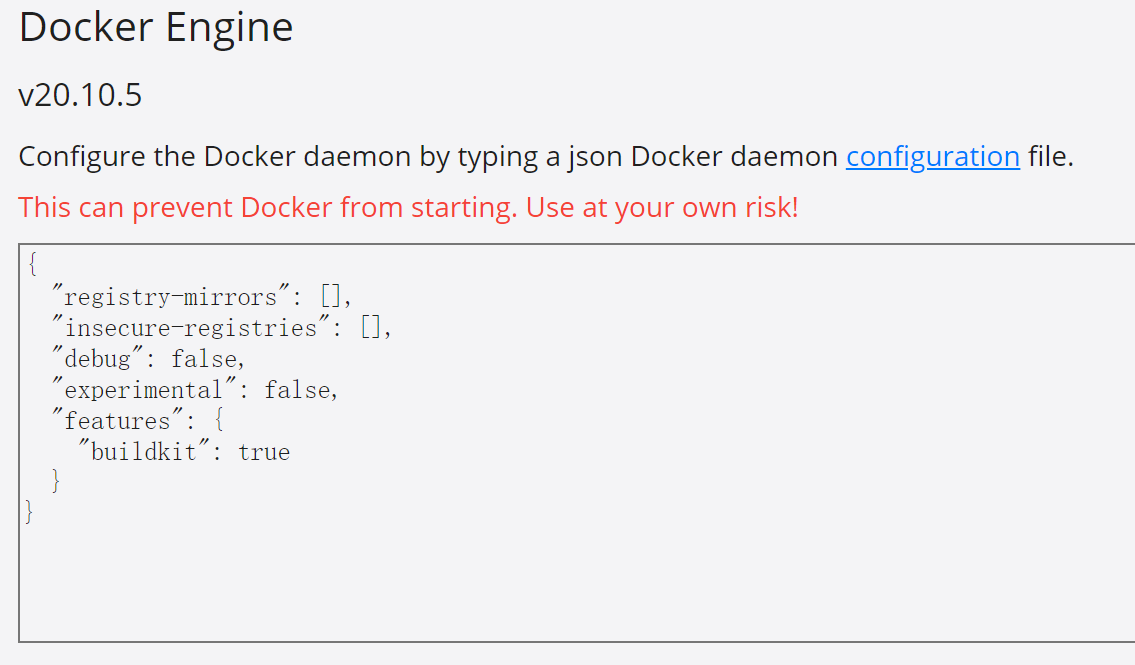
1.安装docker

2.docker未修改的配置文件：



即：

{

"registry-mirrors": [],

"insecure-registries": [],

"debug": false,

"experimental": false,

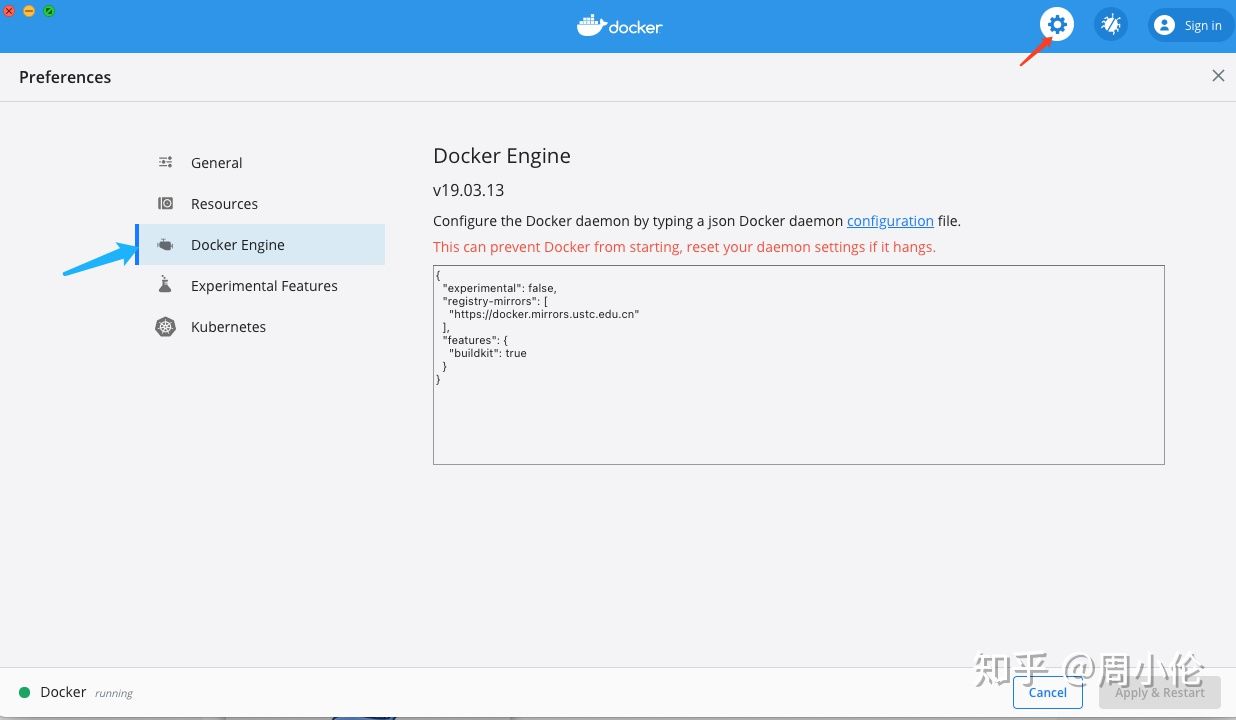
"features": {

"buildkit": true

}

}

修改为：



即：

{

"experimental": false,

"registry-mirrors": [

"https://docker.mirrors.ustc.edu.cn"

],

"features": {

"buildkit": true

}

}

即将配置镜像换为国内镜像。

3.进入项目在本机的文件夹，我的是：C:\Users\frey\Documents\GitHub\DataBase

4.在该文件夹下打开命令行，键入命令：

docker build . -t bustub

开始为本项目在docker中配置环境

5.然后键入命令：

docker create -t -i --name bustub -v $(pwd):/bustub bustub bash

其中$(pwd)换为项目在本机的文件夹，比如我的命令应为：

docker create -t -i --name bustub -v C:\Users\frey\Documents\GitHub\DataBase:/bustub bustub bash

这样在docker中本机文件夹被命名为bustub文件夹

6.执行命令：

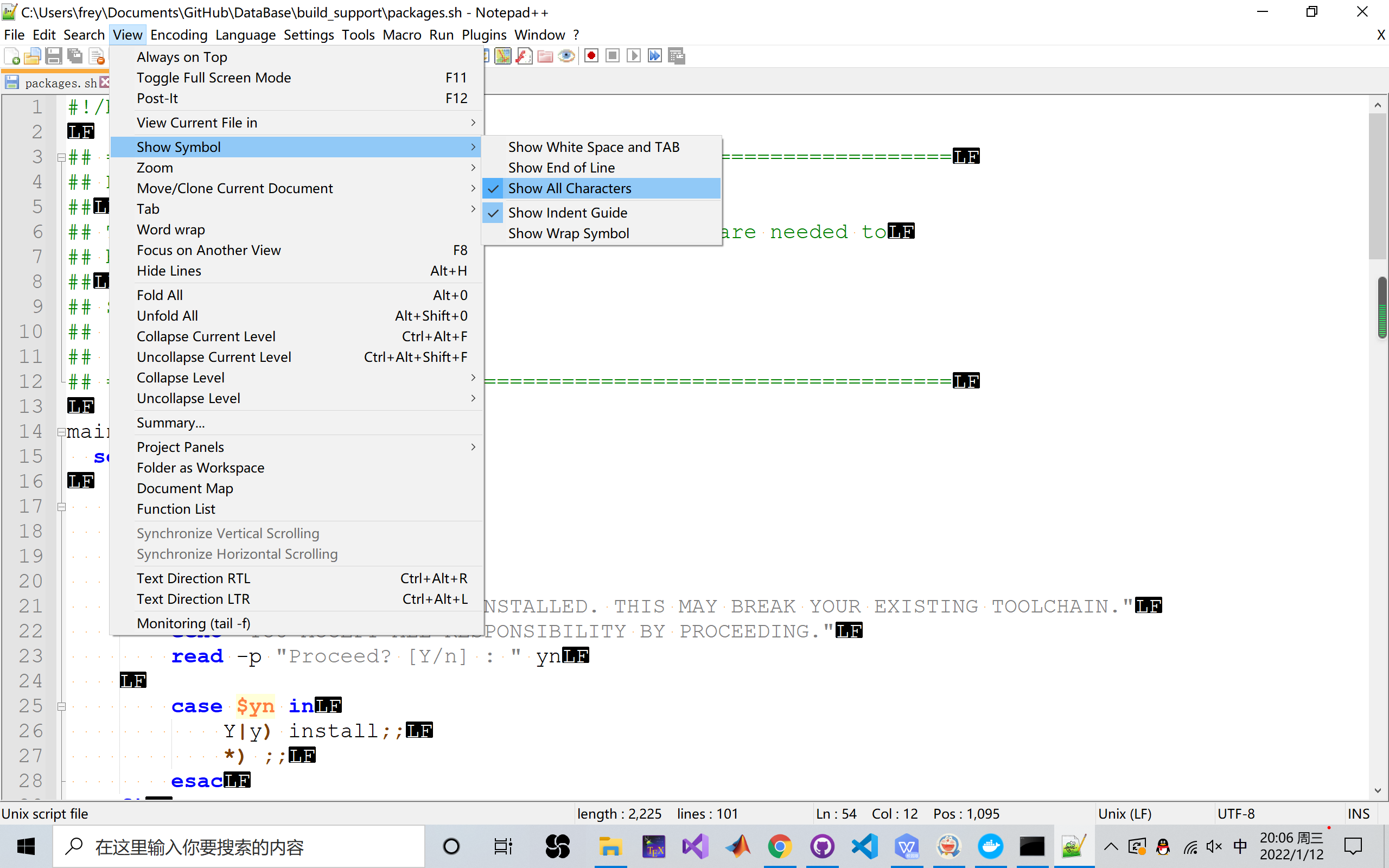
docker start -a -i bustub

这个时候就可以进入我们的容器(docker)里。这个容器就是我们的实验环境。

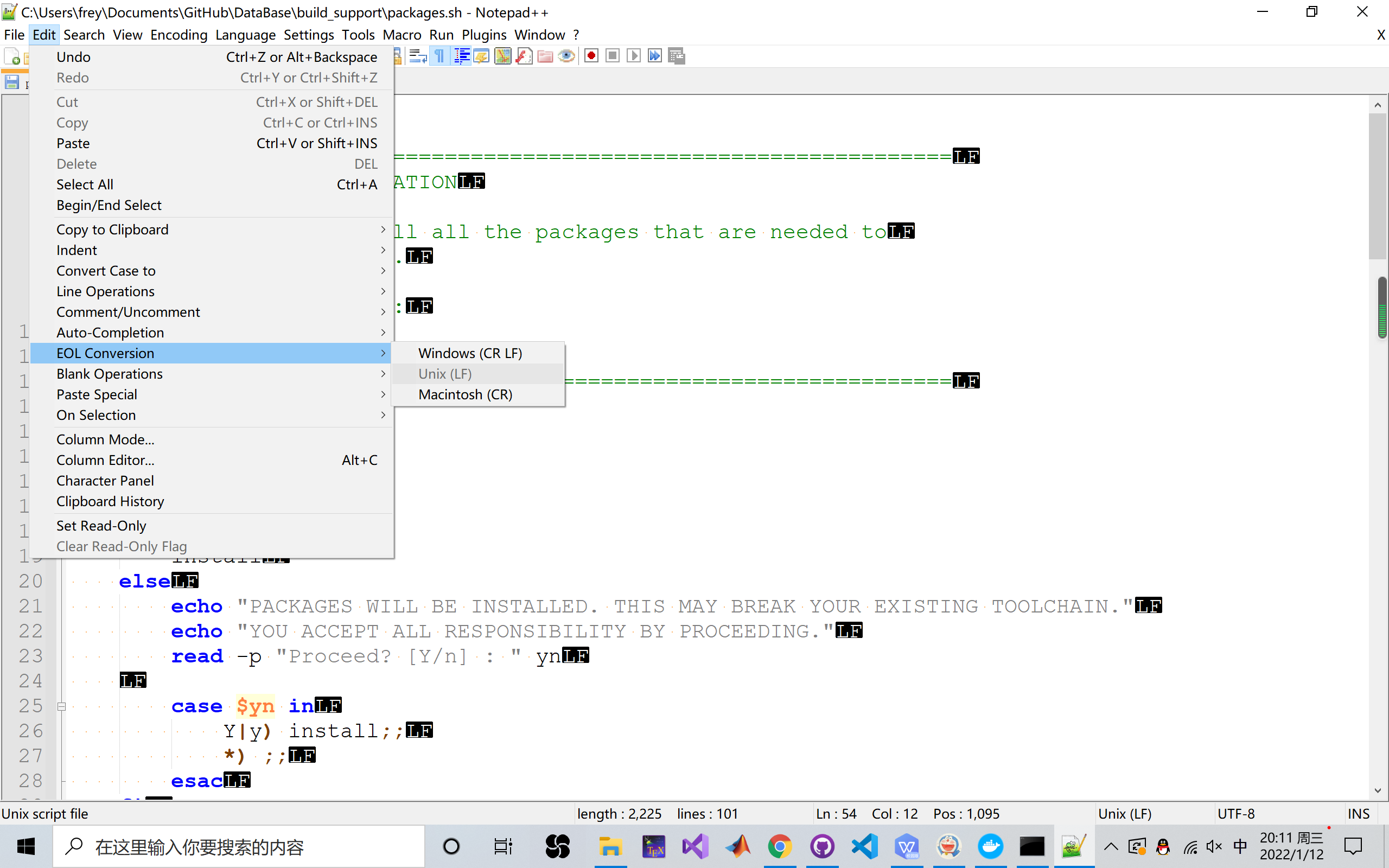
7.执行：cd bustub 进入bustub文件夹

8.然后，将build\_support文件夹下的packages.sh文件用notepad++打开，

按如下方式设置：show All characters,这时会看到结尾符(LF这类的)



注意Unix和Windows中结尾符是不同的，若packages.sh文件的结尾符是Windows样式(CR LF)的，在Docker的Unix环境中执行会出错，应改为Unix结尾符，如下所示更改：



9.按顺序执行以下指令：

mkdir build

cd build

cmake ..

make

编译并构建系统。

10.测试系统构建是否成功：

cd build

make check-tests

若构建成功，则结果如下所示：100% tests passed

