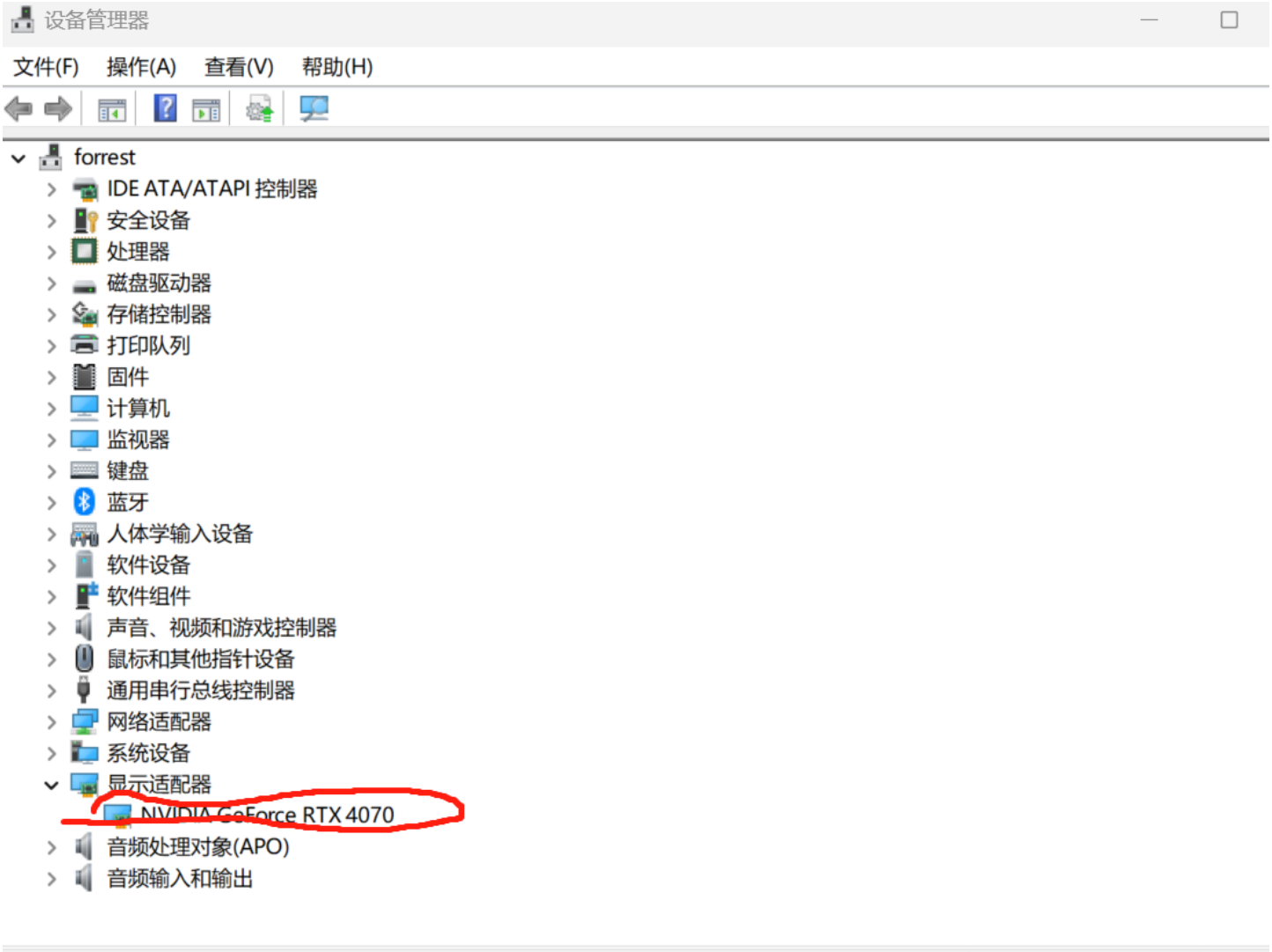
本地计算机安装mpicc和nvcc

详细安装过程参考https://zhuanlan.zhihu.com/p/644960969

本地计算机配置服务器

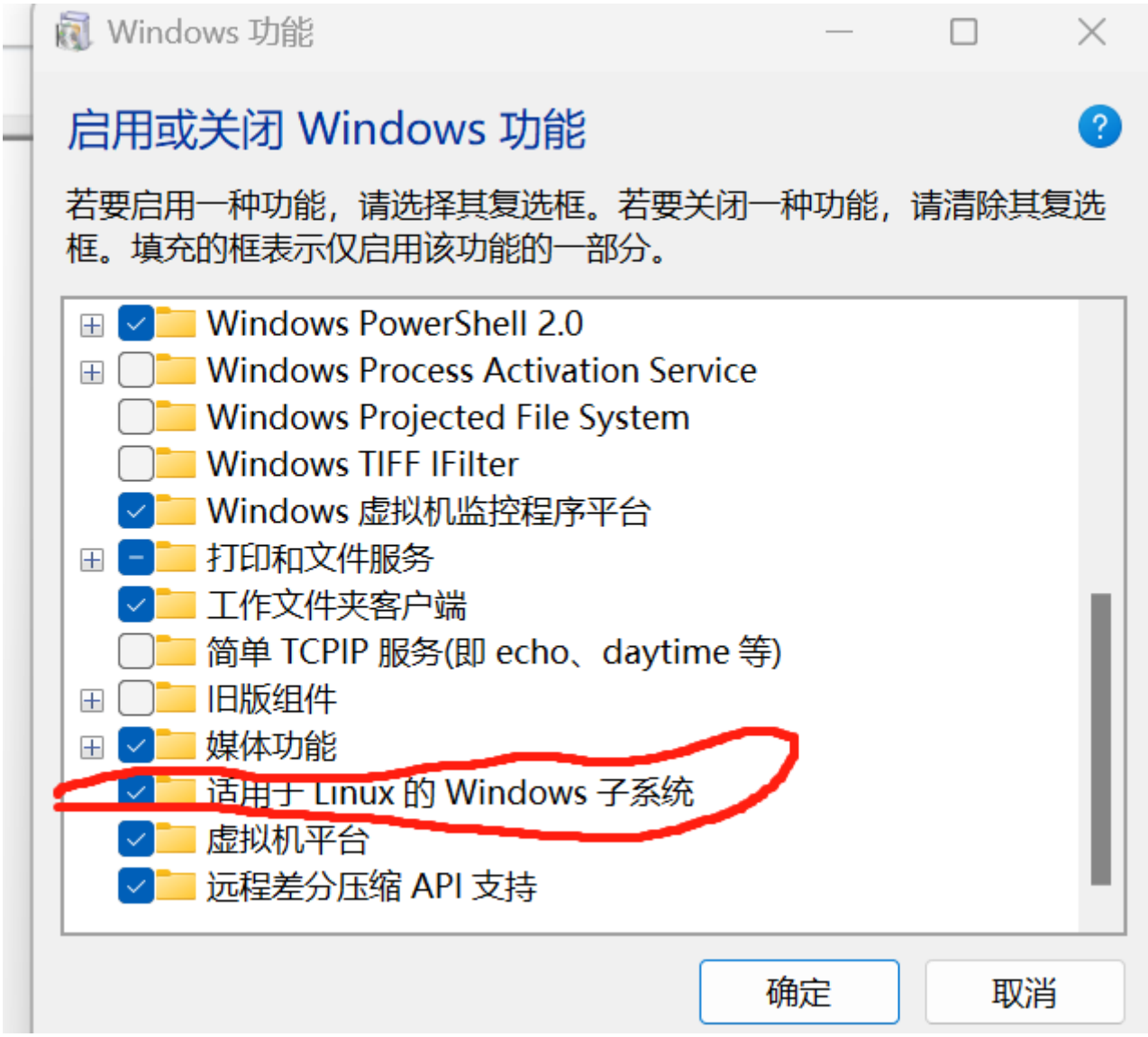
本地计算机默认为windows系统，并且**一定要有显卡**，关于本地计算机是否有显卡可以通过：搜索**控制面板——>硬件与声音——>设备和打印机——>设备管理器——>显示适配器**查询



如上图所示，本人显卡是英伟达公司的GeForce RTX 4070

wsl安装和启动

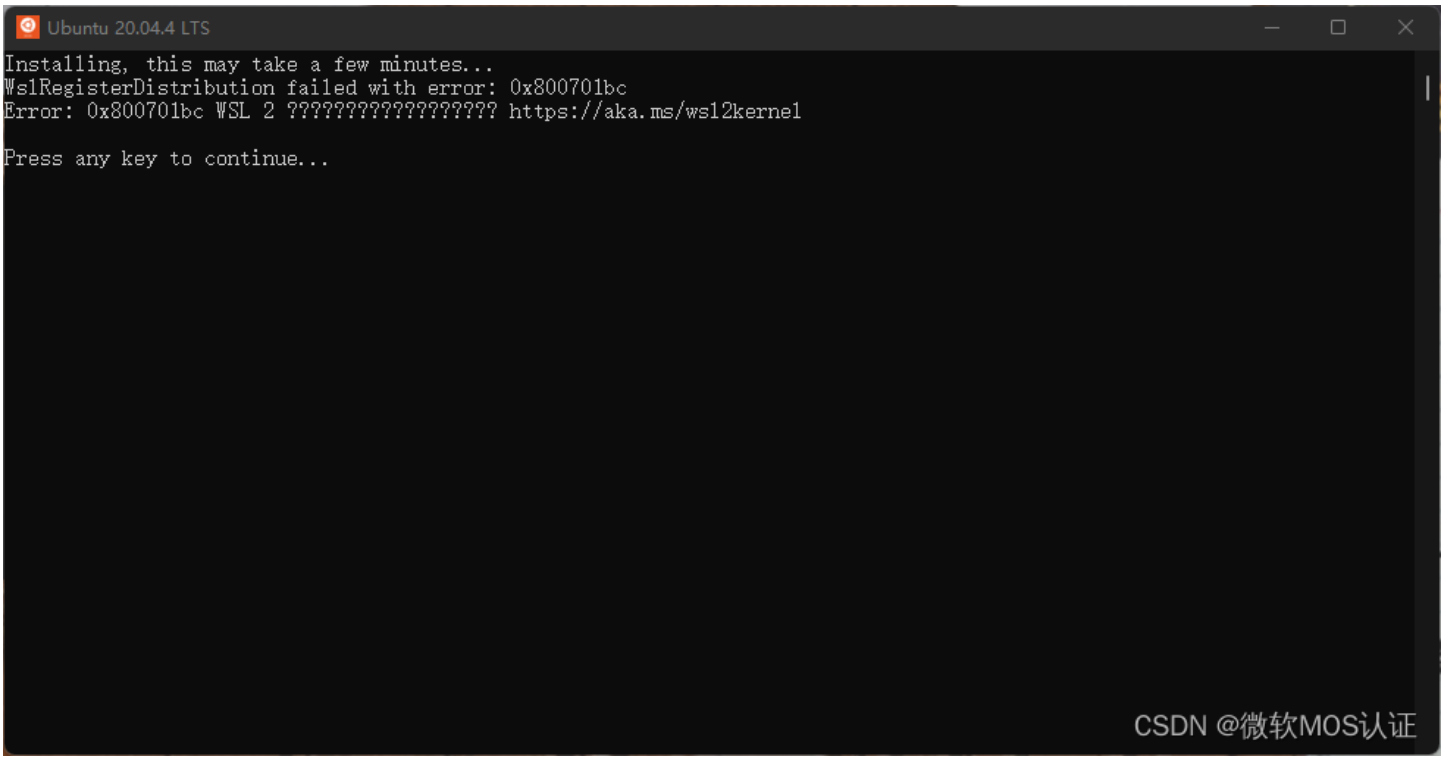
控制面板->程序->启用或关闭Windows功能—>适用于linux的windows子系统



一定要打勾，本人一般会在这里把跟虚拟机有关，linux有关的选项也勾起来，这个时候一般会要求电脑重启。

ubuntu下载安装

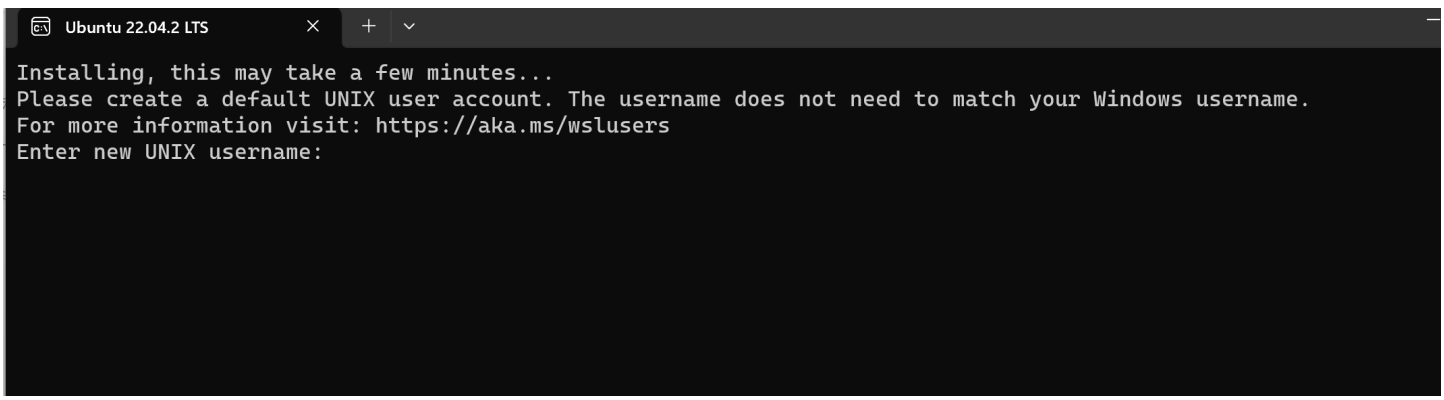
这里本人直接在电脑的应用市场搜索ubuntu，本人选取的是22.04最高版本，安装成功以后打开创建自己的账号和密码即可。



安装好了ubuntu以后，我们尝试打开ubuntu，但是很可能会遇到上面这种报错：Installing, this may take a few minutes... WslRegisterDistribution failed with error: 0x800701bc Error: 0x800701bc WSL 2 ??????????????????? https://aka.ms/wsl2kernel Press any key to continue...此时会卡住，不管摁哪个键都会直接退出，此时上网查找原因，本人参考了这个博客 https://blog.csdn.net/microsoft_mos/article/details/123627295来解决问题，发现需要载安装适用于 x64 计算机的最新 WSL2 Linux 内核更新包

下载链接：https://wslstorestorage.blob.core.windows.net/wslblob/wsl_update_x64.msi

下载这个内核以后直接双击安装即可，安装结束以后再次打开ubuntu，不出意外的话，我们会进入下面这个界面：

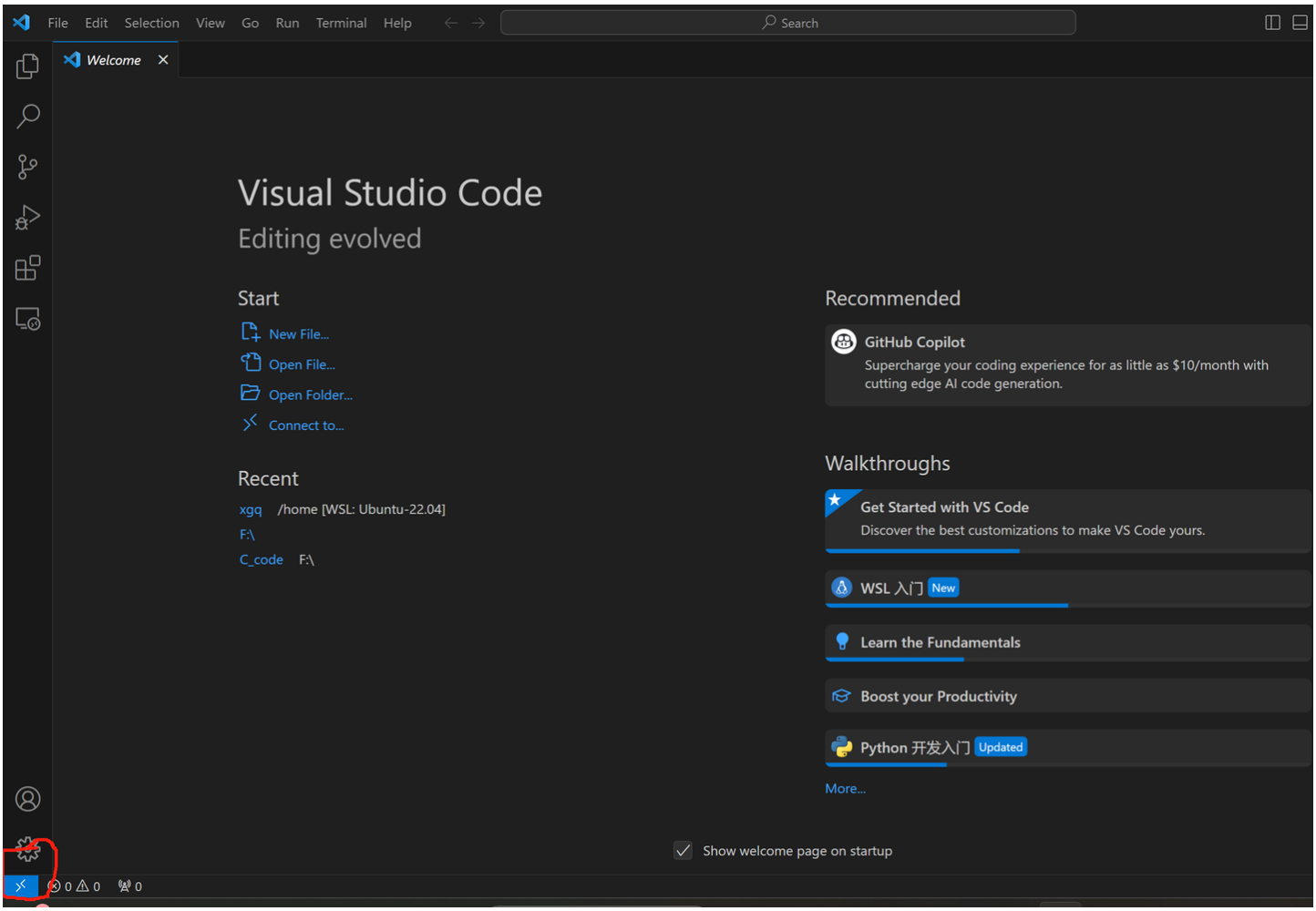


此时我们可以自己创建一个账户和密码

```
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: [redacted]
adduser: Please enter a username matching the regular expression configured
via the NAME_REGEX[_SYSTEM] configuration variable.  Use the `--force-badname'
option to relax this check or reconfigure NAME_REGEX.
Enter new UNIX username: [redacted]
New password:
Retype new password:
passwd: password updated successfully
Installation successful!
适用于 Linux 的 Windows 子系统现已在 Microsoft Store!
你可以通过运行 "wsl.exe --update" 进行升级 或通过访问 https://aka.ms/wslstorepage
从Microsoft Store安装 WSL 将提供最新的 WSL 更新， faster.
有关详细信息，请访问 https://aka.ms/wslstoreinfo<
>

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.10.16.3-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This message is shown once a day. To disable it please create the
/home/[redacted]/.hushlogin file.
[redacted]:~$ ls
```

创建账户名字的时候发现不能使用大写字母创建，所以后面直接改用小写字母创建，自己设置好密码以后就会看见上面这个界面，到此我们就在自己本地计算机创建了一个服务器，在这个服务器上使用的是linux系统，我们可以通过这个平台来自学linux操作命令，比如说我想查看本地计算机的显卡配置，使用命令nvidia-smi。

上面我们只是创建了服务器，但是我们想上传文件，结果我们根本不知道ubuntu把文件放哪了，也不知道如何操作，这个时候虽然我们可以百度搜索，但是比较麻烦。这个时候我会建议安装vscode，网上搜索vscode的教程，安装好vscode以后，打开vscode我们点击左下角那个蓝色的地方，可以打开选项，告诉我们可以连接WSL，点击即可进入我们刚刚自己创建的本地服务器，接下来就可以开始我们的高性能计算编程学习了。



进入服务器以后，我们通过nvidia-smi查看自己计算机的显卡情况，如果本地计算机有显卡，就打打印出显卡的信息，如果没有显卡，这里可能会提示要安装别的（但其实没有显卡安装了也没用，也无法安装驱动），如果没有显卡，就无法跑CUDA代码，只能测试mpi和openmp代码，因此下面我们先介绍mpi和openmp的入门。

gcc,mpicc编译器安装

在正式介绍高性能计算之前，我们需要安装相应的软件和库，高性能计算主要使用C和C++这类编程语言，我们需要安装gcc来编译C语言代码，对于MPI这类并行语言，我们需要安装mpicc来编译，对于CUDA代码，我们需要安装nvcc来编译，这些编译器的安装下面做一个统一的介绍。

gcc，g++安装

我们进入自己搭建的服务器，首先可以通过gcc --version来查看自己本地服务器是否自动安装了gcc，如果本地服务器自动安装了gcc，那么应该会打印出相应的gcc版本，如果服务器没有安装gcc，那么就会出现下面界面来提示我们安装gcc。下面界面是提示我们使用sudo apt install gcc安装，这个sudo表示管理员，我们自己搭建的服务器，我们肯定是管理员，但是对于一般的服务器，我们没有sudo权限，此时需要我们联系服务器对应的管理员，找他来安装gcc。

```
x[redacted]g:~$ gcc --version
Command 'gcc' not found, but can be installed with:
sudo apt install gcc
x[redacted]:~$ sudo apt install gcc
[sudo] password for x[redacted]:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  cpp cpp-11 fontconfig-config fonts-dejavu-core gcc-11 gcc-11-base libasan6 libatomic1 libc-dev-bin libc-devtools
  libc6-dev libcc1-0 libcrypt-dev libdeflate0 libfontconfig1 libfreetype6 libgcc-11-dev libgd3 libgomp1 libisl23
  libitm1 libjbig0 libjpeg-turbo8 libjpeg8 liblsan0 libmpc3 libnsl-dev libquadmath0 libtiff5 libtirpc-dev libtsan0
  libubsan1 libwebp7 libxpm4 linux-libc-dev manpages-dev rpcsvc-proto
Suggested packages:
  cpp-doc gcc-11-locales gcc-multilib make autoconf automake libtool flex bison gdb gcc-doc gcc-11-multilib gcc-11-doc
  glibc-doc libgd-tools
The following NEW packages will be installed:
  cpp cpp-11 fontconfig-config fonts-dejavu-core gcc gcc-11 gcc-11-base libasan6 libatomic1 libc-dev-bin libc-devtools
  libc6-dev libcc1-0 libcrypt-dev libdeflate0 libfontconfig1 libfreetype6 libgcc-11-dev libgd3 libgomp1 libisl23
  libitm1 libjbig0 libjpeg-turbo8 libjpeg8 liblsan0 libmpc3 libnsl-dev libquadmath0 libtiff5 libtirpc-dev libtsan0
  libubsan1 libwebp7 libxpm4 linux-libc-dev manpages-dev rpcsvc-proto
```

不过可惜，根据这种提示我们尝试安装gcc，结果安装失败，本人安装的时候报错如下：

```
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/gcc-11/libasan6_11.3.0-1ubuntu1%7e22.04_amd64.deb  404
 Not Found [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/gcc-12/liblsan0_12.1.0-2ubuntu1%7e22.04_amd64.deb  404
 Not Found [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/gcc-11/libtsan0_11.3.0-1ubuntu1%7e22.04_amd64.deb  404
 Not Found [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/gcc-12/libubsan1_12.1.0-2ubuntu1%7e22.04_amd64.deb  404
  Not Found [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/gcc-12/libquadmath0_12.1.0-2ubuntu1%7e22.04_amd64.deb
404  Not Found [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/gcc-11/libgcc-11-dev_11.3.0-1ubuntu1%7e22.04_amd64.deb
 404  Not Found [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/gcc-11/gcc-11_11.3.0-1ubuntu1%7e22.04_amd64.deb  404  N
ot Found [IP: 91.189.91.81 80]
E: Failed to fetch http://archive.ubuntu.com/ubuntu/pool/main/g/glibc/libc-dev-bin_2.35-0ubuntu3.1_amd64.deb  404  Not F
ound [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/f/freetype/libfreetype6_2.11.1%2bdfsg-1ubuntu0.1_amd64.de
b  404  Not Found [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/t/tiff/libtiff5_4.3.0-6ubuntu0.4_amd64.deb  404  Not Foun
d [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/libx/libxpm/libxpm4_3.5.12-1ubuntu0.22.04.1_amd64.deb  40
4  Not Found [IP: 91.189.91.81 80]
E: Failed to fetch http://archive.ubuntu.com/ubuntu/pool/main/g/glibc/libc-devtools_2.35-0ubuntu3.1_amd64.deb  404  Not
Found [IP: 91.189.91.81 80]
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/l/linux/linux-libc-dev_5.15.0-71.78_amd64.deb  404  Not F
ound [IP: 91.189.91.81 80]
E: Failed to fetch http://archive.ubuntu.com/ubuntu/pool/main/g/glibc/libc6-dev_2.35-0ubuntu3.1_amd64.deb  404  Not Foun
d [IP: 91.189.91.81 80]
E: Unable to fetch some archives, maybe run apt-get update or try with --fix-missing?
```

经过网上查找资料https://wenku.csdn.net/answer/341d71d7beaa58b569f1028b35c0b9c1，发现没有事先更新，因此根据提示，我们先做sudo apt-get update，结束以后，再次使用sudo apt-get install gcc命令，此时保持网络畅通，结束以后，使用gcc --version确认gcc安装成功如下：

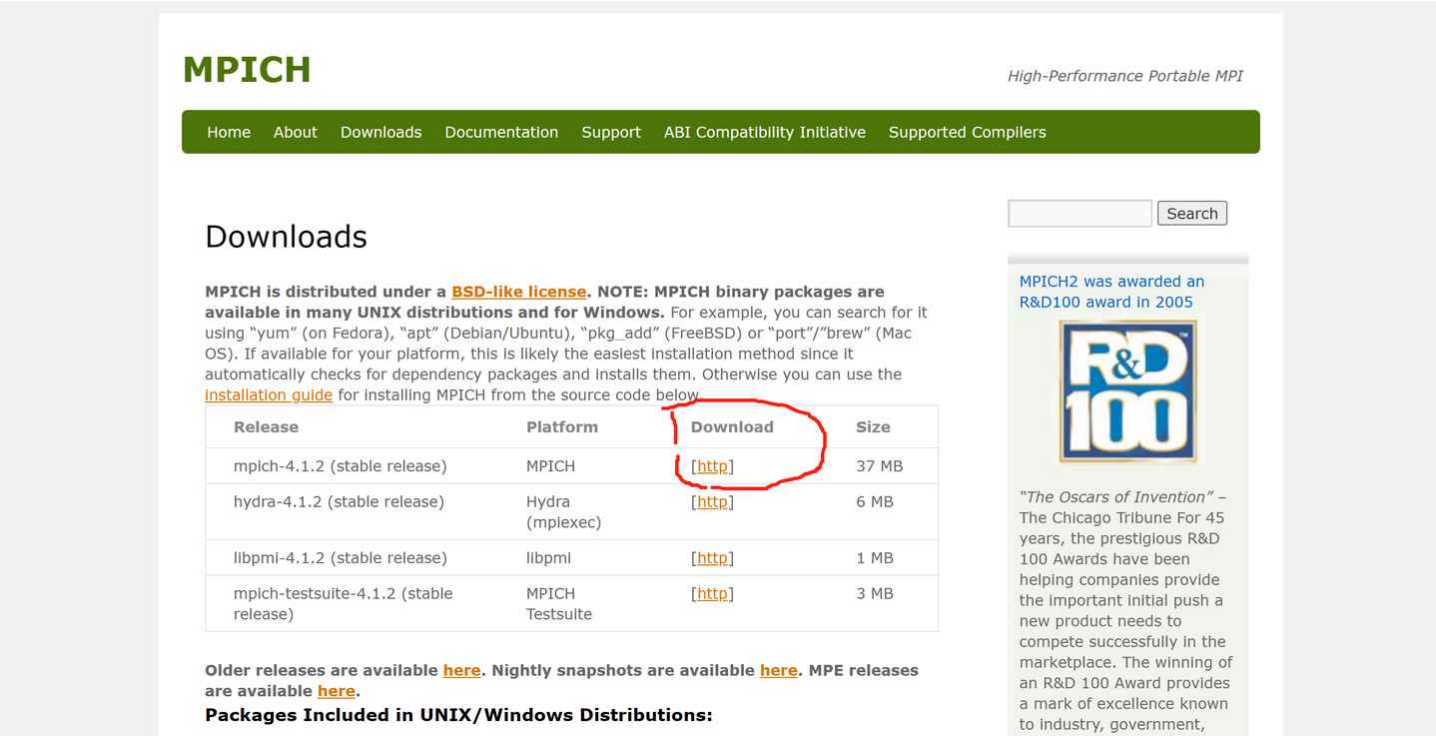
```
gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

自此，我们成功安装了gcc11.4.0，类似的，我们可以通过sudo apt-get install g++来安装g++，g++是编译C++代码的编译器。

接下来我们开始安装mpicc

mpicc的安装

mpicc的安装本人在网上找到了一个教程https://zhuanlan.zhihu.com/p/356705583，根据这个教程，我们需要先去MPI官网https://www.mpich.org/downloads/，进入官网以后选择对应MPI版本，比如说本人选择的是最新的4.1.2版本，选中红色的[http]，右键复制下载链接，然后回到我们自己搭建的服务器。




```
wget https://www.mpich.org/static/downloads/4.1.2/mpich-4.1.2.tar.gz
tar -zvxf mpich-4.1.2.tar.gz
cd mpich-4.1.2/
./configure --disable-fortran --prefix=/usr/local/mpich-4.1.2
make
make install
```

第一行命令：利用wget下载压缩包

第二行命令：利用tar解压

第三行命令：进入文件夹mpich-4.1.2

第四行命令：配置mpi环境，注意这个usr/local其实是在我们账户的home目录里面，我们可以通过下面这部分命令查找usr/local的位置以及改文件夹里面的内容。这里注意--disable-fortran表示我们没有安装fortran，如果事先安装了fortran，这部分修改为./configure --prefix=/usr/local/mpich-4.1.2，我们可以通过fortran --version检查服务器是否安装fortran。

```
:~/mpich-4.1.2$ cd
:~$ pwd
/home/x
:~$ cd ..
:/home$ ls
x
:/home$ cd ..
:/$ ls
bin   dev  home  lib    lib64   lost+found  mnt  proc  run   snap  sys  usr
boot  etc  init  lib32  libx32  media       opt  root  sbin  srv   tmp  var
:/$ cd usr/local/
:/usr/local$ pwd
/usr/local
:/usr/local$ ls
bin  etc  games  include  lib  man  sbin  share  src
```

第五行命令：使用make来编译，make会自动寻找当前文件夹的Makefile或者makefile文件，然后执行对应的编译命令。但是我们直接运行make发现报错，下面的报错信息提醒我们服务器还没有安装make这个工具，因此我们需要根据提示sudo apt install make来安装make。

```
~/mpich-4.1.2$ make
Command 'make' not found, but can be installed with:
sudo apt install make          # version 4.3-4.1build1, or
sudo apt install make-guile    # version 4.3-4.1build1
```

根据提示我们安装make以后，回到mpich-4.1.2文件夹再次使用make，（**如果只安装gcc，不安装g++）结果报错说当前文件夹根本没有makefile文件。**

```
:~$ sudo apt install make
[sudo] password for x
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  make-doc
The following NEW packages will be installed:
  make
0 upgraded, 1 newly installed, 0 to remove and 100 not upgraded.
Need to get 180 kB of archives.
After this operation, 426 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 make amd64 4.3-4.1build1 [180 kB]
Fetched 180 kB in 2s (109 kB/s)
Selecting previously unselected package make.
(Reading database ... 28451 files and directories currently installed.)
Preparing to unpack .../make_4.3-4.1build1_amd64.deb ...
Unpacking make (4.3-4.1build1) ...
Setting up make (4.3-4.1build1) ...
Processing triggers for man-db (2.10.2-1) ...
:~$ cd mpich-4.1.2/
.~/mpich-4.1.2$ make
make: *** No targets specified and no makefile found.  Stop.
```

这里本人发现，如果一开始只安装gcc，不安装fortran,g++，那么到了这里make的时候就会出现上面这种报错，因为在上一步configure的时候就abort了，会提示g++,fortran没有安装导致配置中止，如果不想安装g++,fortran，就需要在配置的时候修改命令为

./configure --disable-fortran --disable-c++ --prefix=/usr/local/mpich-4.1.2

否则就提前安装好g++和fortran，如果仅仅只安装了gcc,g++，那么才是使用./configure --disable-fortran --prefix=/usr/local/mpich-4.1.2来配置。

此时我们安装g++,gcc以后配置成功，再次使用make就能正常编译了，我们查看mpich-4.1.2文件内容也会发现文件夹里面有Makefile文件。

下面我们就可以开始make了，make结束以后应该是下面这个界面：

```
make[2]: Entering directory '/home/xgq/mpich-4.1.2/modules/yaksa'
make[2]: Nothing to be done for 'all'.
make[2]: Leaving directory '/home/xgq/mpich-4.1.2/modules/yaksa'
Making all in src/pmi
make[2]: Entering directory '/home/xgq/mpich-4.1.2/src/pmi'
make[3]: Entering directory '/home/xgq/mpich-4.1.2/src/pmi'
make[3]: Leaving directory '/home/xgq/mpich-4.1.2/src/pmi'
make[2]: Leaving directory '/home/xgq/mpich-4.1.2/src/pmi'
Making all in /home/xgq/mpich-4.1.2/modules/libfabric
make[2]: Entering directory '/home/xgq/mpich-4.1.2/modules/libfabric'
make  all-am
make[3]: Entering directory '/home/xgq/mpich-4.1.2/modules/libfabric'
make[3]: Leaving directory '/home/xgq/mpich-4.1.2/modules/libfabric'
make[2]: Leaving directory '/home/xgq/mpich-4.1.2/modules/libfabric'
Making all in src/mpi/romio
make[2]: Entering directory '/home/xgq/mpich-4.1.2/src/mpi/romio'
make[3]: Entering directory '/home/xgq/mpich-4.1.2/src/mpi/romio'
make[3]: Leaving directory '/home/xgq/mpich-4.1.2/src/mpi/romio'
make[2]: Leaving directory '/home/xgq/mpich-4.1.2/src/mpi/romio'
Making all in src/pm/hydra
make[2]: Entering directory '/home/xgq/mpich-4.1.2/src/pm/hydra'
make  all-recursive
make[3]: Entering directory '/home/xgq/mpich-4.1.2/src/pm/hydra'
Making all in .
make[4]: Entering directory '/home/xgq/mpich-4.1.2/src/pm/hydra'
make[4]: Leaving directory '/home/xgq/mpich-4.1.2/src/pm/hydra'
make[3]: Leaving directory '/home/xgq/mpich-4.1.2/src/pm/hydra'
make[2]: Leaving directory '/home/xgq/mpich-4.1.2/src/pm/hydra'
Making all in .
make[2]: Entering directory '/home/xgq/mpich-4.1.2'
make[2]: Nothing to be done for 'all-am'.
make[2]: Leaving directory '/home/xgq/mpich-4.1.2'
Making all in examples
make[2]: Entering directory '/home/xgq/mpich-4.1.2/examples'
make[2]: Nothing to be done for 'all'.
make[2]: Leaving directory '/home/xgq/mpich-4.1.2/examples'
make[1]: Leaving directory '/home/xgq/mpich-4.1.2'
```

然后我们使用make install命令，但是往往会被提示说缺少权限，下面这个图片说permission denied就是这个意思。

```
 /usr/bin/mkdir -p '/usr/local/mpich-4.1.2/include'
 /usr/bin/install -c -m 644 include/mpio.h include/mpiof.h '/usr/local/mpich-4.1.2/include'
/usr/bin/install: cannot remove '/usr/local/mpich-4.1.2/include/mpio.h': Permission denied
/usr/bin/install: cannot remove '/usr/local/mpich-4.1.2/include/mpiof.h': Permission denied
make[4]: *** [Makefile:4490: install-nodist_includeHEADERS] Error 1
make[4]: Leaving directory '/home/xsq/mpich-4.1.2/src/mpi/romio'
make[3]: *** [Makefile:4826: install-am] Error 2
make[3]: Leaving directory '/home/xsq/mpich-4.1.2/src/mpi/romio'
make[2]: *** [Makefile:4517: install-recursive] Error 1
make[2]: Leaving directory '/home/xsq/mpich-4.1.2/src/mpi/romio'
make[1]: *** [Makefile:31566: install-recursive] Error 1
make[1]: Leaving directory '/home/xsq/mpich-4.1.2'
make: *** [Makefile:31677: install] Error 2
```

面对这种报错，我们需要使用管理员权限（幸好是我们自己搭建的计算机服务器，我们自己就是管理员，如果是公共服务器，往往需要管理员才能做下去），使用sudo make install命令，然后会提示我们添加密码，写完密码以后回车即可安装

```
/usr/bin/install -c -m 644 MPI_Win_unlock_all.html /usr/local/mpich-4.1.2/share/doc/mpich/www3/MPI_Win_unlock_all.html
/usr/bin/install -c -m 644 MPI_Win_wait.html /usr/local/mpich-4.1.2/share/doc/mpich/www3/MPI_Win_wait.html
/usr/bin/install -c -m 644 MPI_Wtick.html /usr/local/mpich-4.1.2/share/doc/mpich/www3/MPI_Wtick.html
/usr/bin/install -c -m 644 MPI_Wtime.html /usr/local/mpich-4.1.2/share/doc/mpich/www3/MPI_Wtime.html
/usr/bin/install -c -m 644 index.htm /usr/local/mpich-4.1.2/share/doc/mpich/www3/index.htm
/usr/bin/install -c -m 644 mpi.cit /usr/local/mpich-4.1.2/share/doc/mpich/www3/mpi.cit
if [ ! -e /usr/local/mpich-4.1.2/share/doc/mpich ] ; then mkdir -p /usr/local/mpich-4.1.2/share/doc/mpich ; fi
if [ -s ./doc/userguide/user.pdf ] ; then /usr/bin/install -c -m 644 ./doc/userguide/user.pdf /usr/local/mpich-4.1.2/share/doc/mpich/user.pdf ; fi
if [ -s ./doc/installguide/install.pdf ] ; then /usr/bin/install -c -m 644 ./doc/installguide/install.pdf /usr/local/mpich-4.1.2/share/doc/mpich/ins
tall.pdf ; fi
if [ -s ./doc/logging/logging.pdf ] ; then /usr/bin/install -c -m 644 ./doc/logging/logging.pdf /usr/local/mpich-4.1.2/share/doc/mpich/logging.pdf ;
 fi
 mkdir -p '/usr/local/mpich-4.1.2/include'
 /usr/bin/install -c -m 644 src/include/mpi_proto.h '/usr/local/mpich-4.1.2/include'
 mkdir -p '/usr/local/mpich-4.1.2/include'
 /usr/bin/install -c -m 644 src/binding/cxx/mpicxx.h src/include/mpi.h '/usr/local/mpich-4.1.2/include'
 mkdir -p '/usr/local/mpich-4.1.2/lib/pkgconfig'
 /usr/bin/install -c -m 644 src/packaging/pkgconfig/mpich.pc '/usr/local/mpich-4.1.2/lib/pkgconfig'
make[3]: Leaving directory '/home/xsq/mpich-4.1.2'
make[2]: Leaving directory '/home/xsq/mpich-4.1.2'
Making install in examples
make[2]: Entering directory '/home/xsq/mpich-4.1.2/examples'
make[3]: Entering directory '/home/xsq/mpich-4.1.2/examples'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/home/xsq/mpich-4.1.2/examples'
make[2]: Leaving directory '/home/xsq/mpich-4.1.2/examples'
make[1]: Leaving directory '/home/xsq/mpich-4.1.2'
```

安装成功以后，应该是上面这个界面，到了这里，就差最后一步了，此时如果我们使用命令mpicc --version，应该还是会说mpicc没有安装，这是因为我们还没有配置环境变量。环境变量的配置如下：

假设上面一路安装都正常了，我们退出mpich-4.1.2文件夹，回到主目录，然后使用命令

```
vi ~/.bashrc
```

打开一个叫.bashrc的文件，这里面内容很多，然后我们需要在这个文件最下面一行（其实哪一行都可以，但是本人习惯放在最后一行，方便之后修改添加内容）添加下面一句话：

```
export PATH="/usr/local/mpich-4.1.2/bin:$PATH"
```

添加结束以后保存文件内容然后退出来，使用下面这句命令激活环境。

```
source ~/.bashrc
```

激活以后，我们此时使用mpicc --version查看mpicc的版本即可发现下面这部分内容被打印出来。

```
gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

此时我们还是不放心的话，可以在目录下新建一个文件命名为mpi_he.c，该文件的内容为

```
#include <mpi.h>
#include <stdio.h>
int main(int argc,char** argv){
    int size;
    int rank;
    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&size);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    printf("the process of %d of %d:hello world\n",rank,size);
    if (rank == 0){
            printf("that is all\n");
    }
    MPI_Finalize();
    return 0;
}
```

然后我们使用

```
mpicc mpi_he.c -o he
mpiexec -n 4 ./he
```

第一个命令是编译命令，第二个命令指定了4个进程来运行这段代码，如果编译正确，最后就会打印出下面这部分内容：

```
the process of 0 of 4:hello world
that is all
the process of 1 of 4:hello world
the process of 2 of 4:hello world
the process of 3 of 4:hello world
```

到此，我们终于成功安装好了mpi，mpi的相关代码以基础知识，可以参考本人的CSDN博客

安装结束以后，上面的这个文件夹mpi-4.1.2可以删除了。

openmp安装

openmp似乎不用安装，只需要编译的时候添加-fopenmp链接库即可，比如说下面我们给定一个openmp实现hello word代码命名为omp_hello.c

```c
#include <omp.h> // omp header file
#include <stdio.h> // standard I/O
int main(int argc, char *argv[]){
  int    nthreads, tid;
  double t0, t1;
  //omp_set_num_threads(4);
  t0 = omp_get_wtime();
#pragma omp parallel private(tid)
  {
    nthreads = omp_get_num_threads(); // get num of threads
    tid = omp_get_thread_num(); // get my thread id
    printf("From thread %d out of %d, Hello World!\n", tid, nthreads);
  }
  t1 = omp_get_wtime();
  nthreads = omp_get_num_threads(); // get num of threads
  tid = omp_get_thread_num(); // get my thread id
  printf("From xiao thread %d out of %d, Hello World!\n", tid, nthreads);
  printf("Time elapsed is %f.\nThat's all, folks!\n", t1-t0);
  return 0;
}
```

然后我们使用下面这部分命令来设置环境变量，编译和执行：

```
export OMP_NUM_THREADS=4
gcc omp_hello.c -o hello -fopenmp -lm
./hello
```

第一句命令表示设置openmp使用4个线程，

第二句命令使用gcc编译文件，并且链接openmp和数学库

第三句执行代码

nvidia编译器nvcc安装

使用nvcc --version命令，系统会提示如何安装nvcc编译器。

上述过程参考https://zhuanlan.zhihu.com/p/644960969