1.apt-get

是一条linux命令，可自动从互联网的软件仓库中搜索、安装、升级、卸载软件或操作系统。

从网上按照教程换了清华的源（并不符合版本来着），安装python3.7.3时才换了正确的阿里源。

2.Vim

Vim是从 vi 发展出来的一个文本编辑器，代码补完、编译及错误跳转等方便编程的功能特别丰富，在程序员中被广泛使用。

Vim是一个程序开发工具而不是文字处理软件

Vim有输入、取代、命令等三种模式

以下是vim的一些常用命令

|  |  |
| --- | --- |
| **搜索替换** | |
| /word | 向光标之下寻找一个名称为 word 的字符串。 |
| ?word | 向光标之上寻找一个字符串名称为 word 的字符串。 |
| n | 重复前一个搜寻的动作。 |

|  |  |
| --- | --- |
| :1,$s/word1/word2/g | 从第一行到最后一行寻找 word1 字符串，并将该字符串取代为 word2 |
| :1,$s/word1/word2/gc | 从第一行到最后一行寻找 word1 字符串，并将该字符串取代为 word2 ！且在取代前显示提示字符给用户确认 (confirm) 是否需要取代 |

|  |  |  |  |
| --- | --- | --- | --- |
| **删除和复制** | | | |
| x, X | x 为向后删除一个字符 (相当于 [del] 按键)， X 为向前删除一个字符(相当于 [backspace] 亦即是退格键) | | |
| dd | 删除游标所在的那一整行 | |
| ndd | n 为数字。删除光标所在的向下 n 行 | |
| yy | | 复制游标所在的那一行 |

|  |  |
| --- | --- |
| p, P | p 为将已复制的数据在光标下一行贴上，P 则为贴在游标上一行 |
| u | 复原前一个动作。 |
| [Ctrl]+r | 重做上一个动作。 |
| . | 小数点。重复前一个动作。 |

|  |  |
| --- | --- |
| **进入输入或取代的编辑模式** | |
| i | 进入输入模式(Insert mode)：『从目前光标所在处输入』 |
| r, R | 进入取代模式(Replace mode)： r 只会取代光标所在的那一个字符一次；R会一直取代光标所在的文字，直到按下 ESC 为止； |
| [Esc] | 退出编辑模式，回到一般模式中 |

|  |  |
| --- | --- |
| **指令行的储存、离开等指令** | |
| :w | 将编辑的数据写入硬盘档案中 |
| :w! | 强制写入该档案。（需要权限） |
| :q | 离开 vi |
| :q! | 强制离开不储存档案。 |
| ! 在 vi 当中，常常具有『强制』的意思 | |
| :wq | 储存后离开，若为 :wq! 则为强制储存后离开 |
| ZZ | 若档案没有更动，则不储存离开，若档案已经被更动过，则储存后离开 |
| :w [filename] | 将编辑的数据储存成另一个档案（类似另存新档） |
| :r [filename] | 将 『filename』 这个档案内容加到游标所在行后面 |
| :! command | 暂时离开 vi 到指令行模式下执行 command 的显示结果例如： 『:! ls /home』即可在 vi 当中察看 /home 底下以 ls 输出的档案信息！ |

**3.Git**

git是一种版本管理工具，而github只是一个远程仓库。实际上没有github这种远程仓库，计算机中git也会建立一个本地的仓库，或者链接到其他的远程仓库。

**以下是git的一些命令**

| **命令** | **功能** |
| --- | --- |
| git init | 在本地的当前目录里初始化git仓库 |
| git status | 查看当前仓库的状态（常用） |
| git add -A | 增加目录中所有的文件到缓存区 |
| git add file | 增加相应文件到缓存区 |
| git commit -m "信息" | 将缓存区中更改提交到本地仓库 |
| git log | 查看当前版本之前的提交记录 |
| git reflog | 查看HEAD的变更记录，包括回退 |
| git branch -b branch\_name | 建立一个新的分支 |
| git diff | 查看当前文件与缓存区文件的差异 |
| git checkout -- file | 取消更改，将缓存区的文件提取覆盖当前文件 |
| git reset --hard 版本号 | 回退到相应版本号，同样也可以回退到未来的版本号 |
| git clean -xf | 删除当前目录中所有未追踪的文件 |
| git config --global core.quotepath false | 处理中文文件名 |

**4.Python3.7.3的安装**

Python的安装实在是困难重重。

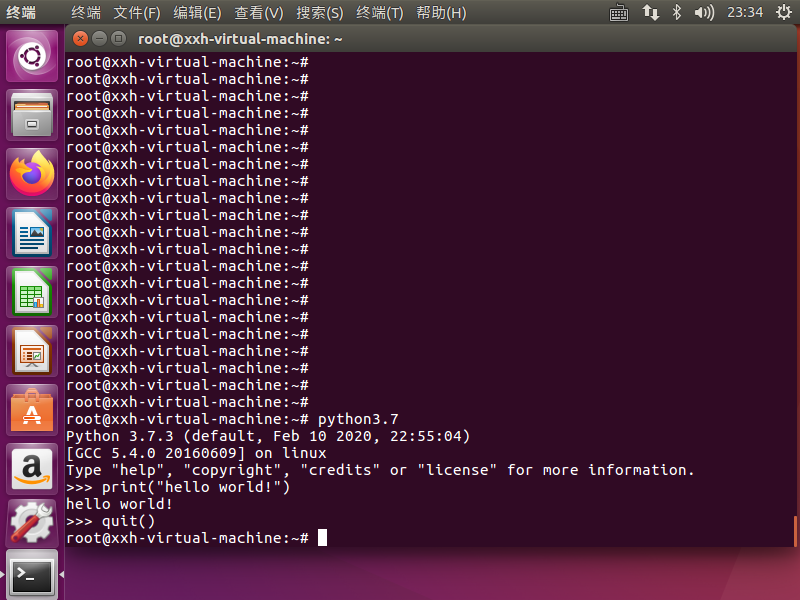
首先查到的一个教程，并没有说到要下载依赖包，导致第一次安装失败。

第二次找到了有依赖包的版本，但是并不相符，其用yum安装依赖包，然而本系统应该使用apt-get来安装，然而这个点在我查找许久之后才发现。

之后又找到了用apt-get安装的版本，安装依赖包时，发现不能locate到package，查找了很久，问了部长才知道时apt-get源的问题。

换好正确的源以后正常安装依赖包，正常编译，安装好之后，建立软连接时发现了异常，并不能解决，后重启了一遍系统，再打开时就不能打开终端了，找了几个方法都无法解决这个状况，因此就重新建立了一个虚拟机。

之后的事情开始顺利起来，因为已经经历过了，所以比较熟练，正常操作以后就把python3.7.3好好地、完整地安装好了，也建立了软连接，以下就是成功的证明。



**5.心得**

果然还是没有紧迫感，导致也是花了很多时间来完成，当然，主要的时间都在第四个任务的完成上面，因为确实遇到了很多困难，具体的说明都在上面了。

结论的话，大概就是先全面查找再进行操作吧。前面的操作都是查到一个有用的就进行操作，然后就可能对系统造成了不可逆的奇怪伤害，因此导致了前一个虚拟机的终端无法打开，才新建了一个虚拟机。所以，在进行操作以前应该进行全面查找，综合考虑之后再进行操作。

另外一个，这两个project我都没有跟着进度写学习报告，所以上面其实没有完成任务的具体过程，导致之后写学习报告的时候，已经把之前的内容大概都忘掉了。所以，下面两个报告，我会跟着进度来写，把学习的过程体现出来。