# Ubuntu16.04

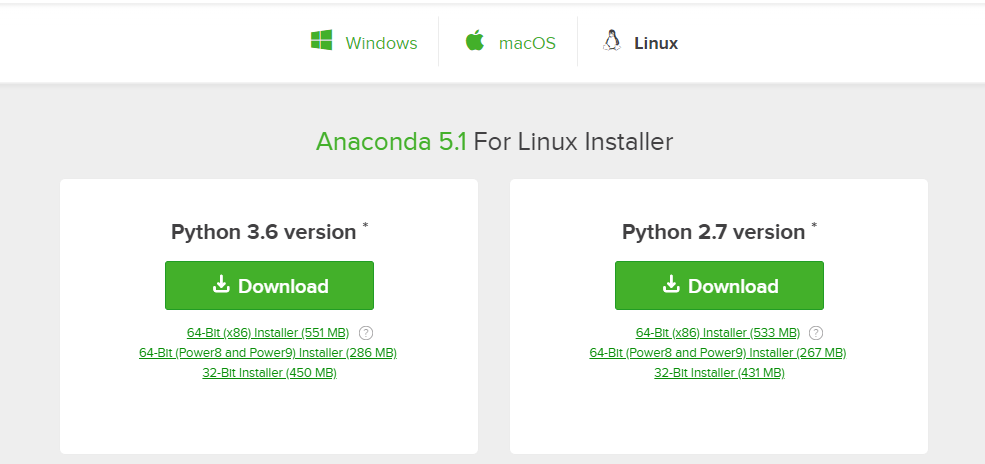
# Anaconda+CUDA+CUDNN+CRF+pycharm

# 安装指南

## 一：anaconda安装

### 1：下载anaconda

官网：<https://www.anaconda.com/download/>



清华大学镜像站：<https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/>



选择版本进行下载

我这里下载的是Anaconda3-5.1.0-Linux-x86\_64.sh

（这个也可以[https://repo.continuum.io/archive/](https://repo.continuum.io/archive/Anaconda3-5.1.0-Linux-x86_64.sh)）

### 2：安装anaconda

打开终端进入到下载好的文件的目录下（或找到下载文件，在其文件夹内打开终端）

首先执行sudo chmod +x Anaconda3-5.1.0-Linux-x86\_64.sh



然后执行./Anaconda3-5.1.0-Linux-x86\_64.sh



根据提示默认输入回车和“yes”

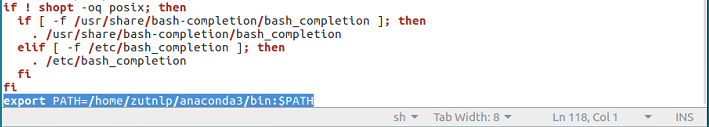
### 3：anaconda环境配置

执行sudo gedit ~/.bashrc进行环境配置



在末尾添加

export PATH=/home/用户名/anaconda3/bin:$PATH



完成后许需重启计算机或注销方能生效

### 4：测试anaconda安装

执行conda -V



出现conda版本则成功

### 5：conda的更新

执行conda update -n base conda



查看更新后的版本



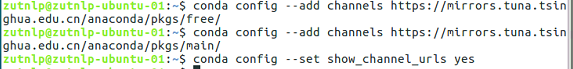
### 6：添加 Anaconda Python 免费仓库

分别运行以下命令

conda config --add channels <https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/>

conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main/

conda config --set show\_channel\_urls yes



### 7：测试源是否更改成功

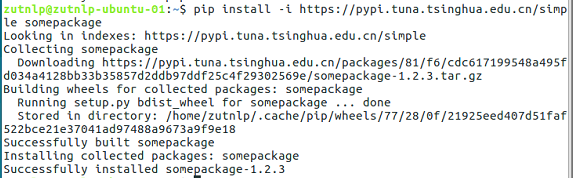
运行 conda install numpy



## 二：pypi 镜像使用帮助

pypi 镜像每 5 分钟同步一次。若临时使用则执行以下命令

pip install -i https://pypi.tuna.tsinghua.edu.cn/simple somepackage

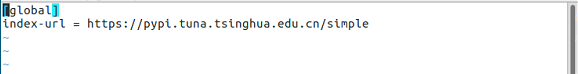


(有时可能会提示pip版本过低，可利用 pip install --upgrade pip更新pip)

若要设为默认

修改 ~/.config/pip/pip.conf，若没有需要自己新建一个，在其中输入一下内用

（~/.config是位于home/用户目录下的隐藏文件，进入home/用户目录下CTRL+H就可以显示[隐藏文件](https://www.baidu.com/s?wd=隐藏文件&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)）



## 三：anaconda的配置与激活

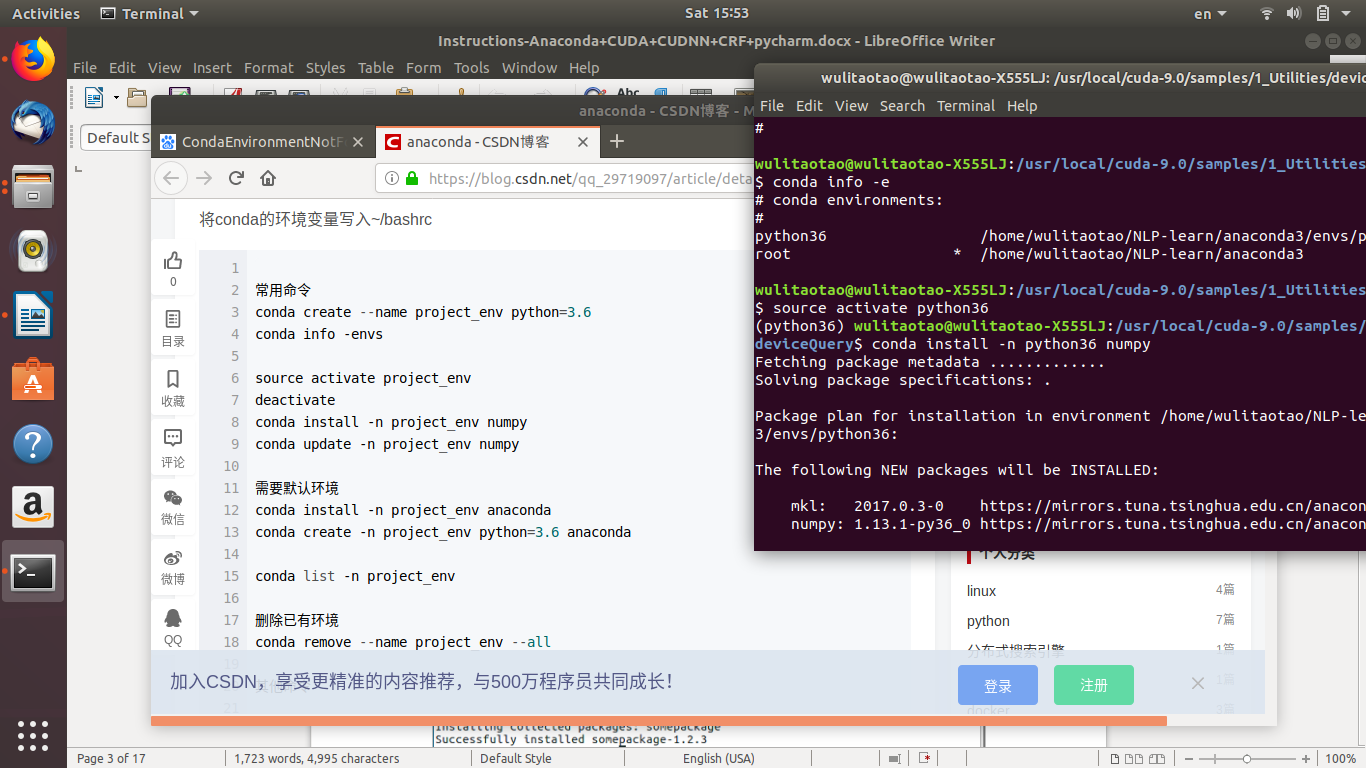
首先执行命令创建一个名为python36的虚拟环境，默认安装就好，然后等待创建完毕。



下边两条命令时进入（激活）环境和退出环境的命令

进入：source activate python36





退出：source deactivate



## 四：安装Cuda

### 1：预安装操作

1. **验证系统是否具有支持CUDA的GPU**

要验证您的GPU是否支持CUDA，请转至您的发行版的“系统属性”，或从命令行输入：

$ lspci | grep -i nvidia

如果您没有看到任何设置，请通过输入更新Linux维护的PCI硬件数据库 更新pciids （一般发现在 / sbin目录）在命令行并重新运行之前 lspci的 命令。

如果您的显卡来自NVIDIA，并且列在[http://developer.nvidia.com/cuda-gpus中](http://developer.nvidia.com/cuda-gpus)，则您的GPU支持CUDA。

CUDA工具包的发行说明还包含支持产品的列表。

1. **验证系统是否运行受支持的Linux版本**

CUDA开发工具只支持某些特定的Linux发行版。这些在CUDA工具包发行说明中列出。

要确定您正在运行的分发版本号和发行版号，请在命令行中输入以下内容：

$ uname -m && cat /etc/\*release

您应该看到类似于以下内容的输出，并针对您的特定系统进行修改：

x86\_64的

红帽企业Linux工作站版本6.0（圣地亚哥）

该 x86\_64的行表示您正在64位系统上运行。其余部分提供有关您的分配的信息。

3.[验证系统是否安装了](https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#verify-that-gcc-is-installed)gcc

要验证系统上安装的gcc的版本，请在命令行上输入以下内容：

gcc --version或gcc -v

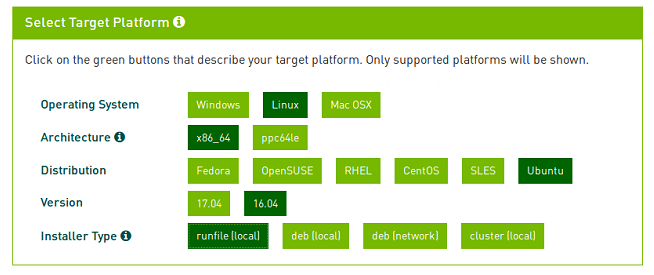
1. [验证系统是否安装了正确的内核头文件和开发包](https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#verify-kernel-packages)

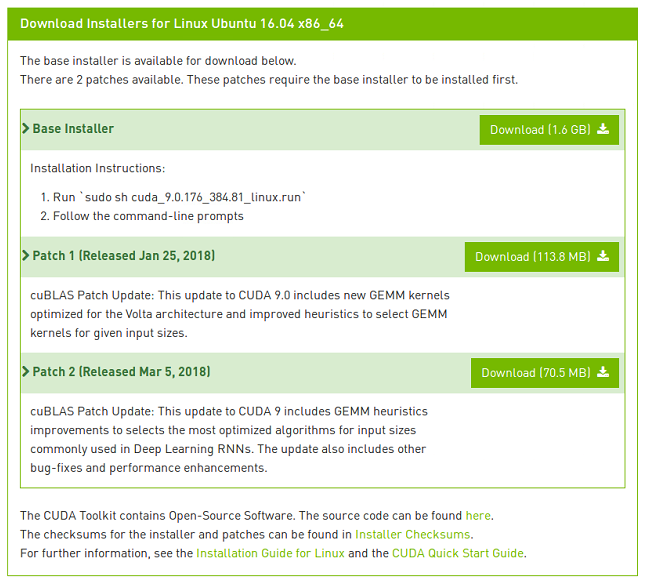
系统运行的内核版本可以通过运行以下命令找到：

$uname -r

2：下载

网址：https://developer.nvidia.com/cuda-90-download-archive





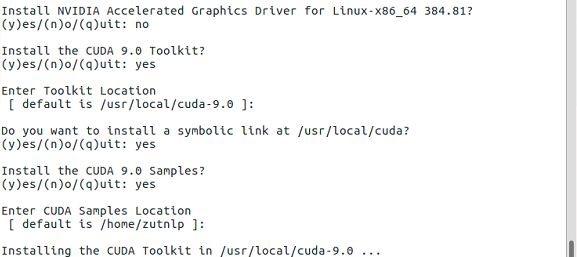
如上图所示，将base installer，patch1，patch2均下载下来

### 3：安装

首先在以下好的文件的文件夹内，执行sudo sh cuda\_9.0.176\_384.81\_linux.run



当出现以下时，提示是否安装NVIDIAAccelerateGraphicsDriver是选择“no”，其他都是同意



等待安装完成:

### 4.配置环境变量

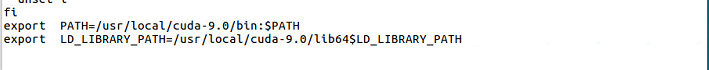
执行sudo gedit  /etc/profile打开配置文件



在末尾添加

export  PATH=/usr/local/cuda-9.0/bin:$PATH

export  LD\_LIBRARY\_PATH=/usr/local/cuda-9.0/lib64$LD\_LIBRARY\_PATH



保存退出即可。

重启或注销系统即可生效。

### 5.验证是否安装成功

分别执行以下三条命令

cd  /usr/local/cuda-9.0/samples/1\_Utilities/deviceQuery

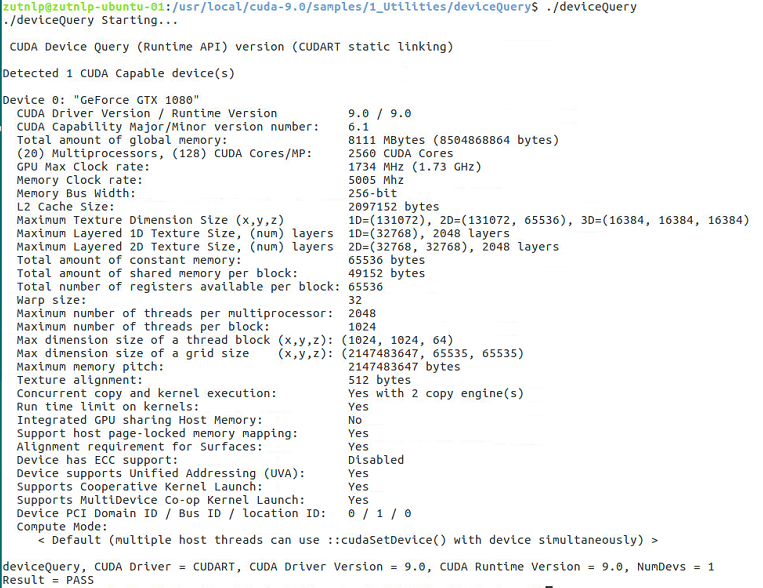
sudo make

./deviceQuery

如果显示的是关于GPU的信息，则说明安装成功了







## 五：安装CUDNN

### 1.下载

<https://developer.nvidia.com/rdp/cudnn-download>

如果之前注册过NVIDIA账户直接下载便可，否则需要先注册才可以下载

选择合适的版本



将下边这个下载



### 2.安装

进入到Downloads，也就是包含cuDNN Tar文件的目录。

用以下命令解压缩cuDNN软件包

tar -xzvf cudnn-9.0-linux-x64-v7.tgz

将以下文件复制到CUDA Toolkit目录中。

sudo cp cuda/include/cudnn.h /usr/local/cuda/include

sudo cp cuda/lib64/libcudnn\* /usr/local/cuda/lib64

授予权限

sudo chmod a+r /usr/local/cuda/include/cudnn.h

sudo chmod a+r /usr/local/cuda/lib64/libcudnn\*

可参考（https://docs.nvidia.com/deeplearning/sdk/cudnn-install/index.html#axzz4qYJp45J2）

## 六：安装tensorflow

pip install tensorflow

进入python

>>> import tensorflow as tf 即可

## 七：安装CRF++

1. 下载地址： <https://taku910.github.io/crfpp/>
2. 解压

tar -zxvf CRF++-0.58.tar.gz



1. 进入CRF++0.58目录下

cd CRF++-0.58

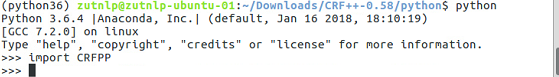
1. ./configure



1. make
2. sudo make install
3. 进入子目录python中
4. python setup.py build
5. python setup.py install
6. 进入python执行import CRFPP

（若出现ImportError: libcrfpp.so.0: cannot open shared object file: No such file or directory

退出python后执行ln -s /usr/local/lib/libcrfpp.so.0 /usr/lib/）之后重新进入python 然后import试试。



## 八：安装pycharm

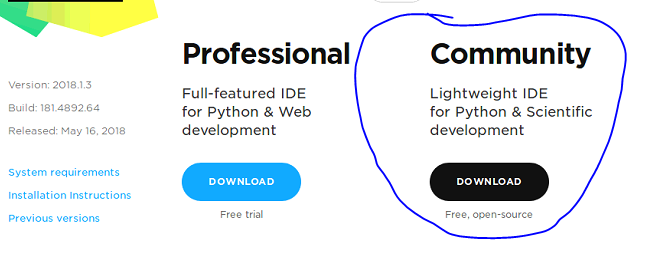
### 1.下载

进入pycharm官网（https://www.jetbrains.com/pycharm/）

点击dowload now



选择社区版，点击下载



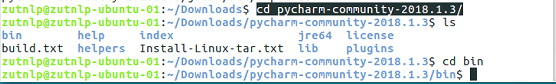
### 2.安装

进入Downloads文件夹

在终端内执行tar -zxvf pycharm-community-2018.1.3.tar.gz解压安装包



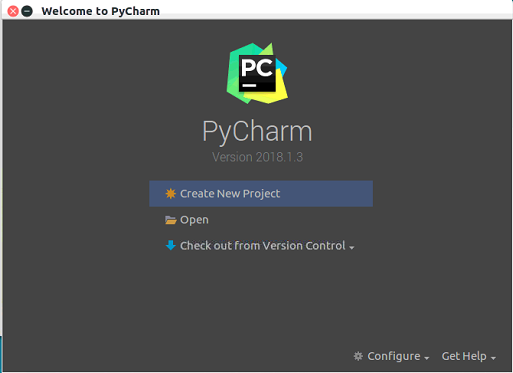
进入解压后的文件夹内的bin文件夹cd pycharm-community-2018.1.3/bin



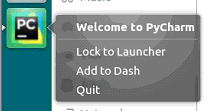
执行./pycharm.sh



默认选择便安装完成，并启动。



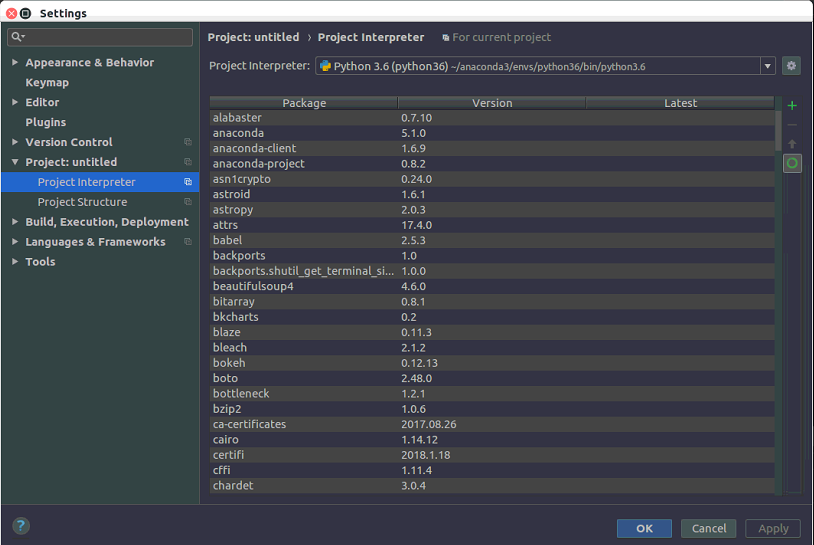
### 3.固定桌面图标



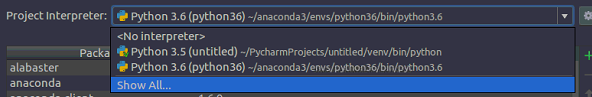
### 4.更换环境

装换python环境至anaconda中的虚拟环境python36

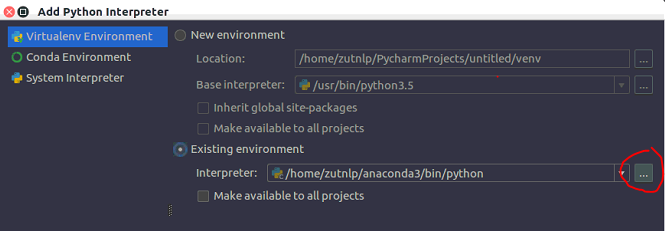
File->settings



点击



点击+号添加路径



选择到anaconda3/envs/python36/bin/python3.6

然后应用即可

安装python3.6教程: <https://www.cnblogs.com/SNine/p/7805068.html>

安装包地址（用wget命令链接 ）：

wget <https://www.python.org/ftp/python/3.6.3/Python-3.6.3.tar.xz>

遇到的问题：

1. 在make install 后提示错误：

import pip

zipimport.ZipImportError: can't decompress data; zlib not available

Makefile:1079: recipe for target 'install' failed

make: \*\*\* [install] Error 1

原因：

This question is answered in the [pyenv wiki](https://github.com/pyenv/pyenv/wiki). You have to prep your build environment first before trying to build a Python interpreter using pyenv.

解决方案：

According to the wiki, you need the following packages installed for Ubuntu:

apt-get install -y make build-essential libssl-dev zlib1g-dev libbz2-dev libreadline-dev libsqlite3-dev wget curl llvm libncurses5-dev xz-utils tk-dev

详细教程：

# [【Linux】Ubuntu安装Python3](https://www.cnblogs.com/SNine/p/7805068.html)

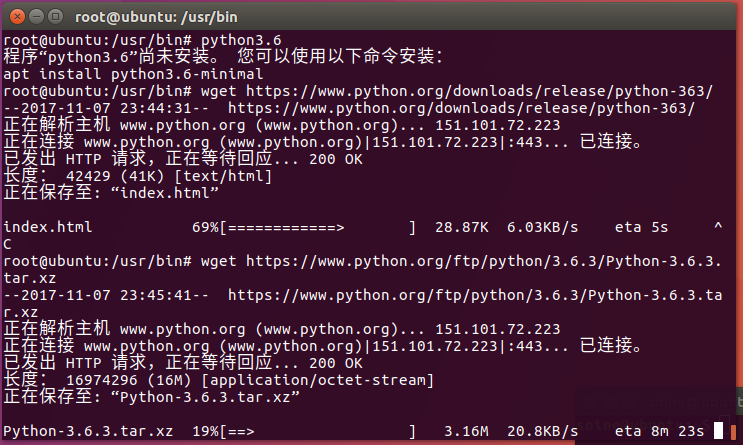
**叙述：**

　　Ubuntu17.04自带Python2.7与Python3.5.3的版本，由于Python2与Python3有着一些差距可能需要安装更新Python3的版本，并且切换默认的Python解释器。

**0x01：**

　　通过wget获取最新python源码，本文发布时最新版本为python3.6.3（https://www.python.org/ftp/python/3.6.3/Python-3.6.3.tar.xz）

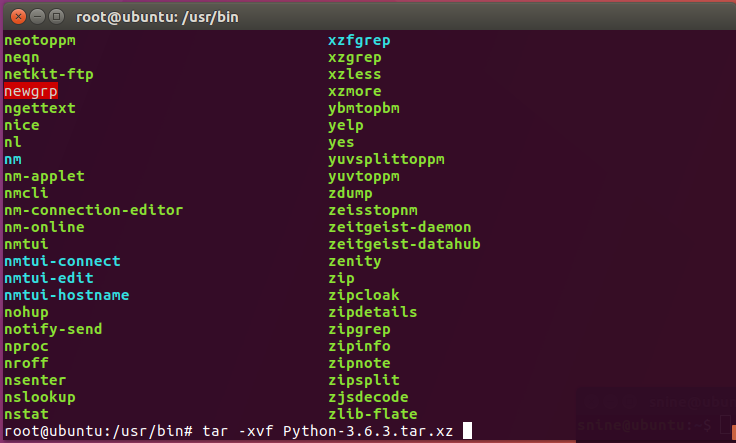
wget https://www.python.org/ftp/python/3.6.3/Python-3.6.3.tar.xz



**0x02：**

　　解压下载好的原文件

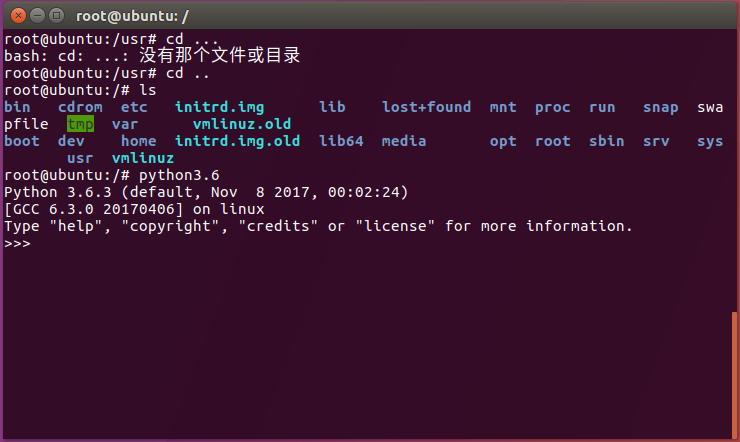
tar -xvf Python-3.6.3.tar.xz



**0x03：**

　　cd到刚解压的目录后，对源文件进行编译安装

cd Python-3.6.3  
./configure  
make  
make install



**0x04：**

　　给Python3起一个别名：alias python=python3

alias python=python3