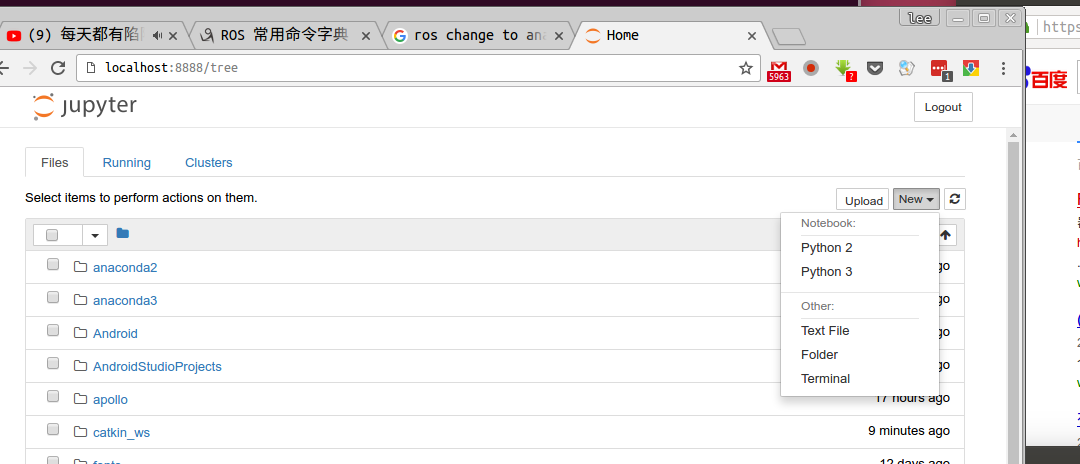
一个用gparted来增加目录容量的帖子，暂时还不敢使用，怕把系统给分坏了，先做个记录吧，不过

/这个目录容量确实是不多了。。。

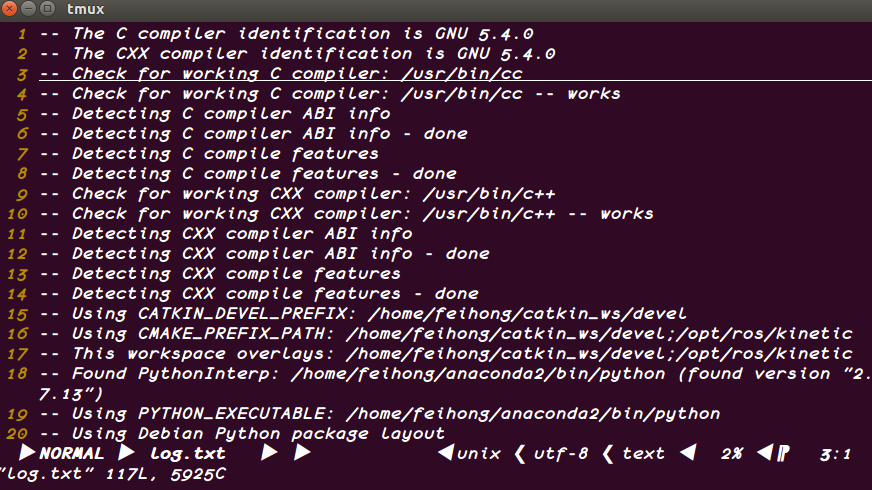
<http://blog.csdn.net/jwq2011/article/details/54599744>

.zshrc命令里面还是可以根据自己的需要来更改python的别名

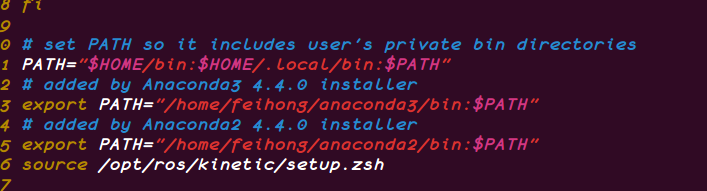
我操，有意思了，改了这个anaconda的顺序之后，ipython的kernel业完全正确了，列表显示完全正常了，而且我亲测了一下，选择python2,python3都是正确版本的kernel，牛逼了

路径切换以后，如果运行rosrun rqt\_graph rqt\_graph命令还是会报错，还是要用pip命令再安装两个包，才行，不过，那个index的错误已经没有了

果然可以看到python\_executable变成了anaconda2了



终于发现了这个特点了，原来读取.bashrc，或者.zshrc的时候，会以更加后面的path为主啊，所以如果我把anaconda的路径给放到后面的话，ros的python解释器就是以anaconda2为主了，牛逼了，再重新init一下，果然是anaconda2,这样，今后切换起来就方便了，不过现在安装命令pip就是anaconda2了，如果anaconda3要按照包，或者可以用conda3,或者直接进入anaconda3的bin目录里面去安装，免得搞困小恶狼



靠，在zshrc里面把anaconda3给注释掉之后，还是走的anaconda3里面的python

还是没有将ros的pythonpath给切换为anacodna3,靠，但是用pip来安装cakin\_pkg，暂时能解决，但是可能还是有dict.keys(),因为现在pip现在是anaconda3里的安装命令，系统默认的python3，是用的pip3

安装ros之后，应为我有2个anaconda，但是ros默认是用的anaconda3了，安装ros用的就是python3,结果导致包找不着了，还是按照下面这个帖子再安装一次包

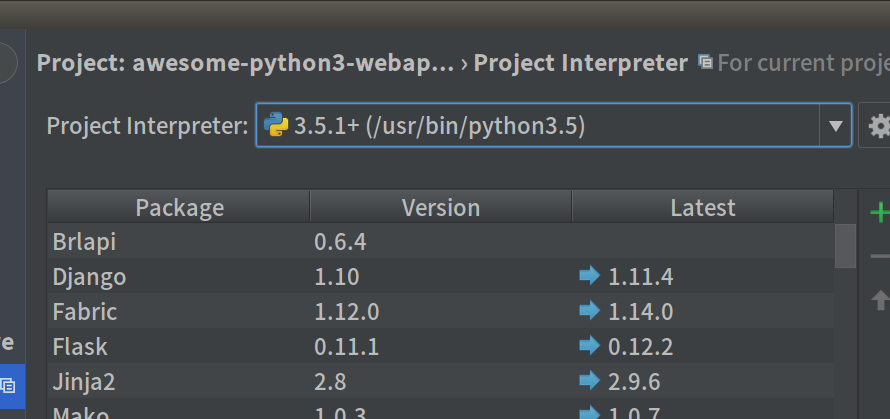
<http://blog.csdn.net/yguoelect/article/details/52887402>

但是关键还是要把python executable的路径从anaconda3改成anaconda2

2017.9.5

今天又在ubuntu下正式的安装了ros了，还好做了笔记，直接使用上次的那个很好的博客来搞，easy，愉快。。

还有个问题，ubuntu下的pycharm里面用的路径还都是系统原生的路径，并没有使用anaconda路径



果然，两个pip ,pip ,pip3 list的内容不是一样的，pip的才是anaconda的list，看来pip3是ubuntu系统原生的，现在系统的.bashrc,.zshrc用的都是anaconda的python(环境变量），看来安装python包还是得用pip才行

奇怪，怎么会有这种问题，，飞得用pip安装一下才行了

安装了anaconda以后安装pip里面的安装包还是要用pip install，最好用sudo pip install,如果用pip3 install安装的话，有时候会不识别，路径还是有点问题，在anaconda3/bin里面的pip是pip而不是pip3

恩，这个文章我觉得更好

<http://www.123kuai.com/index.php?a=show&c=index&catid=9&id=54&m=content>

Pitch,roll,yaw的区别，这个小文章讲的还是可以的，不过作者的x,y,z坐标的选择和我们平时常用的略有区别，这点要注意

<http://blog.163.com/vipwdp@126/blog/static/150224366201281935518196/>

vim的查找替换命令还是用的比较少的

<http://www.cnblogs.com/fakis/archive/2010/07/17/1976595.html>

<http://blog.csdn.net/lanxinju/article/details/5731843>

牛逼，chrome，firefox立马都有lastpass的插件，太好了，这样就不用再次来安装额外的app了

神了，chrome，firefox里面添加的网站，帐号，密码居然可以共享，，牛逼。。。

果然按照作者的操作，把项目clone到我本地的仓库，然后修改json，再push到github，修改是有效的，牛逼。。看来总的仓库，和其他的项目是不冲突的(其他的项目是从外面fork过来的),但是需要在本地的clone过来的那个项目下面进行add,commit,push

等操作，不能在上一级仓库目录push，层级就不对了。所以，在那个目录下，也把几个命令给写道脚本里面去了

我的git博客地址

[https://lhfly5201314.github.io](https://lhfly5201314.github.io/)

那个简历地址

https://lhfly5201314.github.io/easyresume/

github里面开启github pages功能的链接

<https://guides.github.com/features/pages/>

就是这个链接里面给出的提示

<https://github.com/Jaskey/easyresume>

相当于fork了一个别人的博客地址到自己的git io上了

markdown字体前后的\_abcd\_原来是表示意大利字体

chrome的15个小功能

<http://www.jianshu.com/p/oapzzC>

其中，shift+esc打开资源网页，还有搜索框里面粘贴搜索，以及ctrl+shift+i打开开发者工具页面，还是比较不错的

重载()的作用，类自带的构造方法()是为了对象的创建，是类名加上(),重载的()则是给对象用的，可以给对象再次赋值构造



这个网站太牛比了

markdown里面输入数学公式的各种参考。特别的全面的那种。

<https://math.meta.stackexchange.com/questions/5020/mathjax-basic-tutorial-and-quick-reference>

值得收藏，已经收藏到chrome里面了

靠，还是没能解决u盘和手机无法识别的问题，fuck，不爽

Ubuntu usb 解决办法，防止出bug，现在把地址给记下来

<https://askubuntu.com/questions/522515/sandisk-usb-not-detected-by-ubuntu-14-04-but-detected-by-windows-7>

可惜，最后的键位绑定f8,f9之后有点小bug，不能正常的打开，但是命令是完全可以正常打开的。。。还有设置自动带来浏览界面也是有问题，网页是弹开了，但是网页内容是空白的，可惜啊

好奇怪啊，任何绑定的情况，或者vim立马直接重复命令都不行，非要手动键入命令，奇怪，莫非是有什么buffer问题？？

牛逼啊，无意中安装了一个vim立马写markdown并实时预览的插件，相当不错啊，安装简单，使用也方便。。。还是咱们国人写的，文档可读性也好，牛逼了

https://github.com/iamcco/markdown-preview.vim

Yaw rate维基百科的解释非常棒，图文并茂。。

Ubuntu u盘无法识别解决方案1：

<http://www.linuxidc.com/Linux/2013-04/82886.htm>

Jupyter notebook还是shift+l来切换行好

ctrl+j,k来移动cell，vim模式下，有的快捷键还是不好用

确实是实现了复现，现在jupyter来编程，效率又可以提升了，cool

<https://github.com/lambdalisue/jupyter-vim-binding>

chrome立马不能用，这是唯一的一小点遗憾。。。。

牛逼了，照着这个哥们的github，在/jupyter –data-dir目录下clone项目并且安装，果然实现了vim binding，而且在./jupyter立马新建一个custom文件夹，然后新建一个custom.js文件，真的是实现了jj取代esc的作用，碉堡。。

不过，这个在chrome里面没有效果，chrome立马f1直接弹出帮助页面了

firefox立马是ok的，这哥们真是强悍啊五体投地。。。

还可以更改cursor大小，也就是新建一个css文件，javascript真是强大，这哥们神了

问题一个个的解决，爽

ubuntu默认设置的更改实在系统设置里面



我操，这个google chrome不能自动跳转的问题也解决了，网友太强大了，我的搜索能力，关键词选择的能力还是差一些

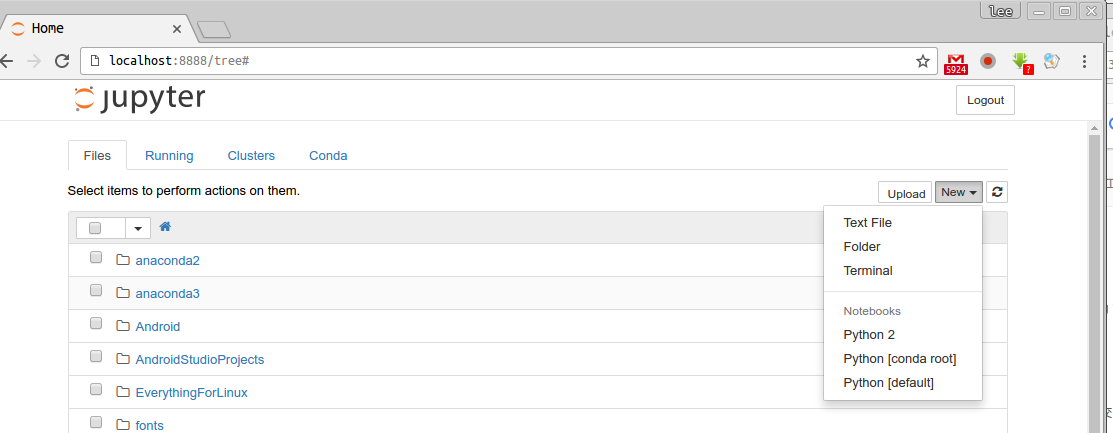
<http://blog.csdn.net/Artprog/article/details/71076111>

亏了这个网友了，不容易。。



新的kernel给安装到/.local/share/jupyter/kernel这个目录下了，两个目录下都有json配置文件了

确实是正确的显示了

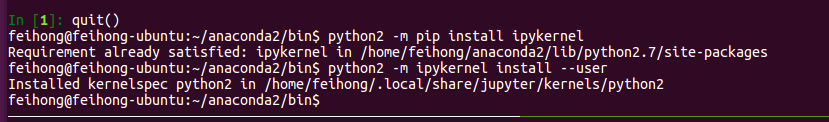


牛逼，终于找到了解决办法，stack牛逼，

<https://stackoverflow.com/questions/30492623/using-both-python-2-x-and-python-3-x-in-ipython-notebook>

链接如下，这两个命令确实有效，键入第二个命令之后会提示你anaconda已经把核给安装好了，然后添加核，切换用户再重启jupyter，果然有两个kernel了，以python2,一个python default就是3了

安装两个anaconda是没有问题的，他会自动的在目录下重命名各自的python为python2,python3，





哦了，暂时找到解决办法了，就是一个jupyter notebook立马可以出现两个两个kernel，要配置起来还是有点复杂的，直接jupyter notebook这个命令，运行的话，只有默认的python3的kernel，但是如果是ipython2 notebook则是有2个kernel可选的，

不过命令过时了。。。

靠，显示是有2个核，但是其实两个核都是2.7的啊，坑啊。，算了不折腾了，实在要用两个的话，分两个命令开就行了啊

安装了jupyter之后，pip卸载果然是不干净的，需要到/usr/local/lib/python相应的目录下去删除

自带的pip的安装路进是/usr/local/lib

cowsay，fortune的有趣小命令

<http://www.cnblogs.com/sukai/archive/2013/06/08/3127031.html>

还是要把源给换成阿里云的，帖子如下

<http://www.cnblogs.com/EasonJim/p/7119156.html>

其实网上帖子一大堆啊，单子帖子立马有一句话说的很重要，

通常更新源之后只需要升级软件列表即可，其余两项小心使用。。

擦，事实证明，往shell脚本上加入 #！/bin/bash这一行的，话，怎么运行都是可以的./sh也ok的，时说之前install-ubuntu这个脚本明明就是有cd，可以直接./sh来运行的啊

又往git仓库里面添加了一点儿重要的文件，比如账户密码，，ros文件等,然后写了个推送的小脚本，很省事儿。。。脚本名字叫copy2git.sh，我把这个脚本也推到github上了，

git add .加一个点号就是表示add仓库目下下的所有文件。还有在shell立马的cd命令，不能直接运行./这种方式，或者bash .sh这种，两种都会新开一个shell，所以导致目录不对，cd失败，但是source .sh或者. ./sh是可以的，两个. .之间有一个空格.

Laptop mode

http://www.cnblogs.com/rancvl/p/5552788.html

我在ubuntu下新建了一个mygitresp的git仓库，这样就可以把要备份的文件，给推送到远端的github上了，很好

2017.8.28再做笔记，原来上次的bundle并没有配置好，是因为最开始的时候clonevundle的时候地址，目录不对，所以.vimrc里面一直报错。。。当时我没有注释掉.vimrc.bundle的引用，结果就没有报错。。。并不是什么单引号，双引号的问题，就是因为bundle更新了，git clone和配置的方法也略有变换，这次我是参照github上的方法，果然ok了，最好是掌握第一手资料，英文原版的最好。。。草，之前看的国内的帖子都太老了，结果git clone地址不对，导致出错了。。fuck

https://github.com/VundleVim/Vundle.vim

终于，按照这个帖子的搞法，终于把bundle给配置到一个vimrc文件立马了

<http://www.jianshu.com/p/mHUR4e>

http://blog.csdn.net/jasonding1354/article/details/45368593

其实和之前配置是一样的，只不过现在bundle命令要换成plugin这个明命令了，更新了的，然后就是plugin 后面接的地址居然不支持””，只支持单引号，这样才不会报错，，坑爹啊。。

udacity的simulator模拟器的下载地址

<https://github.com/udacity/self-driving-car-sim/releases/>

居然还有ros版本的，有趣。。。

原来之前在root的的bash下来source .zshrc会报错是因为尝试着从bash来source一个zsh文件，这样当然是不行的，是需要在root下卡一个zsh再来source才行，因为zsh文件不是sh文件，必须在zsh这个shell立马才能运行，坑爹啊，这个坑业踩了好久。。。。

[**https://github.com/robbyrussell/oh-my-zsh/issues/3691**](https://github.com/robbyrussell/oh-my-zsh/issues/3691)

**https://github.com/robbyrussell/oh-my-zsh/issues/2527**

**下面这个帖子帮了我大忙了**

靠，终于解决了切换到root下无法自动跳转到zsh的问题了，原来是要手动键入zsh的，因为在root的bash立马还没有进行默认shell的切换啊，在root下chsh -s /bin/zsh，再退出再进入，就解决这个问题了，就可以默认来进入zsh了

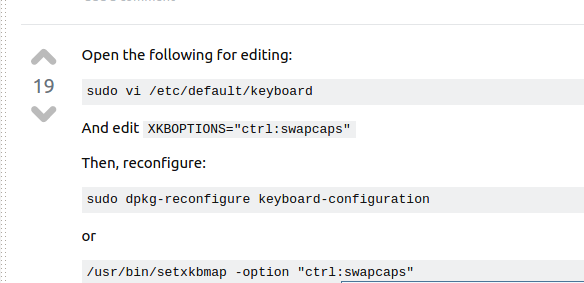
一个解决zsh编码的问题，需要到windows下，进行修改，我先试试看吧

http://blog.csdn.net/dcx1205/article/details/38615127

终于解决了键盘映射的问题了

<https://askubuntu.com/questions/33774/how-do-i-remap-the-caps-lock-and-ctrl-keys>

还是这个帖子有效，下面这种命名真的是有效



zsh立马进行vim映射，这个还是很有效率的,这篇文章里面有代码，在.vimrc立马配置

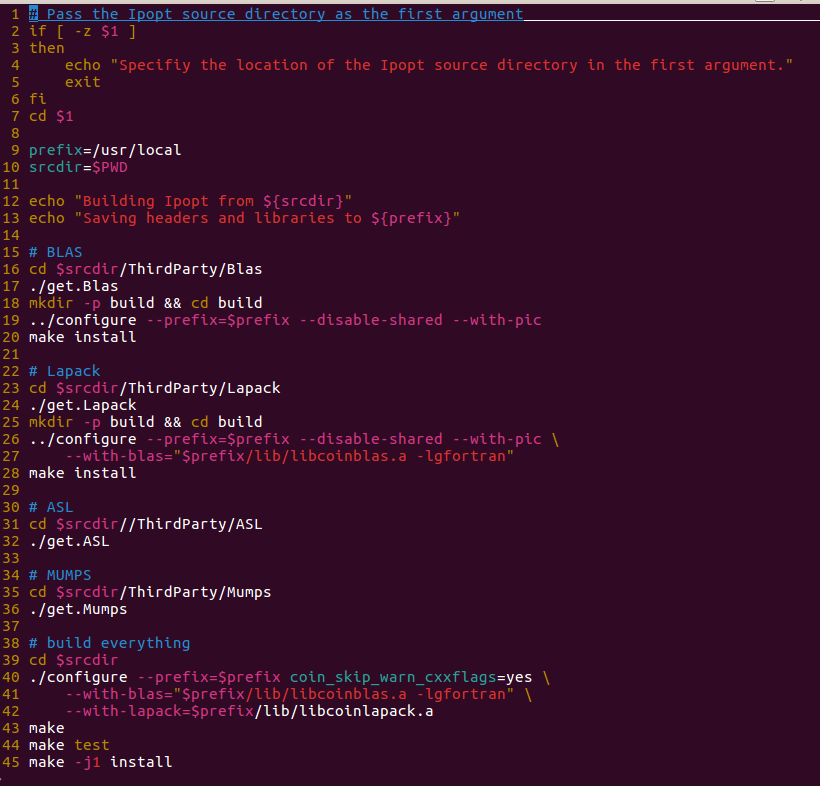
<http://blog.jobbole.com/86571/>

在真正的bantu下，运行项目果然是稳定多了

靠啊，在ubuntu下再次安装ipopt，结果这次安装不成功啊。最后还是有报错，然后直接git clone项目下来，用root，权限，安装了好几次才算成功。。坑爹啊，怎么会这样呢？

udacity的project10，需要安装一些依赖，还好现在是在linux下，windows下彻底搞不了啊

其中要安装一个ipopt,还好有一个脚本，这个ipopt里面其实还要安装很多第三方依赖，ThirdParty，之前下载的ipopt版本不对，压根没有这个ThirdParty目录，一下子没注意，结果按照脚本安装的时候一直报错，报错很长，耽搁了快一个小时，，fuck,,现在下载的是ipopt3.12.8的版本，还是有点大，编译了好久，主要就是项目里面的.configure在起作用，这个configure文件很大，可以加上很多前缀参数



当c代码里面调用readline/readline.h时，首先是需要apt-get 来安装的，Ubuntu并没有自动安装，然后gcc编译的时候要加上-lreadline，不过是要加载.c文件的后面，不是前面，见下图的最下方

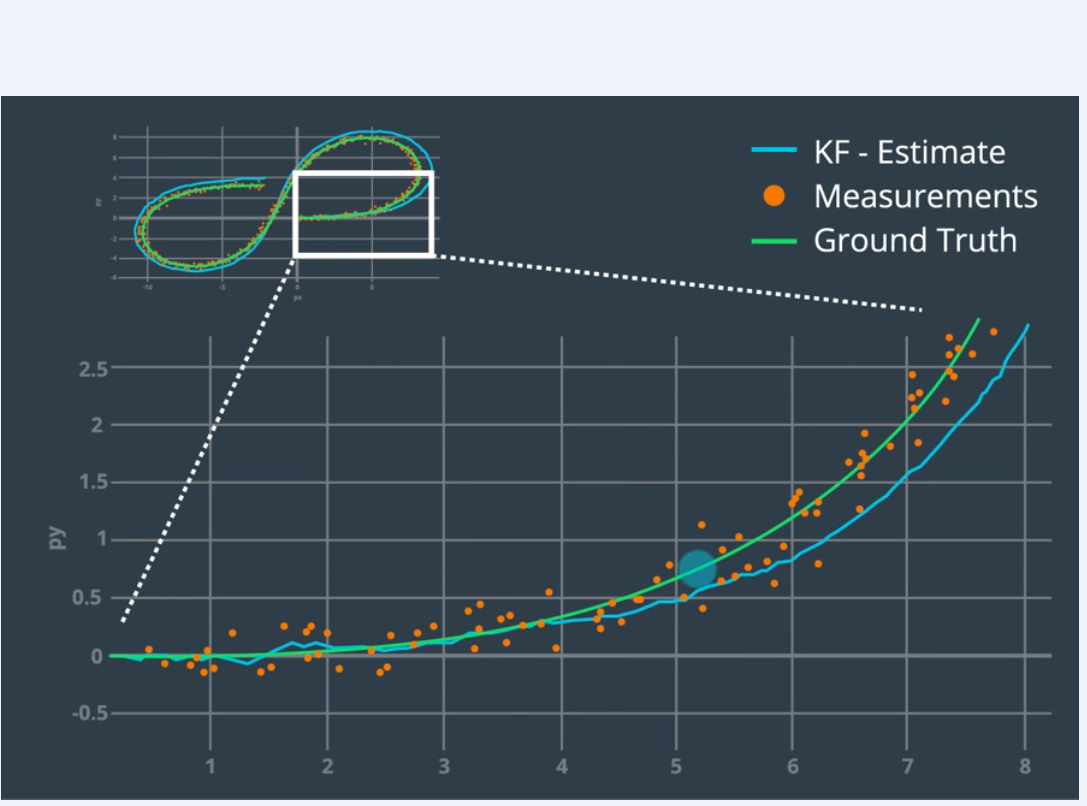


ubuntu向windows传输文件，原来通过xshell和rz,sz通过简单的设置就可以了，下面这个

帖子很棒

<http://www.linuxidc.com/Linux/2015-05/117975.htm>

UKF或者说所有的kalman filter里面都是由measurement来估计出estimate值，然后再去和ground\_truth进行比较，因为measurement是从传感器读取出来的数据，是不准的，所有事需要进行esimate来进行矫正，减少方差，减小不确定性，减少熵。measurement的过程必须是伴随着move，只要有移动，move，就会增加不确定性。estimate的时候则是想办法减少不确定性。而且这里measurement是离散的单个的点。但是ground\_truth和esimate之后的估计值都是连续的



靠，原来今天windows下的内存分配出问题了，只分配了4G，是说看资源管理器内存都是使用90%以上，基本上没开啥东西啊。。。死机了好几次，以为又要重装系统了。。。

坑爹啊。内存对pc的运行速度真是太重要了。。。

作死啊，一定要养成好习惯，常常备份。。

靠了，手贱，把vimrc给删掉了，linux上恢复删除的文件实在是麻烦，只能一步步的再写了，坑爹啊

百度了一下，果然又在各种符号前后加上空格的放在，还是在vimrc里面加入if判断和函数，但是安装网上的方法又是行不通，还是if判断进不去，加了！就能进去了，靠，if判断咋老进不去呢，用下面两个帖子的方法搞的，把内容给粘贴到.vimrc里面

<http://www.cnblogs.com/alwayswangzi/p/6489845.html>

<http://forum.ubuntu.org.cn/viewtopic.php?t=394843>

vim 里面分屏命令，调整窗口大小，这两个帖子讲的

<http://blog.sina.com.cn/s/blog_44c594a70101wur6.html>

<http://blog.csdn.net/fuxingdaima/article/details/8658342>

这个tabular函数定义还是有问题

if exsits(“:Tabular)这段代码进不去啊，是为false的啊

所以，映射和函数都进不去。难道是bundle安装有问题。。。

靠，按照这个帖子

<http://www.cnblogs.com/huanlei/archive/2012/04/13/2445550.html>

用git的方式安装，去virmc里面配置这个这些代码，还是不行。。操

这个老外的帖子讲的很仔细，我也是完全一样配置的(按照视频里面的)但是我的就是不起作用

操在exists函数前面加上!取反，键的映射倒是可以用了，不过那个自动格式化的函数还是不能用，fuck，算了，不折腾了，就这样吧

有用的就可以了,浪费了太多的时间了

<http://www.cnblogs.com/huanlei/archive/2012/04/13/2445550.html>

按照这个帖子的介绍，加入了脚本到.vimrc里面，但是映射有错误啊

vim实用插件推荐2017

<https://zhuanlan.zhihu.com/p/24742679?refer=hack-vim>

这个牛逼

一个关于vim使用的牛逼的国外网站

<http://playterm.org/s/?tags=+vim>

tabular使用教程1

<http://www.cnblogs.com/huanlei/archive/2012/04/13/2445550.html>

还需要添加一个自动对齐的插件才行,就是这个

Bundle "godlygeek/tabular"

代码对齐的神器啊。。。。

现在vimrc文件越来越大了。vim的函数遵循的就是vim的语法，牛逼了

map实在是太好用了。好些capslock不能再vim里面映射。。。

其实这个头文件还有进一步完善美化的可能

用第一个帖子里面的函数还是不太完美，因为如果在cpp文件里面使用#\*\*\*\*，这样添加注释是会报错的，要改成/\*这样才行，所以要经过多次判断才行了，

用这个帖子里面的函数比较好。。。

<http://blog.csdn.net/waleking/article/details/7632743>

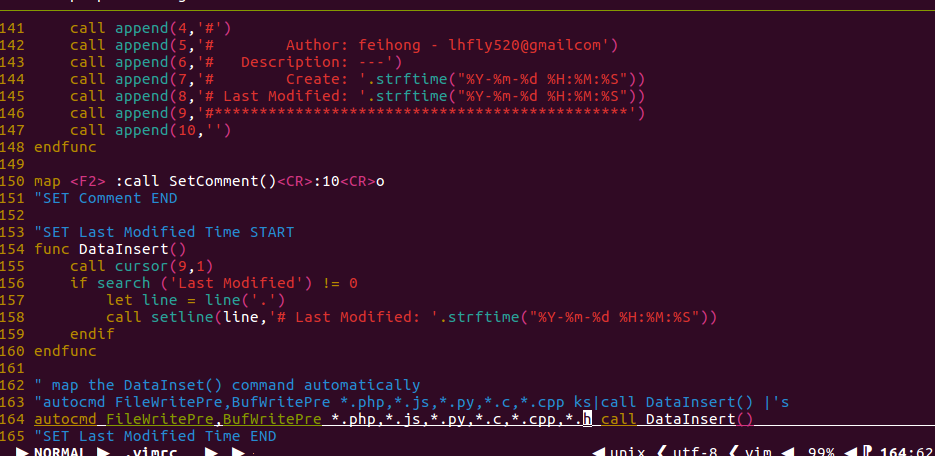
修改时间那个函数是调用成功了，但是第二个帖子里面逻辑是search，每次vim里面:w报错之后都会更新修改时间，但是每次都会调到那个更新时间的的那一行，很郁闷。

按照第二个帖子的方式进行修改，并进行时间格式的整理，果然不用每次保存都跳转了，不错

第二个帖子里面的修改时间函数则不会，不错

但是第二个更新修改时间的函数没有成功调用啊，什么情况。。。

修改了一下调用函数，确实是成功的显示了修改时间了，动态的显示

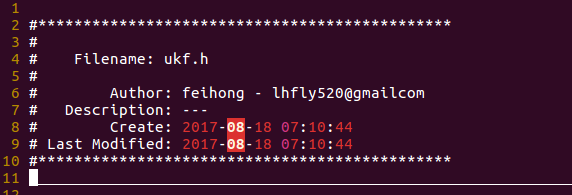


而且把这个功能给映射到F2键了，可以手动的添加

牛逼，又给vim添加了一个小功能

<http://blog.csdn.net/qq844352155/article/details/50513072>

按照上面这个帖子来做的，就是给文件添加文件头注释，这样就越来越像IED了，而且是使用函数来实现的，扩展性很强的



vim代码自动对齐小命令

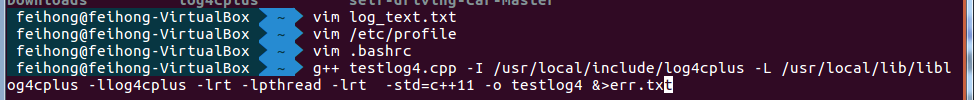
<http://blog.csdn.net/ljphhj/article/details/13316749>

命令就是gg=G,有意思

其实命令还可以更简单一点儿，不需要加上-I ,-L参数，默认会找到，不过自己指定更好一点儿，-I表示include的头文件，-L表示相应的lib



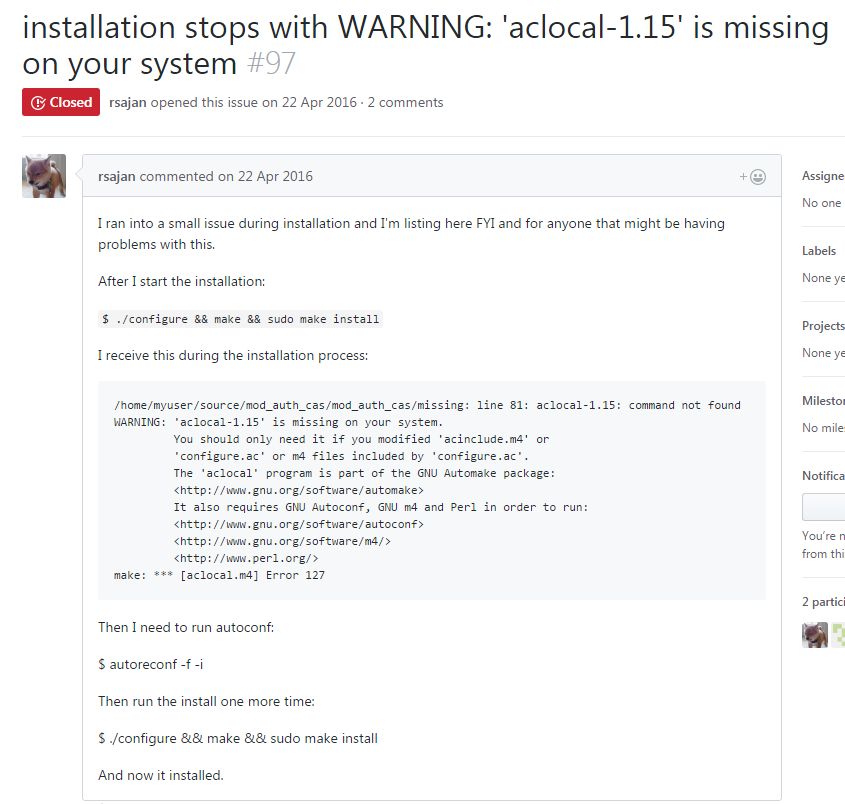
照着这两个帖子，敲了一下log4cplus的demo，确实是可以运行的，实测，有一点要注意的，如果要调用log4cplus是需要指定调用的，所以命令有点复杂，如下，在第二个帖子里面有讲解,而且最后要加上-std=c++11 ，因为log4cplus是要求c++11及以上



<http://blog.csdn.net/yanchen0314/article/details/41114053>

<http://www.cnblogs.com/rosesmall/archive/2012/05/07/2487445.html>(命令解释)

虽然log4cplus安装成功了，但是是安装在/usr/local目录下了，在vim里面无法跳转头文件，虽然我path里面加了这个/usr/local/include目录，还是没有加进去，不知道为啥，但是如果把log4cplus目录拷贝到/usr/include目录下是可以跳转的，/usr/include好像是默认的头文件存放目录



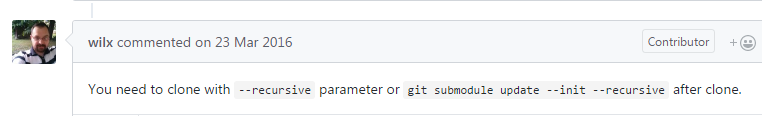
今天尝试着用源码安装log4plus，在Ubuntu上，用gitclone下来

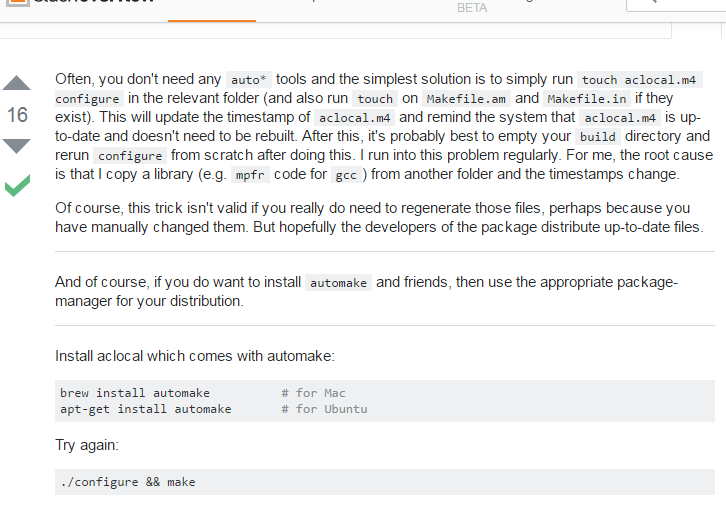
<http://www.cnblogs.com/rosesmall/archive/2012/05/07/2487445.html>，这个贴在帮助很大，但是中途还是碰见一写问题，make的时候报错了，，

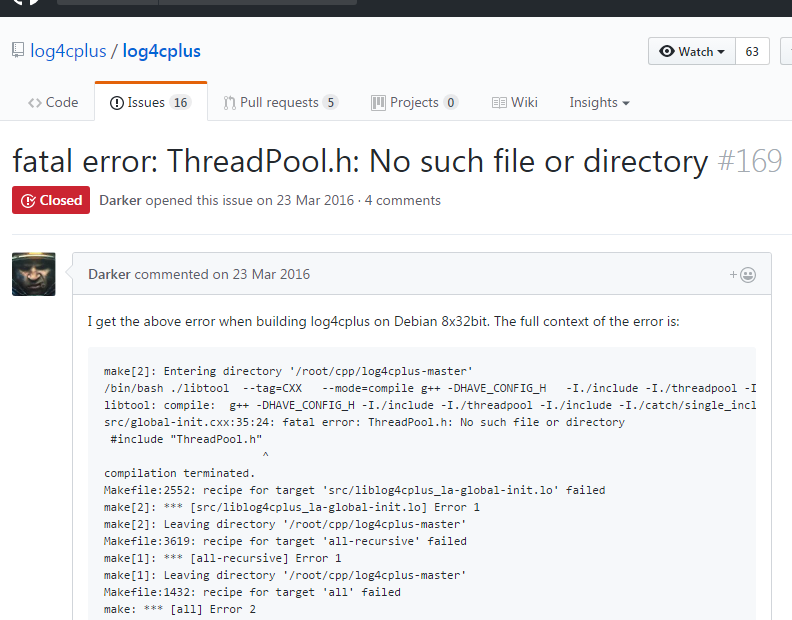
Google了一下，原来是什么时间戳的问题，，还是安装了automake这个工具之后，然后运行autoreconf -f -i 重新就行配置，给一些配置文件重新生成时间戳，然后再运行./configure &&make &&make install ,成功安装，但是make install的时候还是碰见了一个问题，create一个文件夹的时候失败了，permission deny，原来没有加上sudo，然后make编译的时候还碰见另外一个问题，说什么一个头文件找不到，有网友给出了答案，我在clone的项目的目录下，运行git submodule update --init --recursive 果然递归重新生成了目录，再运行就编译成功了，原因还是在git clone目录的时候出现了目录递归的bug

<https://github.com/log4cplus/log4cplus/issues/169>

谢谢github的帖子







又安装了一个自动注释的插件NERDCommenter，nerd系列越来越强大了，vim也越来越强大了

NERDCommenter安装及快捷键

<http://blog.csdn.net/zcube/article/details/42298419>

<http://www.wklken.me/posts/2015/06/07/vim-plugin-nerdcommenter.htm>

其中有更改leader键的映射，这个很有帮助的

NERDCommenter安装之后记得要在.vimrc里面配饰filetype plugin on

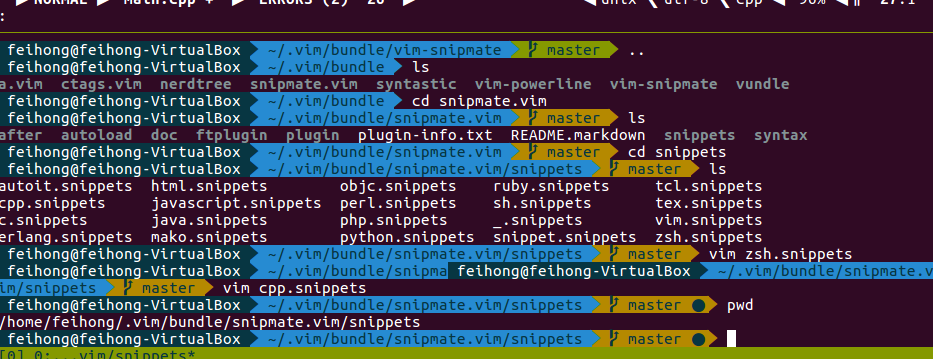
https://github.com/scrooloose/nerdcommenter

这个帖子就是一个自定义的模板

<http://blog.chinaunix.net/uid-29091195-id-3991662.html>

<http://www.cnblogs.com/fstang/archive/2012/12/05/2803964.html>

vim里面一个小插件snippet的目录所在，去bundle插件目录下找，这个插件可以自定义很多模板，可以提高输入的效率，支持各种语言，很赞



gcc编译的时候输出错误太多了，居然用>>来重定向到指定的文件失败了，百度了一下，

居然用&>这种方式是ok的，邪门了

<http://bbs.csdn.net/topics/30445355>

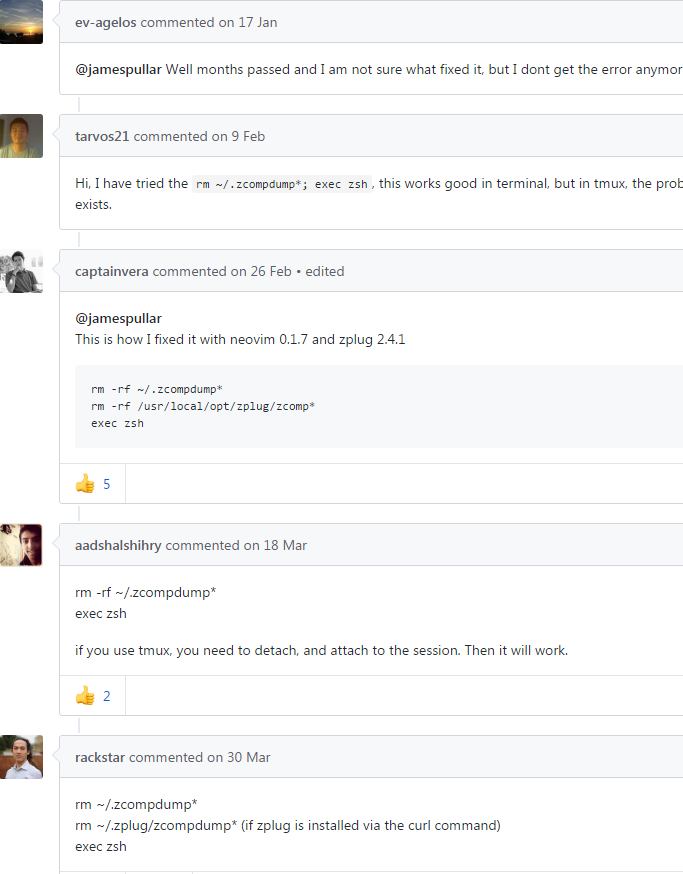
这是csdn上的帖子

妈的，zsh在root用户下，使用有点问题啊，还是环境变量这些东西感觉没有设置好

vim 某个文件，然后tap自动补全，会报一个错

<https://github.com/robbyrussell/oh-my-zsh/issues/518>

在这个帖子里面找到了答案，也是尝试了好多次才找到了答案，需要把.zcompdump想孤单的文件都删除，然后exec zsh，ok了，但是root用户下，zsh有的特性还是没有发挥出来，，，



搜索这个警告的时候ISO C++ forbids converting a string constant to 'char\*'

搜索到下面这篇文章

<http://blog.csdn.net/xyy410874116/article/details/6397549>

真的不错，多各种char\* 字符指针的总结，真的很棒,没想到

这个关于c++里面，深拷贝，浅拷贝的内容很棒

<http://www.cnblogs.com/BlueTzar/articles/1223313.html>

贴住的最后一句话的总结是很好的，简洁好记



这两个帖子给了我启发，我的Ubuntu下只有.profile文件，没有.bash\_profile文件

<http://www.cnblogs.com/imlzk/p/4474356.html>

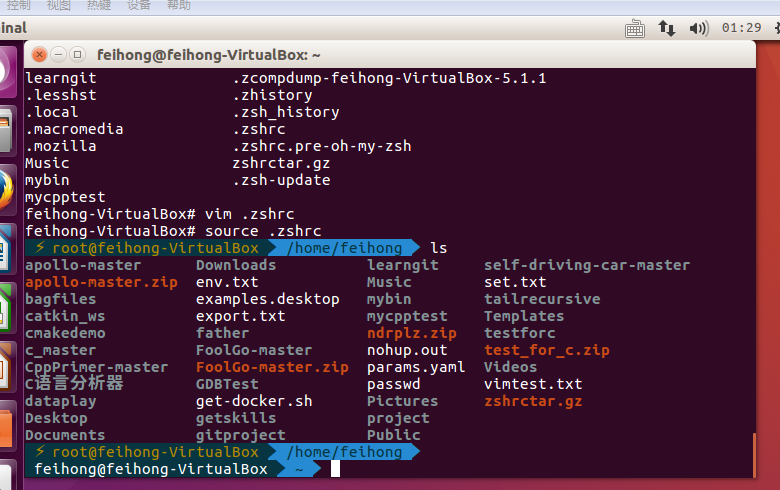
<https://stackoverflow.com/questions/26616003/shopt-command-not-found-in-bashrc-after-shell-updation>

我操，原来是要把.profile的内容给粘贴到.zhsrc里面啊，.profile文件肯定是会执行的，然后再profile里面有关于指向系统自带的bashrc的命令，这样各种环境变量的查找就都正常了

今天Ubuntu切换到root用户下居然没有zsh的显示了，折腾了好半天，，chsh -s /bin/zsh

也不管用，source root下的bashrc，居然还有命令报错，最后是在root用户下souce .zshrc才ok了，，两种bash还是有点冲突的

但是每次切换到root用户都要进行一次source .zshrc，应该可以装到系统自带的bashrc才对啊



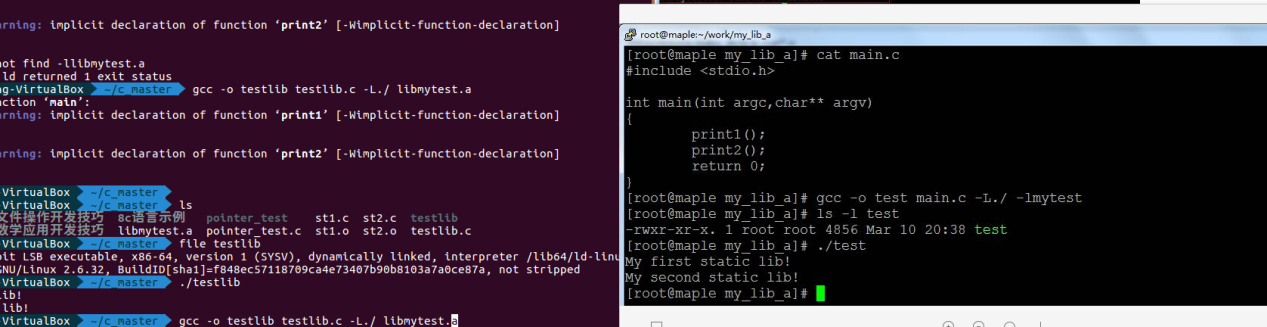
这个文章命令错误还是挺多的，特别是后面的动态库的编译命令，可能是因为他用的是centos，我用的是Ubuntu吧。动态编译so库的时候，用第二种一步到位的命令才能正确的生成so库



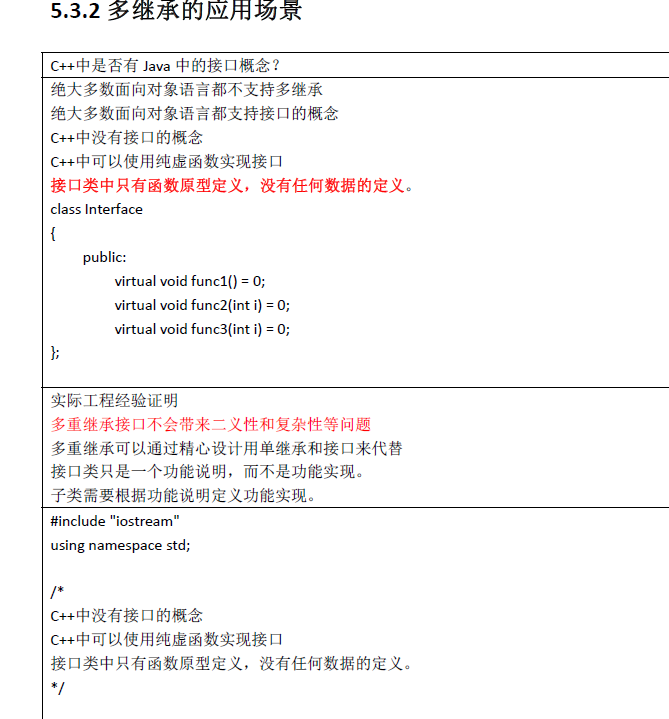
该篇文章的链接如下:

[https://mp.weixin.qq.com/s?\_\_biz=MzAxODI5ODMwOA==&mid=2666540033&idx=1&sn=46161c1502dd44e17e2d47aebb597e6a&chksm=80dce8aab7ab61bce531c67b9883e2060851e26b967230dec668f7b0a6771363dca38e0e68df&mpshare=1&scene=1&srcid=0808MNceiklgPZpXVhdcd5QJ#rd](https://mp.weixin.qq.com/s?__biz=MzAxODI5ODMwOA==&mid=2666540033&idx=1&sn=46161c1502dd44e17e2d47aebb597e6a&chksm=80dce8aab7ab61bce531c67b9883e2060851e26b967230dec668f7b0a6771363dca38e0e68df&mpshare=1&scene=1&srcid=0808MNceiklgPZpXVhdcd5QJ" \l "rd)

静态库编译，作者提供的实例，最后编译的时候有点问题，不能加上那个-l参数，而且库的名称要写全.a不能忘了



c++里面的纯虚函数和java里面的接口作用和使用规则是极其的相似的



常量成员函数



注意，在c++里面，new这个操作符得到的对象是放在对内存里面的，得到的对象永远是一个指针对象，即使下面这样，new出来得到的也是一个pointer对象，而且，new了之后，某个地方肯定要接上delete，new和delete这两个操作符是成双成对的出现的



c++构造的精髓，，父类先构造，先初始化，如果父类构造函数里面有参数，但是又没有提供默认参数，那么子类里面需要显示的再调用父类构造函数，并提供其构造参数



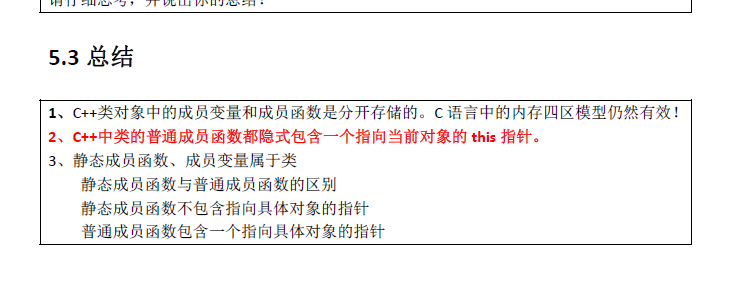
站在编译器的角度去思考程序，牛逼，这样一切都能理解了，编译器的实现肯定有很多数据结构的东西在里面，



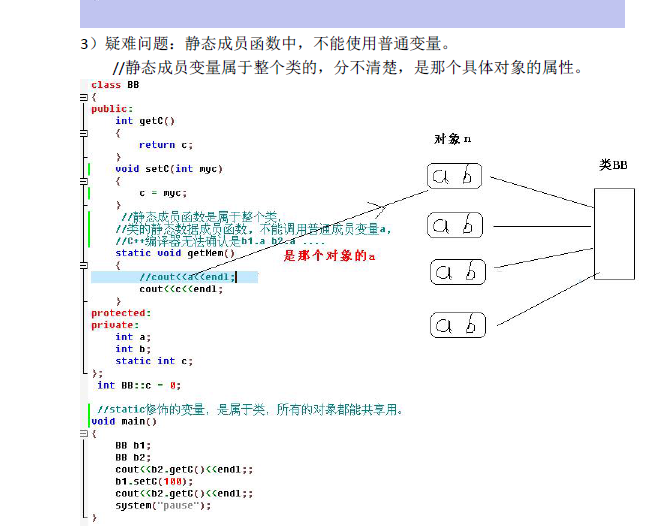
c语言两种方式进行int数组的初始化，new int(8)是数组里面有一个8这个int元素，new int[8]则是初始化8个int元素，申请8个int类型的内存



成员变量和成员函数是分开存储的。要站在编译器的角度来思考问题。。牛逼

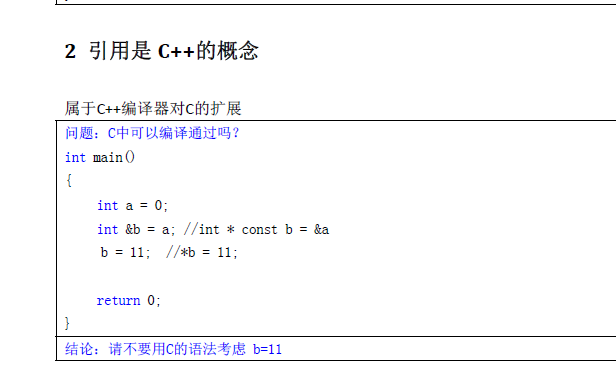


下面的这个点说得真是非常好，静态成员函数中，不能使用普通变量，因为静态成员函数也是所有的对象共有的，如果使用普通变量，不使用静态变量，那么就分不清变量是属于哪个对象的了，所以静态成员函数里面要使用静态变量，也是共有的，属性保持一致。。。





下面这段话对引用的概念解释非常清楚到位，引用是c++的概念，，是编译器对c的扩展，引用时变量的别名，先变成一个常指针，然后赋值的时候就是把指针指向的内容给赋值进行修改。。。



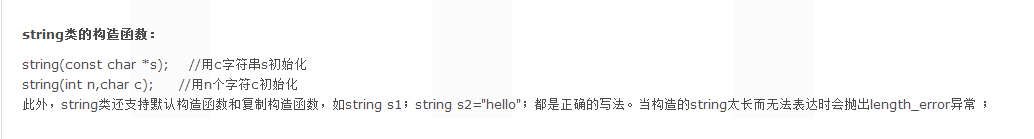
linux下实现windows下c/c++的实现pause()功能

<http://blog.sina.com.cn/s/blog_bad174440101i1a1.html>

ctags安装方法

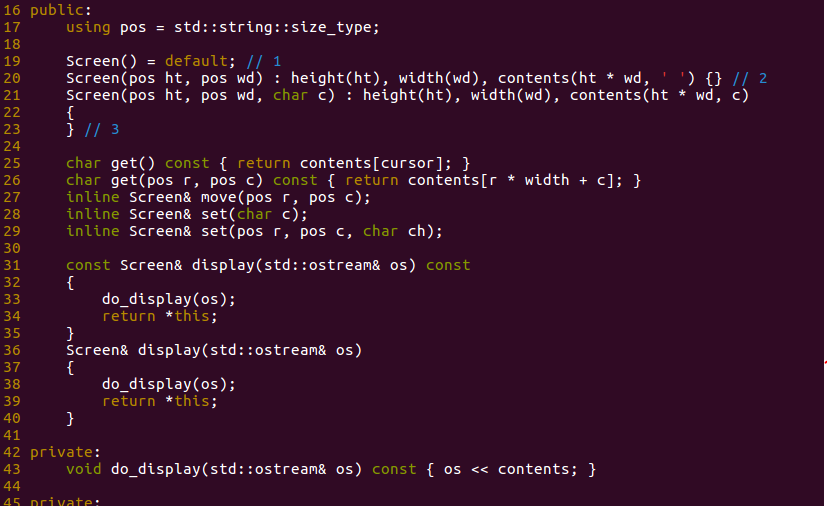
<http://www.cnblogs.com/zl-graduate/p/5777711.html>

这个帖子还是不错的



c++里面的string构造方法，看下面contents是一个string，然后可以(size,’’字符内容)

,不过之前的size是需要先定义size\_type的



c++里面的vector的构造方法，vector(20),就是vector的20个元素，初始值都是0

vector={1,2,3},是直接装入3个元素，和数组的构造很像，数组也是{}直接加入元素



在我的ubuntu下面c++里面的str类型的sizeof是32位，32个字节的

c++for循环有一种简单的用法，基于范围的for循环

用这种方式可以遍历vector，数组里面的元素



这个帖子也讲了一些

<http://blog.csdn.net/yshe_xun/article/details/7315135>

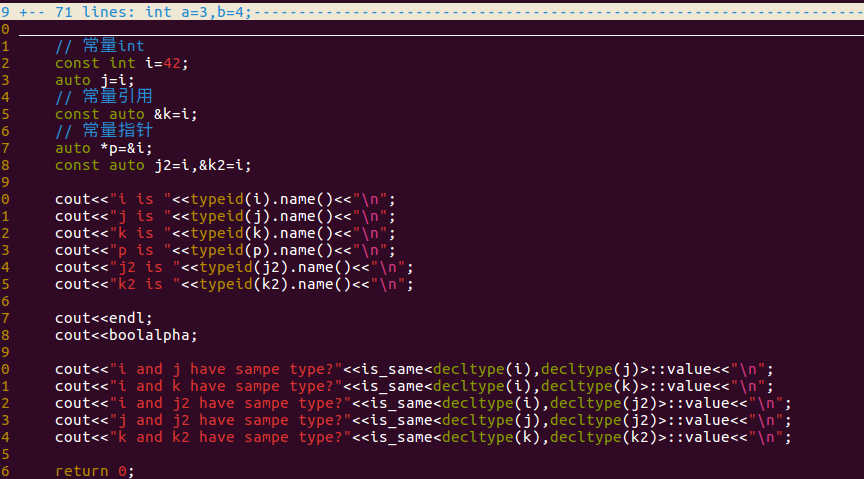
c++还是太不熟悉了。。

这一个帖子

<http://blog.csdn.net/czyt1988/article/details/52812797>

讲了一些

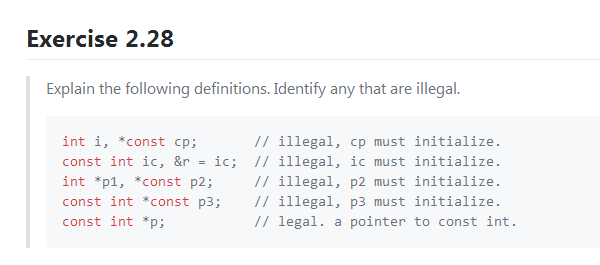
模板的类型判断



下面是指针常量和常量指针的一些区别。指针常量，表示指针指向的是一个常量

常量指针表示这个指针本身携带的内存地址值常量，不能被修改。

常量指针需要被初始化，而指针常量可以不被初始化。。。

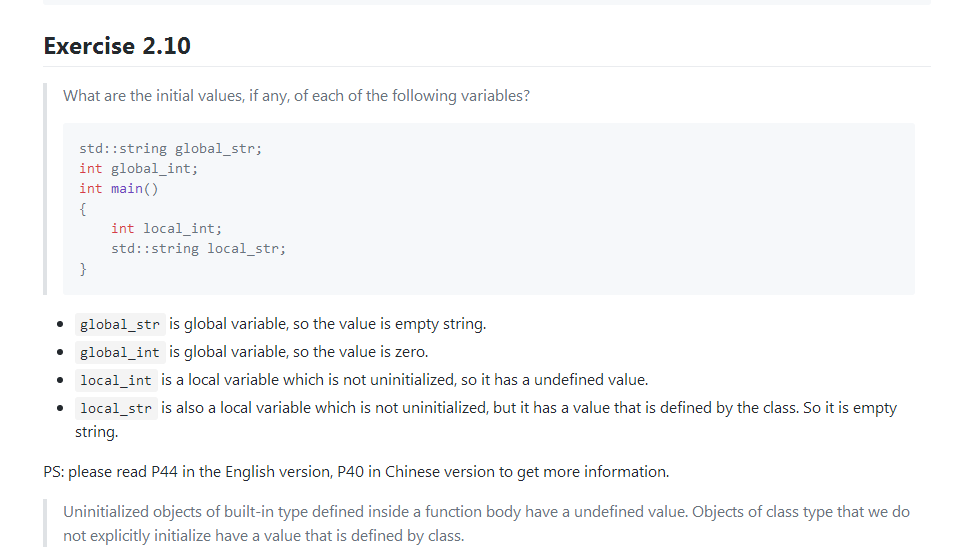


c++里面的引用referrence必须是对一个object的引用

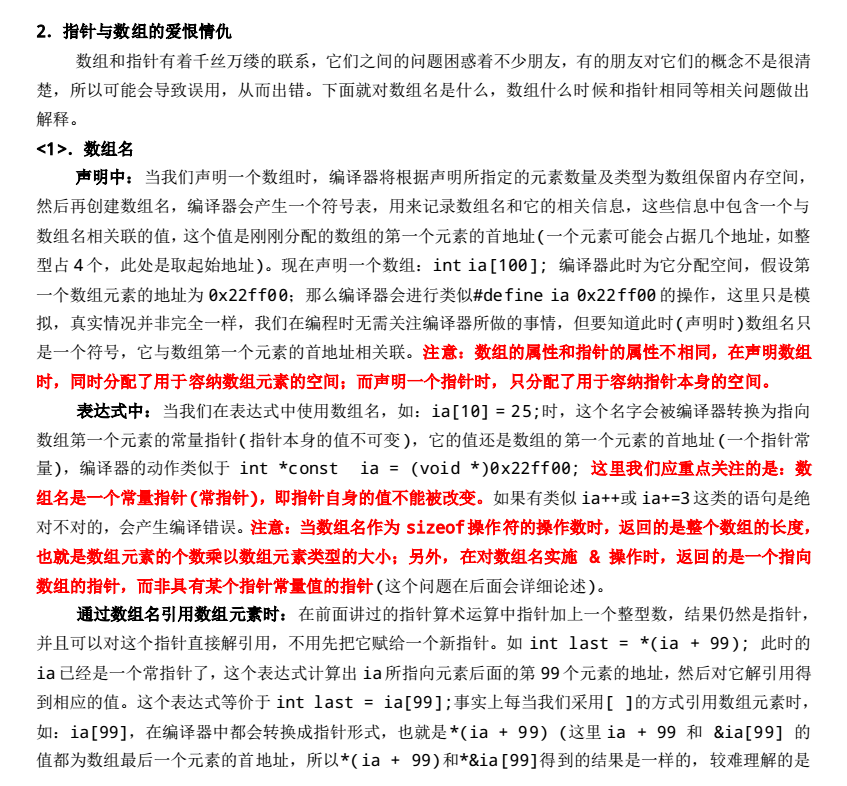
c++里面局部，或者全局的string变量都会被初始化为empty

局部的int则是未初始化，未定义的值，但是打印的时候居然有值，是0

全局的int是初始化为0值



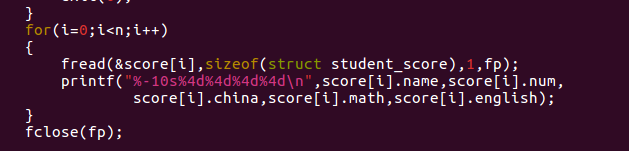
++，+=原来都是赋值操作啊，所以下面的数组名ia++,ia+=3是会报错的，相当于要把一个常量给重新赋值，当然是不对的。但是ia+3则是没有问题的



c语言中几种文件读取模式



-10s%表示s这个字符串占10个bit位，如果不足10个用空格补足



但是在zsh下直接运行这个bashrc会报错的，需要把它粘贴到zhsrc这个文件下，再source，果然ok了。在终端下也可以运行这个命令，但是只对当前终端有效，新开终端之后就不行了。。。

在bashrc下面加入setxkbmap -option ctrl:swapcaps

想不到这种方法比其他两种交换ctrl和capslock的方法都要好使哦

从源码开始编译安装vim的一个教程

<https://github.com/Valloric/YouCompleteMe/wiki/Building-Vim-from-source>

要成功安装YCM，编译安装vim是很有必要的。

g++默认支持c++11的方法，可以取别名。。



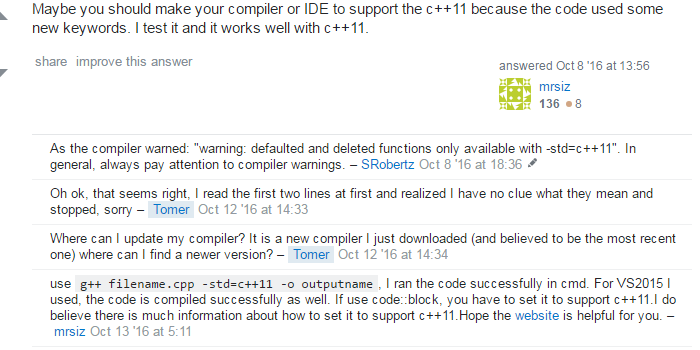
原来vim里面是可以用gf命令来实现跳转的，然后就是ctrl+o跳转会上一次的页面



用g++编译了c++prime里面的代码提示要支持c++11才行，终于在stackoverflow上找到答案了，

<https://stackoverflow.com/questions/39932737/primer-c-sales-item-h-cant-seem-to-complie-the-example-class>

编译的时候要把命令，改一改。。。



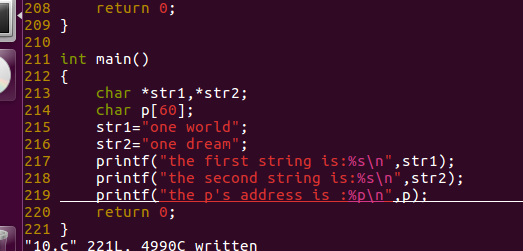
关于syntastic在vim里面的配置，要把，自动跳转给关闭掉，因为如果那里有错误，特别是其他头文件里面，会自动跳转到其他头文件里面，如果不关掉，就跳不回主文件啊。肯定是我没有找到合适的快捷键。。

还有，要在.vimrc里面配置，让systastic支持g++11的语法。。

原来c语言并不会检查数组下标越界的情况，只要有可能内存，即使插入超标的数据，依然可以正常运行。。。。



%p是可以打印指针的地址的，原来指针对象只能这样打印，如果一个变量要查看其地址，当然可以使用&然后再%d来查看，比如一个int a对象，当然可以%d,&a



又是这个博主的一篇贴子，将MLE(最大似然估计),MAP(最大后验概率),

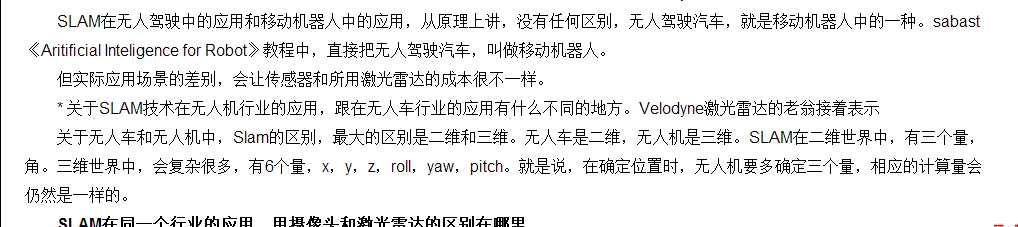
<http://blog.csdn.net/yimingsilence/article/details/70613369>

这个帖子很棒，学到很多东西

<http://blog.csdn.net/yimingsilence/article/details/51701944>

稀疏图，稀疏矩阵，是一致的，graph\_based SLAM,基于图的SLAM算法，很牛逼，graph这种结构对于SLAM真是太重要了

果然，无人驾驶里面是有6个量，在udacity的课程里面也确实是6个量。。。



整个SLAM也是大致分为前端，后端的。。。



Thrun Sebstian的那篇SLAM论文对我来说还是太厚重了，看不懂，后面全是数学符号。。。

SLAM

<https://www.leiphone.com/news/201605/5etiwlnkWnx7x0zb.html>

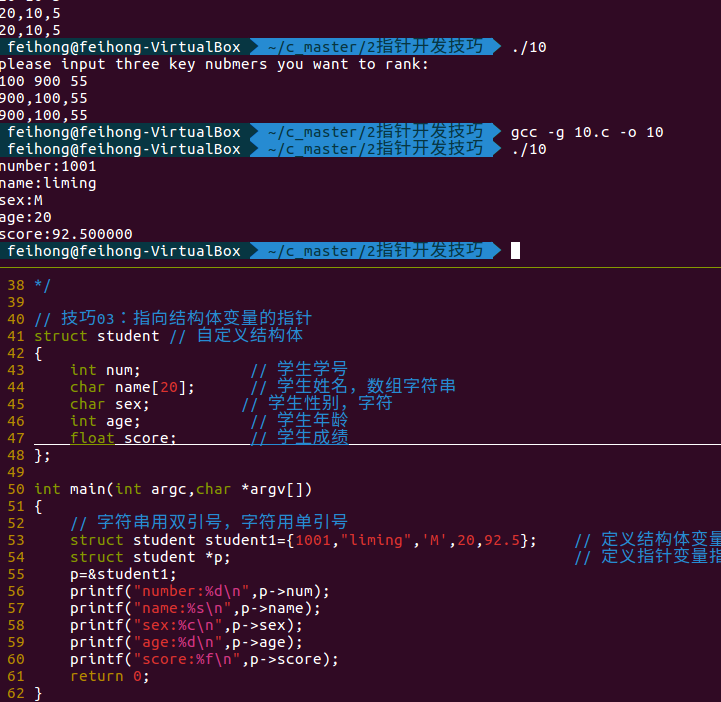
讲SLAM的一篇小文

C语言指针++的问题

<http://blog.csdn.net/kelehaier/article/details/54376772>

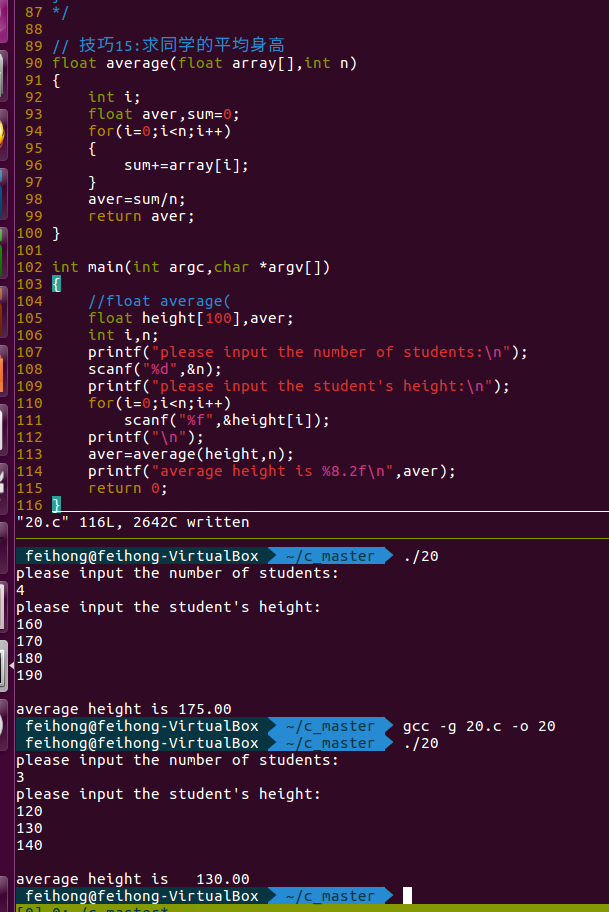
不是普通的变量++，而是根据指针类型来响应的++，看似容易混淆，其实是很巧妙的

这个结构体的小练习，基本上把几种数据类型都用到了，int，float，char，string，小而全

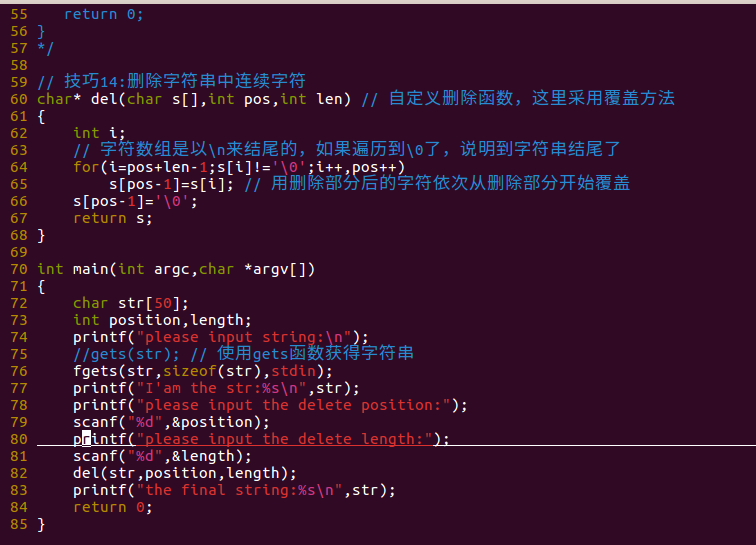


在c语言里面%8.2f这样的用法和Python里面很像啊，.2就是表示取2位有效数字

8则表示这个数字一共占8位。。好几种语言都是这么个规范。。。



c语言里面，字符数组的数组名其实是一个指针，所以，如果下面的del函数里面函数的返回类型写成char的话会报错的，应该是char\*



原来Ubuntu里面自带了xmodmap这个键盘映射工具啊，碉堡了，

<http://blog.csdn.net/iccav/article/details/11706449>

按照这个帖子的做法,直接新建一个小脚本，然后用xmodmap命令运行就ok了，但是不能再shell里面运行，会报错，果然映射成功了。这下手势可以更加的自然了。。。

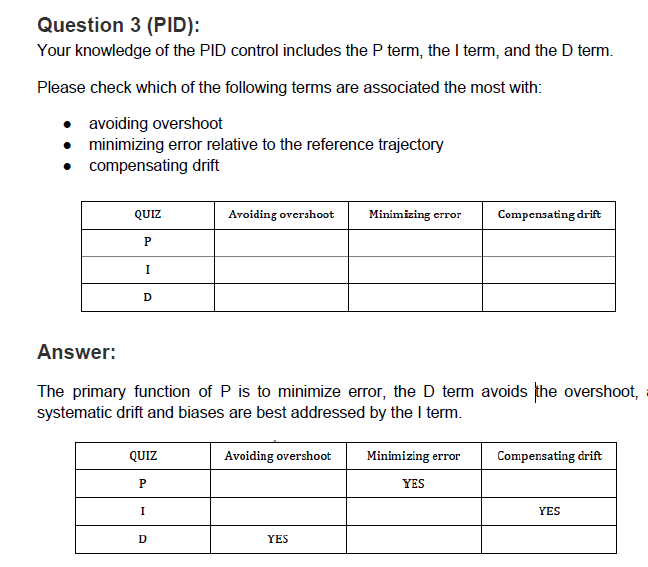
这是最简答的一个教程。。

还有一个帖子更加全面一点，但是我没有尝试，有机会再看看吧。

<http://blog.csdn.net/u014717036/article/details/57082204>

但是其实这特帖子里面的脚本更好一些，上面那个怎么时不时的会失灵，郁闷。。。

喔，现在看来udacity self-drving 的lesson6这个pdf总记得内容很棒，



今后再配置ZSH就按照那两个帖子来弄，还有字体，今天初步取得了成功，不错，不错。。

这下，再把默认shell换成zshshell也没有问题了，难怪，这么多人使用zsh，果然有其妙用，

效率确实有提升的

果然，在.zshrc这个文件里面把原来的source setup.sh文件改为zsh文件再source .zshrc文件就生效了，可以找到ros命令了。。。

原来bash里面也是可以通过set -o vi这个命令来设置vim编辑的模式，但是不能像zsh那样那么方便的绑定快捷键。jj必须不断的按esc要不就是更改键的映射，但是更改键的映射也比较的麻烦。而且zsh里面的jj模式可以直接翻到上一条命令，按k一下。

但是zsh和ros兼容有问题啊，有的命令没有啊，难道是bashrc里面的环境变量设置失效了？？

zsh里面可以使用vim模式，这个确实可以提高效率。。。

尼玛，配置了半天，终于把NerdTree给配置好，纠结，之前，vimrc.bundle里面的语法，有错误,少了一个filetype on,结果syntax on 一直无法起作用，这不坑爹呢么。。找了半天

而且nerdtree在vim里面的命令是NERDTree，并不是NerdTree,,囧。。。

开始要慢慢折腾插件了，一次少安装1,2个插件，不断的熟悉

<http://www.jianshu.com/p/a0b452f8f720>

这个教程还行，就是用bundle来管理插件。。

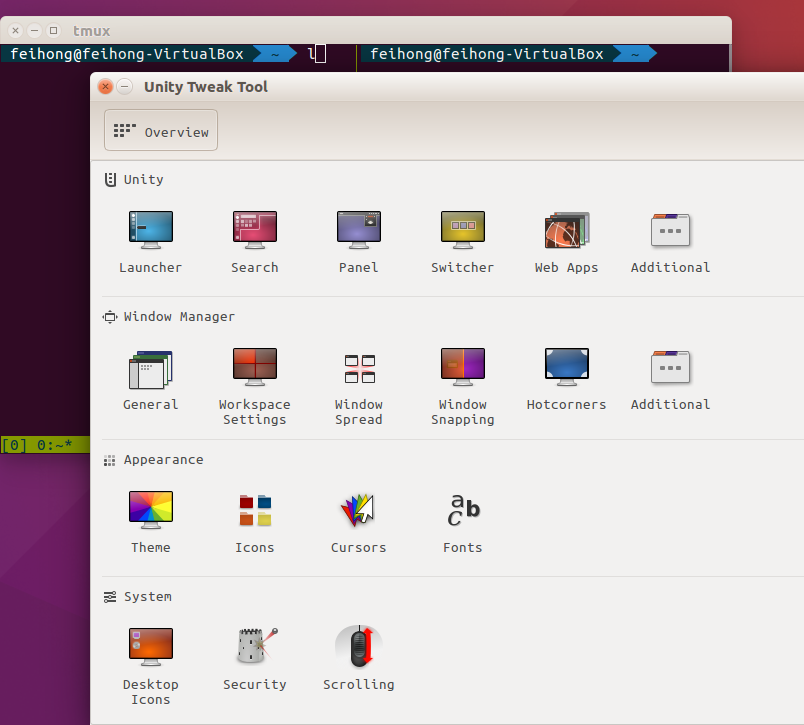
看见很多人推荐zsh，折腾了一下，安装倒是很方便，不过要配置oh-my-zsh,教程 也很简单，碰到设置主题的时候要设置字体，要不然有乱码，Ubuntu下设置字体还需要额外的软件。。。unity-tweak-tool,相当于一个控制面板了，系统默认的setting里面是没有font选项的哦

<http://blog.csdn.net/gatieme/article/details/52741221>

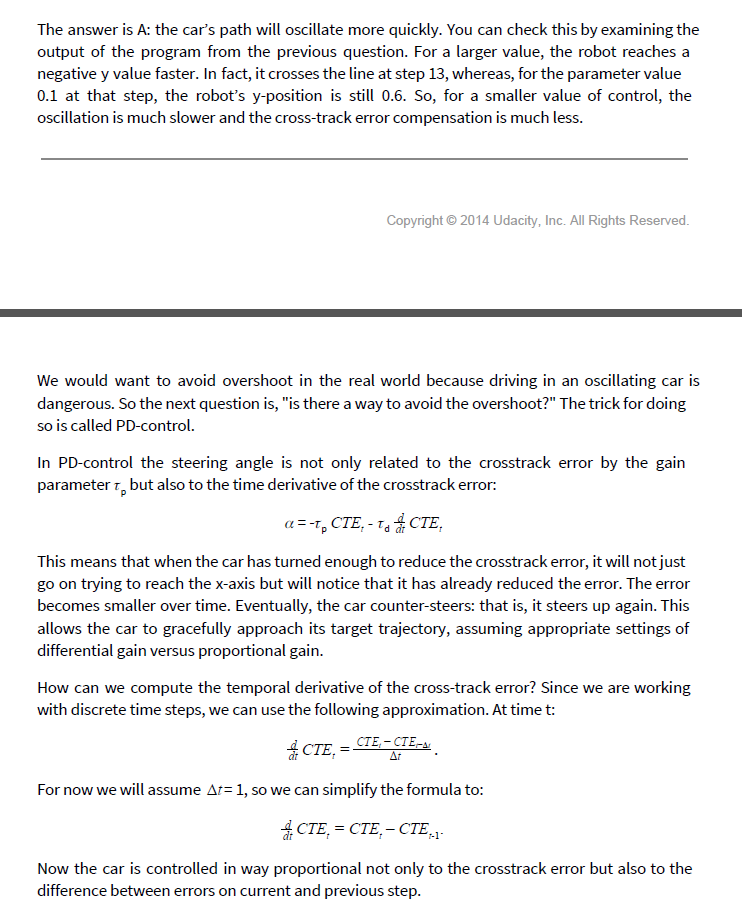
<http://www.jianshu.com/p/9a5c4cb0452d>

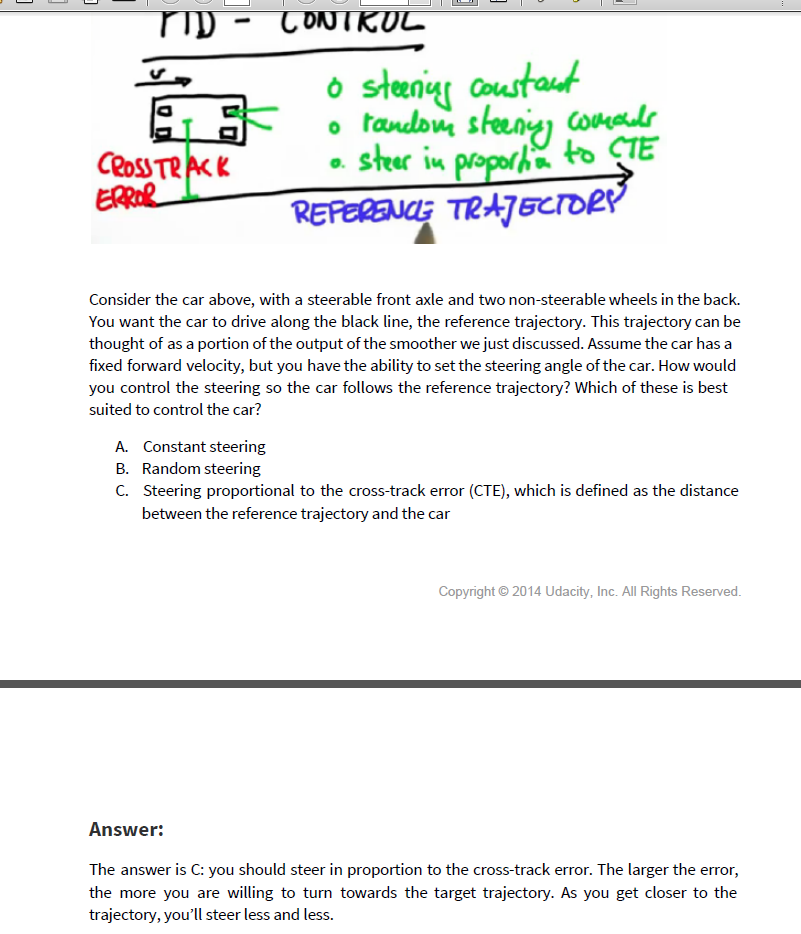
这俩教程不错，主要是安装字体这里，折腾了一会儿。要去github上下载字体，然后install

再配合unity-tweak来选择

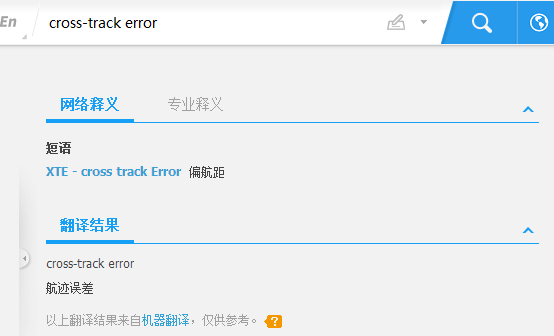


牛逼，这个PID中，单独使用P的时候，效果不是特别好，但是加上D之后，效果呈现4个数量级的提升。D就是error关于t的导数有了这个，当修正路线接近目标路线的时候，导数会记录下来，然后如果震荡过大，那么它会给steering施加反方向的作用，这样，自然就减少了震荡了哦，牛逼啊





这个CTE，偏航距，也就是航迹误差，在udacity的lesson5里面很重要



<http://blog.csdn.net/mr_listening/article/details/51106535>

这个帖子总结得更好，原来尾递归和普通循环是殊途同归，其实两背后实现原理是一样的，而且普通循环不用开辟占空间，效率是最高的。。

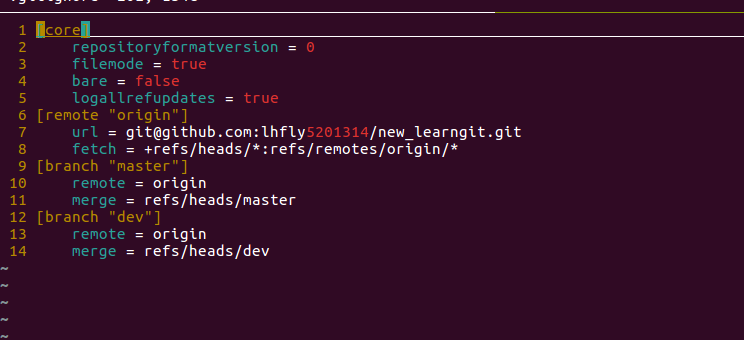
太牛逼了。普通递归，本质上是一棵二叉树，所以空间是呈指数级别来增长得。。。存在大量的元素的重复计算，这个以前在udacity的课程也提到过，如果变成尾递归，那么就从指数增长，变成了线性的，不会有重复运算，就是因为少了重复运算，所以效率大大的提高，少做了很多无用功撒。这个帖子真的很棒。。。

这个帖子不错，讲了普通递归，尾递归，和动态规划的性能比较，其中尾递归，动态规划的性能是远远强于普通递归的，而动态规划中如果把数组声明为全局静态变量，这样就没用引入中间变量了(注意看帖子)那么性能就和尾递归接近了。尾递归主要是只用声明一层变量就足够了，不用保存多层变量，省时，省空间

动态规划用的是迭代，比尾递归需要的空间则是更加的少。。。

<http://blog.csdn.net/fall221/article/details/9156753>

还有配置文件.gitconfig这个文件也很有用，在当前用户的主目录下，我的feihong用户下面有一个，但是root用户则没有，所以我su root之后，相应的git别名都无法使用了，但是如果我把feihong这个用户的.gitconfig文件拷贝到root的主目录下也就是/root之后，这样root这个用户下面，就也可以使用这些配置了，牛逼了哦。而且每个仓库里面都有相应的config文件，在.git这个目录里面，这个config配置文件很有用，可以更改很多东西，尤其远程仓库的选择上，解决了我的一个疑惑。。。



git 学习体会，感觉最后的.gitignore和git里面取别名还是非常的有用的

我把廖雪峰网站上提供的别名都给用上了，还加上了自己的一个别名，叫

git lpo 就是git log --pretty=oneline --abbrv-commit



原来make和cmake之间是有关系的cmake是用来生成Makefile文件的，有了这个Makefile文件，make命令才能正常执行啊，但是Makefile这个文件是要依赖于CMakeLists.txt这个文件的哦。

原来linux shell里面还有这么多快捷键，以前真不知道啊

<http://blog.csdn.net/qdujunjie/article/details/10902507>

<http://blog.chinaunix.net/uid-361890-id-342066.html>

Linux shell里面粘贴是ctrl+shift+v,赋值是ctrl+shift+c，牛逼

原来linux bash里面快速输入上一条命令的方法有这么多，还是！！和ctrl+p最好用。。。牛逼

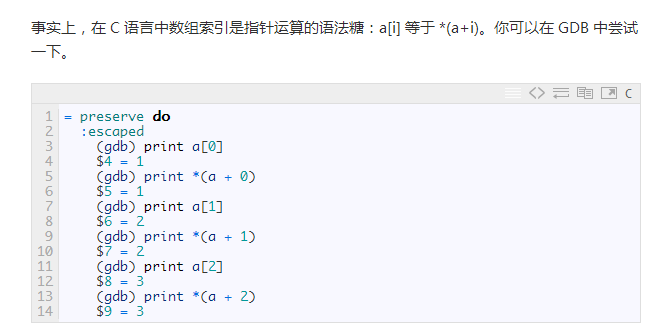


忘了加sudo而要修改只读文件时的小技巧，实用的小技巧啊

<http://www.jb51.net/article/102265.htm>



在C语言中，数组的索引，下标(index)是指针运算的语法糖，只是在指针上加了一个\*



我操，这个讲gdb得帖子有点意思啊

<http://blog.jobbole.com/87482/>

在dgb里面还真是把c语言给当成一种交互式的用法了，这样就和python很像了啊。直接打印查看变量，方便(其实就是一个debug得过程了),略吊啊

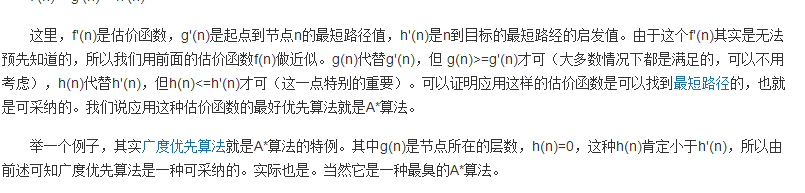
一个学习GDB的好帖子，算是入了个门吧。。

<http://blog.csdn.net/liigo/article/details/582231>

A\*算法效率高，原来是用的启发式的算法。。。牛逼啊

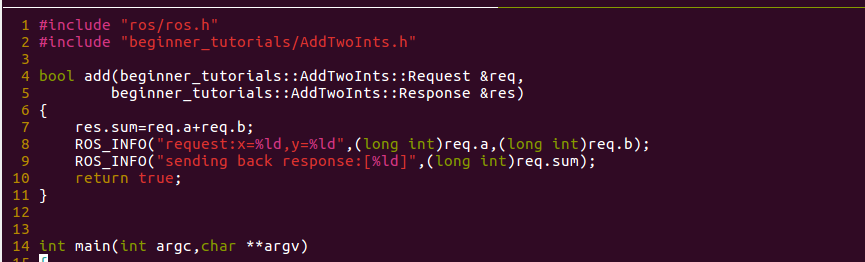
A\*说的启发式算法是要配合h的，这个h就是地图，要配合地图一起使用才能出现优化的结果的

计算最短路径还要一种叫做A\*得算法，但是大话数据结构里面没有提及，这个A\*算法的效果比Drijas算法效果略好，而且广度优先算法是A\*算法的一种特例。。。(o,是说在哪里看到过来着，原来是在udacity得self-driving lesson4得pdf文件里面有讲到。。。



c++里面的long int类型的数据如果要打印显示用的是%ld

long类型的用的是%l,int类型用的是%d



这个在vim里面设置cindent，也即c语言的缩进风格，还是很有用得

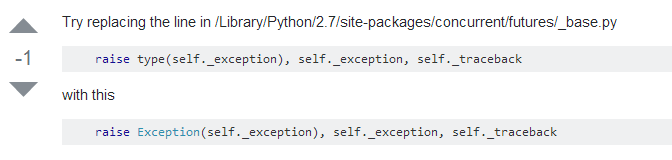


我靠，果然是anaconda的问题，我echo $PYTHONPATH明显的不对，居然catkin空间里面有一个走python3的目录，估计是一开始构建空间的时候就出了问题了。。最开始的时候Ubuntu系统的Python默认是python3，估计构建初始化的时候，就把这个python3的目录给加到空间目录里面去了，导致后面就一直找不到empy了，因为一直走的都是python3的目录了，fuck了，我在bashrc里面把anaconda的两个环境变量都给注释掉了，用了原生的Python目录了，后来运行rqt\_graph,logger\_level等等也正常了，也没有各种报错了，原来一直是anaconda Python path的问题，fuck。。。

再次catkin\_make也没有包找不到empy这个问题了，从打印的日志上就可以看到到走的/bin/python/empy了，果然构建成功了，靠，纠结了一个下午了啊

没想到安装pip3又报了一个错，Google了一下，尝试着修改了一下源码解决了，但是还是心有余悸啊，不知道还会出现啥别的问题，，郁闷。。。

https://stackoverflow.com/questions/44239805/pip-install-error-within-python-3-virtualenv



一个错误把我给卡住了。。。。

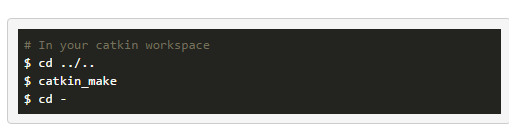
no module name “em”这个模块始终无法导入，问题是出在anaconda身上，还是之前安装了两个anaconda，结果pip并没有重命名，这个问题有点严重了。。。

<http://answers.ros.org/question/39657/importerror-no-module-named-rospkg/>

这个帖子提供了一点信息供参考。。

Google了好久大概就anaconda导致python的path不对，结果包读取不到。。。坑爹。。

cd ../.. 是一下子退2个目录



创建ROS消息和服务这一节里面，运行rosmshow 之前，需要进行catkin\_make，然后再source，要不然rosmsg show 会报错的，而且，msg/Num这个文件里面写入的int64的时候，格式要对，不能写出int 64,分开写就错了。。



确实，，环境变量的改变，或bashrc等文件的改变需要在新开的终端才有小，在一家打开的额终端是不受环境变量的影响的，这几次操作ros的时候就已经深刻的感觉到了

要不断的新开bash才行哦



2017.7.17

今天又碰见了dict.keys()的问题，看来上次其实并没有真正的解决这个问题啊，

Google了一下，还真找到答案了。。。

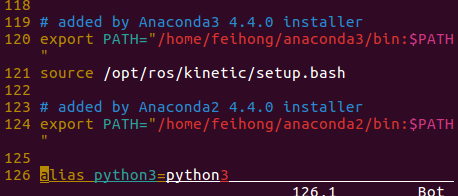
[http://answers.ros.org/question/255218/rosrun-rqt\_graph-rqt\_graph-shows-error/?answer=259605#post-id-259605](http://answers.ros.org/question/255218/rosrun-rqt_graph-rqt_graph-shows-error/?answer=259605" \l "post-id-259605)

这个帖子的情况和我的一样，

就是ros只支持python2.7，但是Ubuntu16.0里面默认采用的是python3.5所以会报错，这里要改一下，默认的Python环境，去bashrc里面alias一下

在最后加上，python3=python3，这样默认就是python2了，python3的时候才会出现python3，

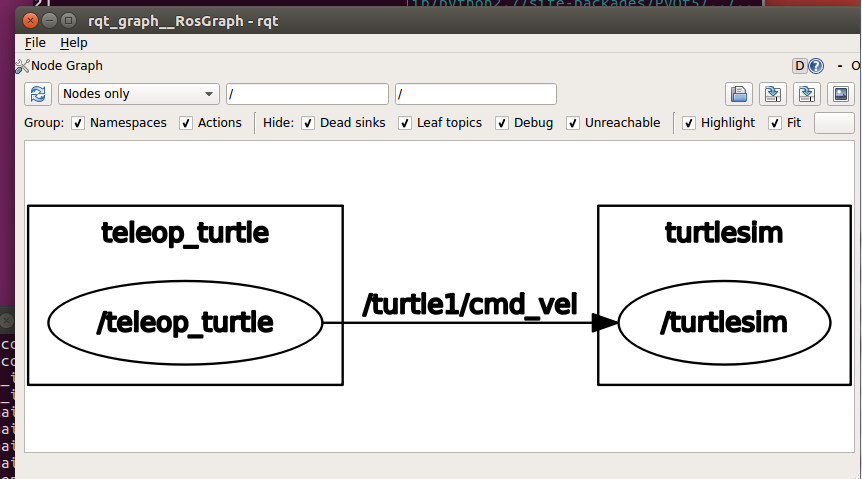
记得source一下，果然不报错了，成功启动了logger



还是在Ubuntu上又给安装上 anaconda2了，因为ros-kinetic好像还只支持python2，它里面的一些lib包都是依赖python2.7的，所以我尝试启动rqt的时候报错可，少了几个依赖，安装ananconda2之后，anaconda2的环境变量自动给加到.bashrc里面了，但是由于两个conda命令有重复，所以，还是像windows里面那样，把conda2里面的conda命令给改为conda2，然后记得，一定要source .bashrc，要不然添加的环境变量是不会起作用的，linux下安装anaconda其实比windows下还方便，但是我其实有点做的不太好，我anaconda的安装目录没有放到opt这个目录下，看来对linux的目录系统理解得还是不透彻，执行得不规范。。

# 其实之前报的错的时候是报的那个dick.keys()不能当作index的错，这个情况之前其实碰见过蛮多次了，显然是用了Python3里面的dict，而没有用python2里面的dict，是这个给力我提示了。。

运行 rosrun rqt\_graph rqt\_graph命令之后终于是显示了这个图形界面了



这个cmake教程的小demo敲完了，真的是很好

<http://www.hahack.com/codes/cmake/>

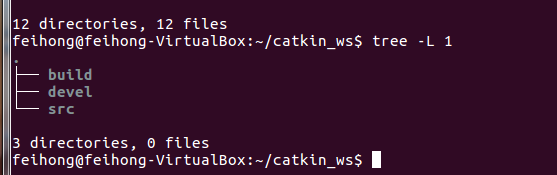
实际开发中可能用到的情况都涵盖了，关键是提供了源码，我完全可以跟着操作的那种，太适合我这种菜鸟了，牛逼了，，

过阵子再练一遍，

再次联系cmake教程里面的demo5的时候，结果我拷贝了一些demo4的文件，导致一些目录都是demo4的，所以一直install不成功。。。fuck，偷懒导致花费了更多的功夫了。。

还有make install 命令最好加上sudo，因为有些文件的赋值是需要sudo权限的

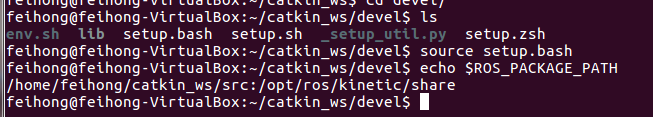
tree -L 数字，数字表示要选择显示的目录层数



昨天到今天一直是设置ros的工作环境不成功，原来是有一个包没有安装成功

<http://blog.csdn.net/yguoelect/article/details/52887402>

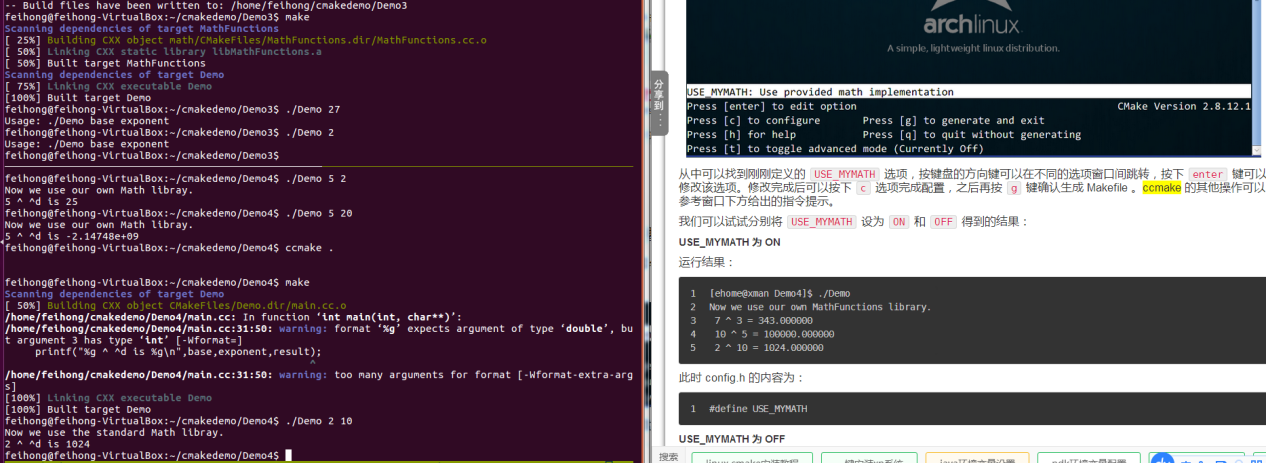
按照这个帖子说的，用pip居然成功安装了，用conda反而失败，提示conda里面，没有版本匹配的安装包，安装成功后果然成功生成了devel文件夹了，里面的配套文件也是正常生成了，然后就是source文件，我source setup.zsh会报错，source setup.bash这个是ok的，任意一都是可以的，还有一个sh，这个和当前电脑使用的shell有关，实验楼实验环境使用的是zsh，我使用的是shell，所以sh，bash的都是可以的，zsh的反而不行。。。



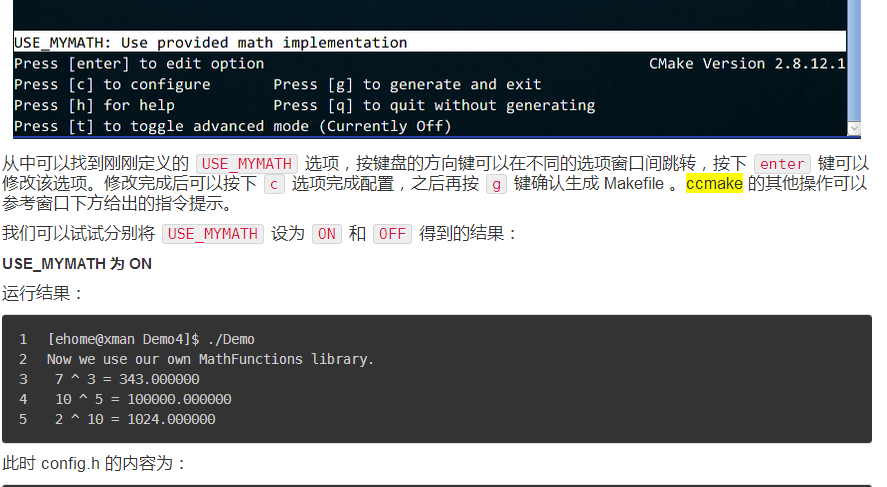
可以看到linux里面环境变量之间是用:冒号进行隔开的，而windows里面则是用;分号进行分割

这个cmake博客教程真的太棒了，实践性很高啊，值得练习。。。

实现了ON,OFF的切换了，就是在edit option里面按enter进行切换，牛逼，确实是可以选择使用不同的define的函数库，生成不同的config文件



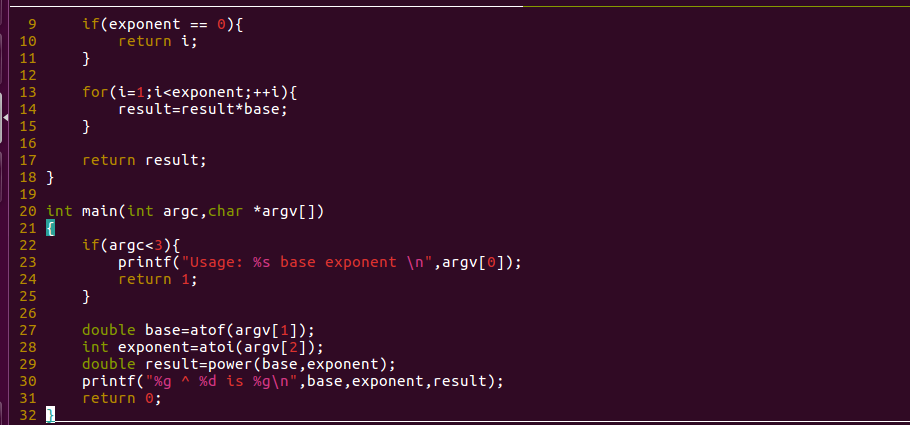
牛逼的ccmake命令，第一次使用的时候要按c，进行配置，要不然，会无法更新一些选项的，configure配置完成之后就可以进行g，generate了，确实是生成了相应的config.h文件了



这个cmake的教程很棒，如果有多个目录，每个目录有需要建一个CMakeLists文件，然后每个目录都需要cmake . 和make一下，这样，library的依赖就不会引用出错了。。

c语言里面的从命令行读取参数也挺有意思的，第一个参数就是文件名本身，好像基本上所有的编程语言都是这样的，，吼吼。。

原来c语言里面的double类型的数据打印是用的%g啊，int类型的是用%d，char类型的是用%c



这一步有错啊，少写了一个步骤，创建工作环境，是有一个init命令的，但是实验楼上没写，导致我的目录下没有cmake文件，结果cmake就一直报错。。坑爹啊，还好再这个网站上查到了，之前在Ubuntu16.0上安装ROS也是靠的这个网站，因为实验楼上是基于Ubuntu14.0和老板的ROS，我的Ubuntu虚拟机上不能实践。。谢谢博主

<http://www.cnblogs.com/liu-fa/>



