1

好，大家可以打开现在目录下的一个叫py\_intro，然后我们从第1节开始，语言编程环境。第一课，有谁知道Guido Van Rossum？python之夫。谁知道他名字怎么念。怎么念？这看起来不像一个条件，

英文名，就是他的名字有点像西班牙语，。。。。。我念的其实不对，但是大部分英语发音的人念他的名字念其实不是这样念的，就是他，他是python作者，然后，我这个，这个notebook里有很多例子。

这些例子有一些我会在课堂上带大家看有一些你们下课了自己去看，有一些很长，然后，这是他的主页，他的主页在github的io上，然后，你在他的这个主页里面你往下你可以看见这个东西有一个soundclick

这是一个他念他自己名字，对，他其实念起来有点像（hu），对，就是荷兰语发音，对，不是很好念，这是python作者，然后，我们这个环境里面七七八八的link，各种各样的link，那么，你们自己用的时候

你们可能注意到了，比方说，额，它这里有个Help，看的见吗，就这里有一大堆reference，有一大堆的reference，那么，这是你的第一个可以查阅的地方，因为它很快，就是我们课上用到的所有东西在这里

都可以查得到Python，Ipython，SciPy一大堆的东西，然后，后面我会告诉你们更快速的查阅这些文档的方法，好，回到python作者这里。看下这个东西，这个大家应该都访问过吧，这就是python的官网，我们访问表较慢

我一会会给大家放一个video，这个video是谁呢，是Peter Norvig 在interview这个Guido Van Rossum.,Peter Norvig是一个很有名的人，应该说我们这个行业里的大佬，然后，他其实也没有念对这个名字，我们可以听一下

（视频 ）我们听最开头的一段，你可以听到他的念法就是一般美国人会念他的名字的念法，他其实也不理了，也没有纠正，然后我们听他下面讲的。就是他刚才讲的这个语言这个语言的abc就是python的前身，

他做这个东西是为了教学的，就是为了教小孩子写程序的，所以，从原图上来说呢，这个语言非常简单，他就是为了大家可以很容易的去教，很容易的去学，然后他讲一些这个语言设计的想法，然后他就讲了一个他自己的困境，

他说我想做一个脚本员，我为什么要做一个脚本员？因为我那个时候，那个，os的脚本环境不是很好用，但是你也可以用c写，用c写，用c写你什么都可以干，但是c很困难，然后写出来的东西，就要很久才能跑起来，有可能你写到最后，最后发现，

我的数据结构不对，然后就要从头再来，整个开发的效率变得很低，所以他决定搞一个中间的语言，所以我们的一些印象就是，他应该比较本强大一些，但是他要比c弱，这一点是非常重要的，尤其是我们这节课，我们用到最后，大家感觉自己在用python写

machine learning的东西，在写deep network，其实不是的，其中大部分code都跟他没有关系，他最终都会掉到python的c的里面当中去，这是真正heavy，真正dirty的work，其实是在python底下的那层东西，我们现在只要记住这点，但是，我们为什么不直接去写c呢？

因为比较麻烦，我们可以谈一些比较有意思的事情，Peter Norvig，这个人就是Peter Norvig，然后你看他的block，他其实很有意思，他很早开始就是做AI的，他80年代的时候就在做AI，然后后来AI在80年代就做不下去了，然后这个时候就叫AI Winter，然后，他去干什么了呢？

他就去写浏览器，他有个手下非常有名，叫恩斯特，他是写scheme的，这是第1个浏览器，他的后面研发了什么火狐，然后，Peter Norvig他，就是影响比较大的事，其实是他去Google. 他去Google呢，他是负责Google，就是整个research和engernning的AP。

就是Google呢，它和其他公司不一样，比方说和Microsoft不一样，Google的工程师和研究员是不分的，他没有独立的两种tite，这种人叫engineer，这种人叫researcher，他们都是混起来的，然后，Peter Norvig在整个Google.setup一整套technic的infor structure，

就比方说Google所有code大家都可以access的，大家用一个code base，然后，比方说Google他会设定那个时代，一共有4个主力的开发语言。

2

唉，比如说我喜欢这个语言，我用这个来，其实他最强的是linex语言，然后，他比较有名的就是，有人说，我们在google搞一个Linex，然后他定了4个语言，是哪4个呢？就是大家会比较，c，c++就是

google比较底层的语言，然后，python，python是在Google那边一开始是被当作脚本使用的，然后后来，machine learning，deep learning出来以后，大家发现可以用python来写一大堆的东西，然后，

java，java这个东西，在最早的年代里面就是用来写客户端的，然后，现在呢？他又发展到APP理，比如说安卓，安卓的应用程序就是用Java写的，然后，还有一种语言，非常重要，是JavaScript.

Google用了非常大的力量在浏览器里面，那个Chrome，在大概2008年左右，忽然崛起的一个重要原因是，Chrome自己做了一个很棒的JavaScript的引擎，JavaScript的第1个GIT的引擎，然后，我们看到他，

对于写程序有什么看法？她对写程序有一段很精辟的论述，我们可以看一下这个链接，我把它总结为不要着急，他为什么写这个东西呢？有一阵有一些书很流行，什么21天学会c++？之类的，21天学会c加加是什么呢？

其实你想想我们课不是21天学会c++，我们是两天学会Python，好像也就是他这个例子，就是你不可能24小时内学会什么东西，你这是胡扯的，然后他讲了一大篇，这个文章非常有名，有好多翻译，他说，

如果你想要学会编程序，你需要用10年，Teach Yourself Rpogramming in Ten Years。然后10年，这个数听起来好像是胡编的。不是的，他这个数据不是胡编的，他是怎么来的，他是怎么算出来的？要10年？

这个人的书，他只要写书就一定是New York. times出版的，很早时候买过他一本书，但是我不知道他是谁，后来我又买了他一本书，我后来发现他俩是一个人，他还在反复出现，他每过几年，就会有一本，

他写的一本书就是它Peter Norvivg一万小时规则，什么意思？就是你想掌握任何事情，我说的不对啊，你想变成一个领域里面的超级高手，你需要1万小时，大家可以想一想，1万小时是一个很大很大的数，

假设你每天花两个小时，你每天拿出来单独两个小时干一件事情，你需要付出额外很多的努力的，那么你需要5000天，5000你除一下，不是10年，大概是10年15年，非常辛苦。就是如果你看他的说的话，你会发现，

你根本没法去驳倒他，因为他非常雄见，他举了很多例子，他举了一个Beatles.，是一个乐队，Beatles在回英国前，他们在汉堡，在演奏啊，演唱啊，不停的打磨他们的技能，然后他说他们花了1万小时，所以非常的了不起，

然后第2个人，比尔盖茨，比尔盖茨为什么符合这个数字呢？1万个小时，他说比尔盖茨13岁的时候，他家里面有条件给他搞一个计算机，比尔盖茨是1955年出生的，他13岁的时候是1968年，1968年对于大多数人来说，

根本就没有听说过计算机，1968年还是大前期的时代，所以他差不多用了十几年的时间，把自己变成一个非常skillful的人，我在微软听过一些故事，就是他能记住自己十几年前写过的code，而且扣的时候非常小，

12岁，13岁，也符合这个规律，反正他举了一大堆人的例子，但是呢，你知道就是一旦提出一个这样的观点，就是说，1万小时才行，越会有人challenge你，为什么，我就觉得你这一万个小时不需要

我们看一下这个1万小时天才理论，他这本书，刚才Peter Norvig讲的outlier，这个作者。然后我们来看一个质疑，（视频）我觉得他这里非常好，就是因为大家focus非常容易聚焦在数字上，就是你说一句话，

然后这句话里面有一个数字，大家一般就会去记那个数字，所以，大家只记住了1万，而忘记了1万什么？他argue的点就是1万个小时是做世界冠军，但是我不需要做世界冠军，我只是想学一些我感兴趣的东西，

我甚至不需要是个专家，我只要自己感到很happy，自己感到很fine就ok了，但是拿出1万个小时来，你算10年15年20年，很多人就被吓退了，哎，我好像我年龄还够吗？我也不是十三岁就开始access computer.

我还要学programing，他argue的点其实是，他怎么challenge这个理论？他challenge这个理论其实在传播过程中，大家对他的误解，一个误传，这个理论就变成了1万小时，你无论干任何事情都要1万小时，

不是的，其实你不需要1万个小时，然后他在后面就提出来一个，一个数字也很夸张，我想该哭的点在这里差不多都讲完了，他说20个小时，我们在课堂上差不多就有12个小时，我们课下还要再练。好不管怎么样，

我们可以开始start了，20个小时是一个很好的，它不会引起你心理上任何的压力，20个小时很cheap，我就是浪费20个小时，我经常浪费20个小时，20个小时就是两三天的时间，这是一个周末，你把一个周末，

全部用来试一试。我们再来看一看这个谁，Guido，他在Hithub上面有账号，你们回去可以点一下这个link看一下，看一看，github上会有什么东西？推特上，也更新得很勤，他在flow上也回答问题，但是，

关于Python的问题，他不在flow上回答，他在flow回答的都是谷歌产品的问题，然后我呢写这个教案的时候，我就去看了一下他这个推特的账号，发现一个很有意思的事情，我发现他更新的很及时，他这个在说什么？

这有个前两天比较热门的话题，它上面这个图很有意思啊，为什么Microsoft? 这个是Github先爆出来的事，因为这些程序员在Github上commit，然后github是Microsoft的产品，然后这有点，这个跟Microsoft没多大关系，

但是你再仔细看，看这里，南方早报，南方早报是马云的报纸，所以就是这些媒体啊，贴了个这样的图片，其实是别有用心的，你看这个争论，起始点是马云的瞎放炮，然后看不到马云，东方早报搞了个Microsoft.

就好像Microsoft在搞。我们回到正题，Python,是我们是不需要知道任何东西的，我们马上就可以开始写code的，但是在写之前呢，我先跟大家强调一点，这一点，一定会让你很难受的，你一定会遇到这个问题，

这个问题就是python的版本，可以用一个字来形容，就是乱，非常非常的乱，我也不知道怎么回事，可能跟他的communicate很有关系，它主要有两大类版，不是有两个版本，一堆叫2，一堆叫3.然后呢？

他不是向后兼容的，就是说你以前写的脚本在，python3上可能是run的，然后，你如果用python3写，在python2上就不能执行，因为语法变了，东西没有了，哪怕是3. 在3里面的东西是不一样的，3.6.4 3.6.5.

就有区别，然后我们用这个，其实我们今天用的，我们整个课程里用到的python，都不会触及到这个问题，因为这些东西都跟新的IO有关系，另外，其实我们也已经解决了这个问题，我们已经把版本固定在3.6.5了，

我们用的解决方法就是Doctor.我不光fix了python，我们还fix来和python相关的一大堆问题。大家可以试一下，就是你们就可以自己试一下这个东西，就是比方说这里，有一个new看见了没有？你在第1个页面，这里有个new

然后你点new有个terminal，你会起来一个terminal，这个terminal是doctor里面的terminal. 你可以试一下，python的版本，然后，所以你看，这是一个很好玩的tool，里面有一个notebook. 我们还有一个terminal的access.

就是我们现在这个notebook. 有python的terminal. 我们没有其他语言，我们这个continer上其实是有很多语言的，我们可以试一试，我们上面有个c语言，给你们演示一下，看这是一个c的编译器，现在可以用c code，

然后你运行这个complier，可执行文件，C编译器通常是个很大的工作，我们这个工具做的原则是，如果有小的，就选那个小的，如果有更小的，那就选更小的，把他选到最小的，这个tcc的东西，非常非常非常小，

这个tcc的作者也是一个很有名的programer，我以后会跟你们讲，这是一个非常好的东西，你看这是我们刚才已经看到了这个，就是python，比较难的一端是c. 比较简单的是shell，然后我们这个里面其实是有c的compyler.

其实还有一些其他东西，我以后再告诉你们，另外一个比较著名的是，就是今天我给大家讲这个课，其实就是，我觉得如果，这个第1部分我大部分也还没有注意听，就略过了，无所谓，反正是讲一些知识性的东西，

不知道也无所谓，但是，这个东西你是绕不过去的，就是如果你不理解这个概念，你会遇到很多很多麻烦，如果你能理解这个概念，你会发现，哦，这个世界原来是相通的，所有东西都是遵循同样的规则的，然后，

我想讲的三个主要的概念就是，shell，Script，REPL，shell和script你们肯定知道是啥东西，powershell，REPL是首先要重点讲的，shell是一个很简单的东西，我有一个命令，我想要命令计算机干一些事情，

我就可以告诉他，他就像长一个耳朵，我们去告诉他，这是一个命令输入的地方，还有一个script. script. 其实就是一个命令，这是我有一堆命令，script就可以把它理解成我的命令，然后，Repl.是4个单词的缩写，

read，evaluate，print，loop，这4个单词是站在shell，计算机的角度，来理解的，read是计算机的read， 计算机的read的命令，然后他read了这个命令，然后他读懂了，然后他就会去执行这个命令，evaluate，

然后它执行完了以后，他就会告诉你执行的结果，就像你让一个人去干一件事情，他干完了后回来向你汇报一下，他会告诉你结果是什么，然后这三个其实就构成了一个loop，就是计算机，读你的命令，然后他执行，

他打印，然后你发现哎不对，我搞错了，然后再来，又搞错了再来，就是这样，这是一个loop，这个loop，是一个非常关键的概念，因为我们刚才讲过就是1万小时理论，20个小时理论，不管它多少小时理论，

你都可以把它细分成很多loop，不是多少个小时来决定你是否学会了，而是你run了多少个loop，如果你run的loop的时间非常非常短，你有输入，马上能看到结果，然后立即修改，比方说你这个loop，可以改到一秒钟以下，

甚至比这个还快，甚至以飞快的速度在滚动这个，而且对于一个程序员来说，最关键的一点是，我们其实是非常容易分散注意力的，我们经常就被别的事情给吸引走了，如果这个loop的时间很长，

我开始告诉他，唉，我去执行这个事情，然后他就开始去执行，可能要等6个小时，中间你干什么？不算6个小时就算半个小时，中间你都会去干别的事情，比如打开网页看看别的东西，拿出手机来刷一刷，然后你lost了，

你focus lost了，不知道自己在干什么，你的效率就会非常低，但是如果这个loop很短，会紧紧的抓着你，它把你给滚到这个过程中去，你的大脑会跟他一直都在交流，你的工作效率会非常高，那么你的20个小时，比方说，

一个小时的loop，你其实知只有20分钟，我们做一下练习，这个练习就是有点，有点不算python的内容，我做这个练习重要的原因是，python的这些概念，其实是从unix的系统里面来的，那么你想理解这些概念，

python当然很好，当然，包括我们今天用的猪皮特，也非常好，但是我们还是需要到原则上去understand这些东西是怎么来的？他的basic idea是非常非常简单。这里有一个很长的命令行，你们可以把它复制下来，然后运行一下，

注意一下，这里应该有一些空格的。第2个笑话

3

然后你复制了，然后把它粘贴到你们刚才打开的shell里面，好，这条命令是干什么的呢？正好明天去get数据，正行命令就是到网络上这个地址，去拿一个数据包下来，然后把她解包到你本地的一个folder.

这个foler叫msxpy。你会看到这个东西，看到了吗？有一大堆TXT. 然后我来解释这些TXT是什么东西，大家都get到这个数据了吗？这个数据不大，这个数据的数据量很小，大家应该很快就能get了，这是这个玩意，

我们这个课以前，微软有一个video的网页，就长这个样子，你看这有一大堆录像，然后这些录像里面有字幕，大家注意看这个字幕的吗？然后把这个字幕从录像里面抠出来，然后变成你们刚才看到的那些TXT.

然后大概就这17个，然后就有17个txt. 就是这17个video的字幕，然后我们就用这个字幕，来搞什么事情？我们可以大概看一下这个文件，进到这个目录里面，字幕文件大家大概有接触过的吧，很好理解，

就前10行19号照的这个小的，然后下一个，然后输出数据长这个样子，哪怕不了解这个格式，你一看也明白，这是序号，第四个，开始结束，然后这是一句话，有一些信息对我没用，对，我们现在这个练习没有用，

比如时间轴，我们根本不关心她在哪里出现，我们也不关心他什么时候第几位出现，他是干什么的？我来讲一讲我们的目的，就是啊，我现在要看17个video. 然后我英语有点不灵，我可能有些字我不认识，

有一个理论就是说，如果一个video里面有超过5%的单词，你不认识，你是看不懂的，就是，你必须把认识的词的量提到90% 95%. 的水平，你看一个video最好的办法就是事先知道这里面有多少词，然后你去上课的

时候就会有一个的单事先的单词列表，我们想干的事情，就是把这17个video里面的字幕文件，变成一个单词列表，然后我看看这17个video里面都出现了哪些单词？然后我们试一下，大家可以猜一下，

17个video里面大概会有多少个单词？这个video加起来大概总共有三个小时左右，大概会出现多少个英语单词？500？1000？2000个，大家算出来的平均值差不多，对，1000多个，至少，大家先试下，大家先在那个目录里面，

一会儿我讲这个clean.sh是干什么的。先把这个流程走完，看到结果了吗？要看到结果的是吧？对，就是变成了一串单词，这个脚本就是干这个的，刚才那个file的形式，你回忆一下，它里面是一句话，一句话这样的，

然后把它变成单词列表，再来一下，再接两个，这个在我们一会儿后面出现的video里面还会出现，有的东西就是你不了解，我没接触过unix我不知道这东西，但是你还知道他是干什么的，这两个单词都非常明显，

排序和去重。他的想法很简单，就是，我有10万个单词，里面某个单词出现了500遍，我要把它变成一个单词，我怎么搞？他排的是按字母来排的，我们的目的是要搞到一个单词表，出现了一次也要出现，

大家可以试一下，然后我们看到这个单词表了吗？然后后面还可以再接，word count。看看有多少个，1000多，一般会少于100. 不管这个video会有多长，最后你需要记的单词，你不会的单词，只要100个左右，

word count是记录词，字，行，只看第1个数就可以了，记录有多少个行？多少个词？有多少个字符？这个比较丑的东西，就是这个脚本，其实它不丑，哦，我教你怎么看这个东西，我们阅读代码，揭示有规律的，

就像看书看图，或者看文章，一定要知道这个结构是什么样的，你要顺着这个结构，他这个结构可能是一种什么形式？他可能是一个比方说，从右往左或者从左往右，那么你读这个脚本，它是非常直观的组成方式，

看到这些竖线了吗？这些竖线就是把各个部分给粘帖起来了，也就说我们这里有12345. 78. 把它弄成了一串，然后中间有7个，管道的符号把它们连接起来，我们可以挨个看一下，第1个很有意思，在干什么？

阶段，用空格把单词分开，那里面会有很多单词，我现在把空格都换成一个换号，所以这个行就会变成很多很多行，一行一个单词，就这意思，好第2个，对，也有很多出现在句首就是大写，把这个错误去掉就是，

把大写的都变成小写的，就是一个字，后面这些同样的dirty work. 他是把windows的换行符给去掉，然后我们开始去标点，这个标点不要，这个标点不要，所有标点都不要，然后还有这个，这个是干嘛的？

这个有点震泽这个我稍微来解释一下，这里有两个尖，这个尖这个尖，这个尖的意思是不要az和大写的a到z. 这个尖的意思是开头，就有点困扰一个符号两个意思，她和他出现位置有关，在中括号里面是不要的意思，

他在一个正则的对头，最前面就是开头，他的意思是所有不是字母开头的，所有不是字母开头的都去掉，这样那些括号就去掉了，所有里面还有数字的都去掉，它是一个单词，但是可能是一个很怪的单词，#@！%

打错了，反正也是错，也就不需要这个单词，因为英语单词没有数字，什么单词里面有数字啊？阿拉伯数字，其实是个错误，不应该叫阿拉伯数字，传到西方去的，他是印度数字，具体说是印度的一种古老的语言，

梵语，（讲故事）我写个梵文字号可能写的不对，我问一个印度人这个怎么发音？他说这个字读o数三下，因为这个文字里面说了数三下，我从没想到一个语言的发音异常都是可以被规定的，我们汉语有音调，

这个发什么？上升还是下降？它没有语调，他说汉语你都学会发的语调，英语他有重音，有很多张嘴的词，中国人也说不好，但是这个语言从来没有规定说这个语言这个音要发多长时间，下一个音要花多长时间？

就跟音乐一样，有节拍的，他的文字里面有这种东西3？很surprise. 就是你学计算机语言也会有同样的经历，我很熟悉这些c的，Shell.python，c啊我很快就可以学会，我半天就会学会一门新的语言，是一旦你遇

到一个让你非常吃惊的东西，哎，好像根本没有想到，根本没有见到的东西，这个感觉就非常非常，另外一个新的语言，就比如你们现在学的python，它不会改变你的思维方式，不客气的说，他就不值得学，

就除非她能改变你的思维方式，你才值得去学它，如果你学完还是这样想，那就没有意思了，我们继续，（故事完）。好下一个继续，jupyter来开始熟悉我们的编程环境，这是我们的地球，木星比你想象的大得多，

目前是差一点变成恒星的行星，这个是大红斑，是个风暴，从我们看到木星的时候就有这个东西了，春节的时候有个很热的电影，叫流浪地球，地球飞了，然后木星，叫jupiter，希腊神话里射箭的爱神，那个神的名字，

然后我们用的环境怎么念，叫jupyter，pyter，python读py。。。这个网站一定要记住，抽出一点时间，与大家交流的写代码的人，都在这个网站，Hacker News，如果你有什么问题，比较简单的，三秒钟告诉我答案的，

但是所有重要的问题都不是这个网上可以解决的，比如15分钟可以解决它可以分为好几个月甚至几年的时间，关于设计的问题，关于更大规模的问题，这个网站有很多很好的帖子. 当然也有很多很多人很无聊的瞎扯，

这个能找到，这个东西怎么念？jupyter的概念我来讲一讲，我们刚才已经介绍过shell了，也介绍过Repl的概念，在jupyter里面有个概念叫cell，你们看我这个屏幕，就是这种东西，一个个方块框起来的东西一个单元，

好这种东西，这是一个运行的小的loop，在里面输入，你在这里面看到结果，然后循环，然后你编辑，就在这里面，怎么编辑？知道怎么编辑这个网页吗？你把这个光标移到这面，然后按一下回车键，就可以开始编辑了，

我们给的环境，每个人编辑自己的，但我也可以搞一个大家一起编制的，每个人变异自己的，就是我讲的任何东西，你就可以试，删除，或者加上去，你改，你可以自己去编辑，还有一个，运行，运行就是可以退出编辑的状态，

按下esc退出编辑的状态，然后你按着shift. 就可以运行这个东西了，然后你运行完了以后，然后他就会把结果给你打印出来，在下面，一会我试一下，我们先写第1行代码，hello world. 大家可以试一下，

大家时候你只要按一下我们已经写好的代码，都是按着shift再按回车，然后再改一改，比如说这里这些string随便写一写，然后你会看到它运行银行出来，就像这样子，比方说，我们加一个感叹号，就这个意思，

它是可以立即运行的，然后你的focus的点，周老师两三行代码，所有关于环境所有关于依赖的库，这些7788上课之前，我们都不需要管，运行代码就行，这个东西非常popular. 你跟人家讨论什么？很多人在用猪皮特做笔记。

4

其实不是的，她完全差不多是照抄的，另一个软件的，他就是copy的，我们可以看一下，这个软件是数学，这个软件还有个替代品，其实不是，他们两个差别非常大，这个软件注意一下，阿姨说了，你首先看到的是个图，

一个三维的图，还有公式，他在输的时候，先打了一些文字，basic，每一个sub title，你注意到它右边的那些结构了吗？这是他的整个笔记，只在一个，树形结构下面，我们用的猪皮的其实是把，是把这个东西给cut掉了，

我因为很多人其实就只想写一个Markdown一样，一类标题，第2类标题，这是code，但是，基本上，他们的想法很像，他就输入一个公式，如果你这里仔细看的话，有一个输入，相当于是一个repl的r，然后一个输出，相当于repl的p，

然后这里做了一个计算，他的说说下面你就可以看到，还有一大堆东西，这个软件就是，你们有机会一定要用一下就是，有机会去买它，你们可以到他的网站去试一下，worfram，为什么这个软件很牛呢？我见过这个软件，

我用过这个软件系统，我对他的印象最深，为什么这样讲呢？你看你们都装过很多软件，你们通常装软件少有一步叫做software licence，你要agree，然后那个东西我估计没有任何人看，就是那个很长非常boring的Document，但是这个软件的licence，

非常短，而且你就不可能不看，因为它的licence也很特别，它的licence不是通常那些软件，如果你用这个软件出了什么错？造成了什么问题？跟我们无关，比方说你用这个软件出现的安全问题，死了或者什么样的，跟我们无关，

但是他的软件很屌，真的很屌，第1条，你不能用它做原子弹，就是说可以用它做原子弹，然后第2条不能开发大规模杀伤性武器，他，nasa的工程计算就是用它，他可以算，他可以算得非常非常精确，你们可以试一下，

你们可以拿一个随便的计算器，算一下，1万的阶乘这种东西，100个阶层都是很大的数据，他就不一样，它的精度很高，它会变成一个，你买了我的软件，不能转口给伊朗，叙利亚那些国家，一大堆东西，说明这个软件真的有用，真的能解决问题，

然后这个剩下部分你们可以看看，然后如果你们有兴趣的话，可以看一下，worlf里面有一整套东西。加深一下印象，这个东西我刚刚已经讲过了，这个东西的结构是什么样的？这个东西我刚才没讲，后面刻的github.昨天有同学问，

我到哪里去事先看一下这些课我可不可以预习，我可不可以先做一些功课？去这个网站就行，这个link是我们课程里面体系里面最简单的一部分， 最基础的一部分，（video）。就是，10周年的时候，乔布斯录了一个视频祝贺worlf，

他这个人是从来不会跟任何无力的事情，我另一个公司另一个产品跟他没有关系，他想把他的一个产品和，worf联合起来，我们家mathematics. 你会发现我很多很好的想法，就像写notebook. 然后你的文本，你的图像，和你的代码，

都在同一个page上，然后你们再看一个类似的格式的东西，然后这里放的是一个ppt，各种各样的展现形式，这是一个很powerful的东西，你跟很多程序员交流，程序员说我不需要这个东西，我的代码就是文档，你在某种程度上可以这样说，

你可以在代码里写一些注释，但是即使你在代码里写的注释，有一个问题你是很难解决的，就是，你的代码和你的本能其实是一维的，就是你阅读的时候，按照字母顺序，在一维空间上越多，有很多信息你就是没法在一维空间上展示，

比方说图，还是三维的图，比方说video，代码里嵌个video，比方说image.各种各样的东西，还有就是大家想一想，即使是一个符号，一个数学公式，你回忆一下你看到的数学公式，为什么你的代码总是写的比数学公式？你看一个paper里面有很多数学公式，

你要把它变成一个可以运行的东西，你的代码一定比你那个paper里面的数学公式长，有些公式很短的，它非常的简洁，他的原因不是因为数学公式另外一个符号，这些符，其实是因为他排布在二维空间了，他不是这样子，他可以到处出现了，

在上面，在下面，在哪里？所以你的眼睛只需要在一个很小的范围内，就可以识别这个东西，你也不能到处看，然后你的脑子也不用keep整个image. 就知道这个公式在讲什么。读code呢，就是，code长了以后，你没有很好的记忆，你就知道整个structure是什么样子，

尤其是大的data base，class，很多类，几千个这样的几万个的function，然后你这时候你想知道自己在哪，你就非常不容易，大部人的反应是什么？大部分的反应其实是，就是他也不知道自己在哪，但是他在这个系统里面他知道，从哪来的？要去哪？所以它的功能只需要

有些数据，然后我就到那些数据那边去了，然后我写一个结果，但是从来没干过，好多code就是这样的，因为你很难把它放到一个非常小的几何空间，一个屏幕这么大，你只能放这么点，实际是可以的，有这样的记述，有这样的语言，会让你非常focus.

你几乎不用编辑器，我们这个快结束的时候，看一个加强我们记忆的东西。他其实是在介绍unix。他讲的basic idea就是，你可以把Unix理解成非常非常多的，Body. 所谓软件的模块化，我改一部分，不需要改另一部分，改变很小，我在这些model的基础上，

这个想法其实是比较特殊的，

5

这个Video后面部分课下自己把它看完，这个人很有名的，他可能最出名的工作就是c，还有AWK，这些图现在都还在，就是比方说我用的这个，这个系统，就是Mac，Mac他其实下面就是unix,unix

都是这样子， 你会看到就是几十年以前的，的一些idea. 现在都还在，甚至连那个程序的名字都还没有改，sort还叫sort，uniq少掉两个字母。然后，他把他整个搞下来，basic idea是非常像的，同样的例子就是，Mathematics. 对这种写notebook的idea其实也是非常早出现的，这是nature program. 二三十年以后我们还在用，包括后面课要学到的machine learning，就是neural network，其实neural network是非常非常早出现的，neural network是50年代就有的事，computer hardware machine learning。然后，我们回到课，ok，今天的第1部分就是编程环境，其实我们只写了一个code，只写了python的hellow word，但是讲了很多history也好，跟这个环境相关的东西，大家有一个概念，python从哪来的？它跟unix环境有极深的联系，然后，在这个环境里面编程，比方说，这些string的概念，这些shell的概念，REPL的概念，都会反复出现，然后我们最后呢，再来熟悉一下这个编程的环境，我刚才给你们show了一下这个document. 它这个help里面，有很多document，还有一个东西，就是说，我想知道我怎样快速的操作这个，文档，比方说，这些快捷键我怎么去查？有一个快捷键的快捷键，叫这个快捷键会帮助你去查其他的快捷键，在这个东西在这个系统里面应该是，Shift加f健. 会出现一个search list. 这是一个非常快速的index. 然后你可以search比方说，我想搜所有跟cell相关的，你看这个列表其实是非常长，你其实最经常用的，是编辑记录和编辑的状态，和执行，和执行的状态编辑执行，也就是两三个常用的快捷键，我们在这节课上，其实你们最有可能输入的是这个，都是一个Python语句，但是这个notebook里面，大部分都不是python的cell，你注意看一下这个这里，这个cell是一个code，然后这个code后面跟着的是python3，然后下一个cell，下一个cell是一个markdown，下一个cell是一个markdown，如果你新建一个，比如说你按一下b，就可以在现在这个cell之前下面插入cell，这个cell现在是code，然后你按一下m它就会变成一个markdown，然后再按一下回车，就会进入到这里面，然后我就可以编辑它，把它打印出来就会像这个样子，这个，快结束来吧，这节课主要的目的就是让大家setup环境，然后上手练，知道这个loop，然后知道Python这个语言是从哪来的？有一些兴趣，对发生这样，我们还讲了很多和python不太相关的东西，主要是让大家有这个热情，继续听下去，我们是不是应该来一个提问或者回答的环节？踊跃的发言，有什么问题，有什么想法？或者是你觉得，唉，这个课讲的太简单，是不是应该搞一点有意思的东西，你讲的我有点跟不上，我们是不是可以放慢点？没有ide，不需要ide，写在网页里面，它的好处是把所有你需要的东西汇集起来，图片录像公式，文本，（题解答环节：家里电脑怎么跑）这是image（doctor），吹牛中。。。。。。有谁还有问题？大家有没有想就是下课的时候自己搞一些东西，我的想法是我们从一开始就上课试一试，然后，回家就完事了，我们应该做一些什么做来学，python的官方文档里是有教程的，。QA环节，github环节

6

github环境讲解，主要图像解释。看见来吗，这里有很多system命令，看起来有很多file，但是，不知道大家注意到没有，还有什么奇怪的地方？奇怪之处在哪？有人看出来来吗，大小都一样，不同的程序大小都一样，它根本就没有不同的程序，他就一个可执行文件，整个系统的所有的，所有的就一个file，把所有的都build一个文件里，这个文件非常小，就你越往下看就会发现，就是可以在很小的space. 就可以在非常小的里面实现特别强大的功能，就是一兆，几兆，它带给你的benefit就是，你看，我们刚才用shell，我们刚开始装这个机器的时候，有很多问题，Windows不是一个非常适合的环境，他装的时候还会重启n次，重启之后又加上update. 你要搞半个小时，但是如果你在一个unix环境里面，你可以直接doctor pull，两分钟就好了。今天上午就到这里，吃饭

7

任何时候你写东西的时候可能有错的，这是简洁对吧？然后在这基础上你可以再改进，我觉得就是我让你自己看书，看文章还是学东西，最好的一个办法是，是抄错，就是，我来这不是来，follow你的，你说一遍，我重复一遍，这就没意思了，我要挑一个错的出来，我挑一个错对你好处极大，我给你们讲个故事。。。所以要挑错，我们做的任何事情，比方说，其实我们今天讲的这个课，他可能会讲，你有兴趣，你可以自己去search一下，自己查一下，你看看其他程序员怎么吐槽py，怎么吐槽spy的，就是他有什么问题？我从来没有见过任何一个软件，就是没有人吐槽的，他总是有各种各样的问题，他总是需要改进，所以你看完了这个东西之后，你就知道有一些东西，它之所以现在是这个样子，它其实是历史进程，他是不得已这样子，一些被称feature的东西，其实是一些bug，这其实是我软件的一些feature, 其实他不是，他是想修补一个bug. 他要修补出新的bug. 然后他又有一个新的feature，其实还是一个bug，这个过程其实是永远没有尽头的，但是一旦你能够突破这个点，发现这里面有bug. 你就可以进入到下一个level，然后还会有新的bug, 新的problem. 然后如果你再能去突破，你又可以上到一个level. 因为你学习的曲线不是这样子的，你当然希望是这样子，然后他有可能是这样子，但它一定不是一个线性的东西，会有某一个积累的过程，有一个突破的点，那么，方法是什么呢？就是，要多想要多试，我们在下面的课里面会，会刻意的安排一些环节，我们会犯一些错误，大家会发现，这里有一个错，Error.bug，这是编程序，其实不是，编程序其实是你要，解决问题，要找到 要fixed 这个bug. 他这个改进很好，希望大家以后能多发现新的改进方法，可以开始下面的课来吧，人都到了吧，关于那个环境，我们课下再讨论，上完课我们再讲这个环境，高博士来做这个环境，今天上午讲的东西，给大家留一个小作业吧，就是回去多阅读，读很多link，一旦你开始读你会发现里面有很多，就是address，他是一个完全发散的东西，你也不知道书的难点在什么地方，读得越多的，你的头脑就越可能build一个map，你会把这些件联系起来，他们其实是一个非常小的circle，他们互相之间都有directly cost，现在我们开始来讲一些简单的东西，我知道这里面有一些同学已经有python的经验了，有多少人平常工作会用python？举下手我看看，好，你们几个可以出去了，开玩笑的，就是说，这个，你如果平常用python，你听我这个课其实是有点浪费时间，但是，大部分人他是不熟悉的，所以我们还是要照顾大部分人的，所以基本的东西开始讲，所以你们可以帮帮大家，我们数学也一样，还有一些很基础的甚至一些中学的数学，但是我们又没法说，我们就直接skip掉，自己课下去看，因为，即使是中学的数学，有一些你也记不住，所以我们？如果大家觉得节奏很慢，或者是内容太简单，就及时跟我反馈，先从变量和数据类型开始，咱们先算一个公式，这个东西大家都很熟悉了吧，去过健身房的应该，都测过这个东西，它其实是一个很简单的数学公式，体重处以你的身高平方，记住，我们一会儿要用这个公式，比方说这里有一个身高，这个#后面是一个comments，如果你没有接触过python的话，就是，这种深绿颜色的写在井号后面的部分，它其实不是代码，他只是在解释代码在做什么，对，注释，你们可以在自己的机器上运行一下，看一下结果，嗯，现在我们看见了一个赋值号，它其实不叫等号，一个是除，一个是平方，它这个平方的符号，很有意思，它乘，乘两次，他这个平方，然后我们来运行一下，为什么，发生了什么？刚才有一个是忽然消失的，然后出了些错，看一下这个错在说什么，你不会一下子写了一个程序，一行没有改错，直接得出结果，你肯定会出错的，它上面有建议，回去一行，我现在run会正确吗？要从头回到最开始的地方，得从这里开始，然后，现在weight也有了，现在bmi有了，ok，出来了，21.算出一个值来，我们其实做机器学习，或者是做人工智能，从数学中，纯粹数学，的抽象角度很像，跟刚才的过程很像，你看我们刚才其实是，一个人的两个feature. 身高和体重，然后我们把这两个feature变成了一个值，一个值就变成一个数，就是两个数变成了一个数，然后你就可以用这个字来表示，比如说胖瘦，这其实是一个很好的例子，但是有很多machine learning的过程，他不是只有两个feature，对吧？他可能有2亿个feature，他可能有无穷多个feature，但是，what,你可以把所有这些问题归纳为算出一个数。有一类问题叫分类。他其实也是算出一个数来。一个number。啊你们自己run一下这个刚才的，可以试一下。也可以自己算一下自己是不是胖还是瘦。这其实是你们自己的笔记。这相当于你们在我的讲义上面直接记笔记。自己写commit，自己加，其实这个notebook，他其实就是一个jason。你可以用编辑器打开。是这样子的。他就说一个jason。很大很大的jason。所以你看它的file很小。PPT没法运行，也没法在PPT里写一段空的，而且运行，PPT没法实时的改，做不了这个，而且PPT你想输入公式是不行的，那些那些大家说我看到的那些video. 都在云上面，都在我们的存储器里面，而且如果你们自己想编辑的话，不要这样子，也不用带一个很大的几百兆，他其实是一个里面有一个process. 他前段是一个web. 然后这两个直接是有message的，他会发一个message给这个python，python会run出一个结果，然后再发回给这个web，他会认的出来，所以就是你看，这个地方，看起来了吗？这个session是一个python three的session. 他会对应的一个python的解释器，就是你可以在里面混数学，可以在里面混自然语言，但是，我知道有这样的项目，我没见过成功的，在里面换不同的语言，比方说我这个cell，用一个python，下一个不是python，下一个是c加加，下一个是java，这样子不可能的，但一个page是可以的， 那你整个page是算另外一个，你可以换成另外一种语言，但是你不能mix，有人做过很多这样的实验，给github上面做了7788. 但是我想他的点是什么？是有个数学公式，我用三种语言实现它，给大家秀出来，这样子也可以，那样也可以，但其实你想，你可以用三个tab. 同时看三个file，开三个浏览器，每个tab里面都有相同的page，如果大家觉得这是一个非常好的，非常重要的，可以给大家上一个，这就是一个cell的概念，大家可以算一下，这是一个很简单的数字计算，然后谈到数字计算呢，我们就来讲一讲数据类型，我觉得如果没有编程经验的人第1次接触，有两点非常疑惑。

8

你从小到大学的数学,等号就只有一个意思等于，那么这里的意思就是赋值。

数据类型float、int、str、bool，你看的还是的类型其实跟他们在存储器中的分布有很大的关系，其实他们都有一个优点就是不用管内存。

Pyhon里面有个表达式type，可以打印出来是什么类型。

Float，直观来说浮点数，大家想过为什么要跟整数分开，浮点数想做什么事情呢，其实想做一个map。

这是一个数轴，这里面有多少个数，无穷多个，问题就来了，8个比特(bit) 构成1个字节(byte)。每个bit或者是0或者是1,所以一个8位的字节包含256种可能的0,1组合，所以需要做一些取舍。我用某些位存精度，用吗某些位存十的多少次方，所以他不能精确的表示差距。整数在浮点数，1=1.0，他在存储器里面是这样放的。

理解程序和程序语言，需要换个角度。本来是一个程序员的问题，我可以把它变成一个数学问题。

大家有没有注意到我一下跳前面去，快捷键contro+A。你怎么快速编辑，怎么在二维里导航，看看文档光标。

你在shell里面编辑，跟我们的Jupyter环境里面是一样的，在cell里编辑运行。

这是一个编辑概念。

python里面有一个jplpython。

我们讲string，这里看起来像一个值，就是他不是的。他是一个容器，一个值得集合。

以后课程看到的是向量和运算。

在数据里面我们看到的没有一个类型是向量。他没有一个独立的数据类型。

因为他被封装在一个容器里面，所以会看到很多for循环。

bool型

你知道bool概念是谁发现了吗，一个数学老师，我不知道这个老师这个名字，我知道他外甥叫名字，Geoffrey Hinton。

现在的Ai跟这个有很大的关系，就是深度网络。他们家出了很多了不起的人物。

2 + True 为什么是3，猜一下false是多少？ture是1，把取值范围0和1的数转为整数。

有一些不明显的问题，需要我们注意，+ 不是的数学里面的加号，他就是一个符号。

你可以把它理解为是一个函数名，这个函数在参数一个是整数一个是bool，所以结果是整数。

"hello" + 2，"hello" + 2，没有定义了，处理不了。

9

"hello" + "world" ，字符串可以连加。这不叫加，其实是把他们串起来。

这个符号什么意思，是根据他的上下值决定的。

这需要一个你自己不断反复训练的过程，bool型加一个整型是一个整型。

77 % 17 ， 取模函数名mod，算一个除法余数。

商是什么？这是商，他会自动运算。

字符串

这其实是一个markdown，关联了python环境。

大家想一想，这个结果是什么？他没上上上上海的车。

上\*3就是上上上

一个字符乘以一个数字，字符串有这个操作就是重复三次，加法就是把他拼起来。

Python语法不是那么整齐，需要缩进。

如果+代表往字符串增加元素我，那么-/运算是取字符串的子集？

用过C语言转过来，可能会问怎么运算的。

他这个是从0开始算的，第一个是上。

字符串，一个起点0，一个终点-1。

休息

10

休息

https://github.com/microsoft/ai-edu/tree/master/C-开发工具与环境/mpp，这个就是我们的安装的文档。

Docker、git，我们唯一的依赖就是这两个。

目前没有git，下次我们会把git的放进去。

大家以后可以checkon。

大家以后可以在github上建立账号。

他其实是一个文件的寄存，非常非常简单。

他有一个很简单的方法，就是反复用哈希函数。

Git，它的设计理念非常好。

Git命令有这么多可以试一下。

长用的命令不超过十个。

最后一句就是今天run的。

我们可以往上拉一个git。

然后在线的把它改到pullrequest

你们现在干的事，从网上拉一个。

好，大家有兴趣自己看一下。

好，我们回到上面的课。

这是一个表达式。表达式的结果是一个布尔型。判断bmi > 20

有很多种表达。逻辑运算，可以结合，可以化解。大于或小于等于有not这个概念。其实这两个是一样的。

这是我们第一次看到了真正的等号。

大家看这一题，or要是左边或者右边有一个ture，那他就是对的。

他肯定不会算错。

有了这个单元我们就可以介绍branch

分支的意思就是如果这些不行，就做那一些。

语法有缩进。这if下面的意思就是说，如果这个条件满足啦，行，这个。

Python的语法就有过缩进，这不是可有可无。所以一定要注意。

大家回去可以试一试。

其实只有两点不好，一个tab一个；要注意。

嵌套，就是在往里在往里。

我们这边缩进4个。

好，else，这是else。

11

主要是理论上可以无穷多个。

有些写过是c code语言的人就会有这个感觉，那写一个表查这个，结果不同。

这里的冒号，接着这个。

有很多好工具可以帮你code，是什么style根本不重要，关键是要一致。你不能一会儿这样，一会儿是那样的。

关于这个bmi在这个环境里面编辑。就要刚开始讲的那个公式。

有一个link我们看一下。我们去看数据。我不知道他的取样，是从哪里取的？这个东西要很多。

西方人跟东方人的身高体重数据是不一样的。老人，小孩，男人，女人都不加区分，这是不对的。

这段代码是打印的，胖的瘦的。好大家可以试一下。这个有点看的不太清楚。你们自己的那个表格应该看得很清楚。

因为他这个因为他这个table的range，会变得很局促。应该是差不多的数据源。大家可以把这种东西写全。

这算是一个程序。身高体重胖还是瘦告诉你，就算bmi，我就前两天，前两天我去健身房。我觉得用这个公式计算也差不多了。

循环是跟类型相关的。

列表也是容器。字符串是字符的容器。列表是数据的容器。

我们从通用的list开始。

这里一些什么事？区别在哪里？他不是一个字符串，它变成了一个list。

字符串变成了一个list。这些list里面放了一些数据，这里可以是数据也可以不是数据。什么都可以。你可以混合往里面塞的。像一个大杂烩一样。

如果string可以干一些事，那么list也可以干。

可以比你0小，可以是-1。我有更好的方案，把他正负依次排开。

你把它一个字符，变成一个值。我们就用这个来。

就是你把一个自然语言的一句话，变成一个list，所有的空间是一个空间。然后向量的语言也是一个空间。

不是说任何两句话组合起来就是一句有意义的话。有异议的话，在所有组合里面是非常小的概率。

你就说你总能把，这个空间映射到那个空间。把一个字符串弄成一个值。因为只有我们可以算。因为值我们可以算出来。我们很善于做算术。

这是一个很好的概率题。有时候概率小。其实很大。

这里所有人，比我们大一天，或者早一天，或者小一天。绝对大于50%。

继续。

看一下。前面是它的字符串，后面是值。

你看我们脑子怎么想的？有一个外部输入，会变成很多这个。有电磁波的，有声波的。但是他进入你的神经网络里，就是电和煤。

是我们就想一个问题，是不是所有东西都可以计算的？能不能把我们所知的一切都要计算来代替？

你能不能表示他是另一个问题？这个东西是不是通过计算可以解决的？要重新定义科学。所有东西都可以变成计算。我们要处理的数据其实都是自然语言。比如胖瘦啊。可以不停地改变。模型变得很复杂。最终都可以计算。这是一个哲学问题。它是一个科学的方法。以前做一个实验。我们要描述一个现象。假设。然后我们证明这个假设对不对？有没有新的发现？交流的方式什么？如果不是要自然语言写的。他就是可以算的，像我这样写就是可以算的。那我也很能很容易验证这个东西是不是对的。这也是以前都可以的基本方法。但是我的模型之前依赖于计算的。所以他的网站是很，古怪的。大家可以去看一下。它是有结构的。像Google 他并没有一个大的数据库。

12

可以是不同的类型。我们可以中间换一个数值。

反正我也没有运行。我也就没有改数据。

这是我们第一次，看那个叫dilit。你看他写这样个保留字。

你，就是现在去学习。嗯，从小学，中学，大学数学，好像python都放在后面。我们要先讲布尔运算，然后我们才会去讲branch。

循环需要多练练出感觉的。

然后我们的意思是把他们的type元素都打印出来。

现在程序就开始有意思了，内容就多少了起来，变得非常有意思啦。

这里我们搞了一个range，range等于1是指生成一个数1。

我们到现在为止，我们所有的数据都是我们自己敲上去的，现在开始，生成一个数组。最简单的例子就是序列。

好大家可以试一下。输入range100怎么样？

这就是一个函数。他的意思是产生一个数组到这个数为止。

然后，这是你说的这种clock啊，这个函数，他不是一个值，它是一对值。在python里面也是一个数据结构。

信息

回去试一下。啊？以为自己不会写错，其实不是这样的。然后也会debug弄半天。

13

来，先做个题，阶乘。当然这个题目还是有点问题。

阶乘就是，求一个数N乘到N-1个数。

我在这个定义里面就开始用这个函数。

他的语法跟for很像的。我在这里面就可以引用啦。他进去之后又可以引用了。然后你就可以退出来。

这是非常power的。

你可以把那些七七八八的都表示为函数。这idea被一个人给实现了 有一个著名的语言叫lic。

我们看这个了。注意到，出现一个\*号。他还得跑。我们就一个阶乘。你不小心写了一个很大的数。 或者什么一个死循环。然后你就会经历这种情况。就好像整个系统没有反应了。然后你就会发现这里有个新号。不是每个bug都给你一个明确的提示。

然后，这个时候，你就难以，debug了

他为什么会报一个错误。他说最大的maximum recursion depth exceeded in comparison

他好像陷入了一个很深的递归。达到了，达到了某种边界。跑了那么长时间。为什么？大家想一想，我们定义0了吗？所以我会一直乘下去。然后0往下乘，后面全部没有定义了。其实我们阶乘还是大于零的一个自然数。0！是1。0！不0吗？

我们刚才定义是一个递归的定义。我们把它改成循环。这里加了loop(n)。

这样你就可以一直往下乘啦。这是另外一个实现。其实你们可以改一下当时那个条件。如果输入是零或者一点时候。他在阶乘都是一，就把它改对了。

自己去手动试一下。

这是我们看到的第一个公式。组合数大家还记得吗？知到这么回事吗？求组合数里面有很多阶层乘。

我这儿有五个东西。我同学吧。从这里面选三个。有多少选择？多少？大家可以算一下。这是他的定义。你们可以算一算，计算机硬算。

这个在古典概率学里面会经常用到的。

但是很简单，你可以用阶乘来实现，其实就是函数。你有一个公式，然后把他带入就可以了。

这就把公式重新写了一遍。改成了我们的函数。

大家可以自己试一下。这是一个函数。我们定义函数的时候我们知道他是个函数，这其实是没什么意义的。我们需要知道他什么意思？

或者是二元的加。他只是说了你的类型和对应关系，没有告诉你值什么样子。这是一个特别有用的思想。

然后我们再看max。很简单，求最大值。看到了吗？我们其实我们觉得是一样的值。其实是不一样的。刚才那东西。就是我们，我没改之前。他是输入类型的。然后没改之前是这样的。给他很多参数，他能处理很多参数。他是一堆整数变成了一个值。另外一个是一个数组变成了一个值。他们是不一样的。

方法。我们讲到这里，其实会看到一些面向对象的东西。然后面向对象的基本想法。我们第一个对象。然后这个对象有固定的的方法属性。

然后这里的数据是无序的。然后我们把它，sort一下。

你会发现sort数据是有顺序的。

到哪里去查这东西呢？

我们可以进入文档。

然后又有，然后我们就会找到很多方法。其实我想找的就是他。这是一个class，然后我要得到一个。

然后，这是一个reverse。其实是sort一种方法。

我们可以试一下。然后就会带来一个新的问题。我想知道所有的类型。我们试一下。

我们看到一个参数。方向。什么是方向？就是从小到大。他缺省的从小到大。我们把他从大到小。

通过查文档。我们发现会有个什么函数通过查文档。我们发现会有个什么函数实现这个问题。参数实现这个问题。reverse比如其他什么的。

因为一旦做编程。你的很多时间都是在，search文档。他是一个很长很长的列表。然后你又看到这么多的函数。你需要查找。

我们的数据是有域的。这是一个dict的，我们来把它排序。然后他的年龄。你发现它们都活的很久的。在那个年代，他们平均的寿命是30岁。他活了七八十岁，他们。

无名函数，他在程序员里面有特殊的含义。有名字反而是累赘。

还是刚才那个例子。从大到小排。刚刚试过了。按年龄排序。改个无名函数。我们返回age。他要输入是一个dic。返回lambda。

14

它的输出是一个number. 然后这个number. 是这个key的值，这个key其实接受的是一个函数，那么每一次他这个sort.我对这个东西做sort.这个东西是没有定义的，就是凭什么说的是按照名字sort? 还是出生年月日时，它没有定义的，所以你要给我个定义，这个定义我们用key来指定，比如说这个key，我们的定义其实是一个函数，然后我们去把它写在这，当然你也可以在外面写一个sort的function. 但是你看这样一行就方便了，非常简单的一个嫂sort，然后他的输出应该是这样子的，我们来试一下，大家也可以试一下，大家把这个索尼亚斯排在最后，不知道他活了多长时间，未定，不知道多长时间，就是他还没有死，这是另外一个排法，这个排法比刚才写的简单一点，刚才这个参数名字我们随便起了一个，这个就能看明白，其实你可以不管什么就是一个，这次简单就是从小到大，谁出生最早？第1个就是孔子，第二就是你了，到印度，就是搞函数这个东西，他说就是把一个复杂的东西搞成一个black box，然后就是有输入有输出，然后我们就会搞很多很非常多的东西，打包成一个library. 来试一下，在你的terminal里面装一下这个包，你们可以new一个terminal，然后在里面装一下，这个包是现装的，它是一些算法的集合，所以你看我们现在是一个Python notebook. 所以当我们想写shell的时候，我们就只能把它写成一个highlight code. 装了以后，他会有一些输出告诉你装了，试下面这个，没反应是对的，有反应说没错，我们就试一下这个库行不行。你不能不会这个，这是干嘛？这是用别人的库，你自己又不会写，嗯，但是你可以一直用别人的，大部分情况下，几乎你能想到的，几乎你能碰到的所有的功能都有人实现，但是你可以选择，这其实也是一个艺术，以你自己独特的眼光，所以这个行，那我们可以用一下，来试一下，如果你在里面装好了这个的，你们可以执行这个，就是举一下例子，哪里去找到那些库，我想找一些库，我应该去哪里找？有多少个，有18万个，有没有大家用过的？这些是下载多的，pyTouch. 我们不会用这个库，还有很多库，这很有意思，如果你想干一件事情，第一，你再看看是不是你在干一件没有必要的事情，就是别人已经弄好了，别人有答案了，然后通常，如果你干的这个事情，你一下就找到答案了，就因为这件事情你不怎么知道干，但是如果你找的这个答案不错，根据我的经验，他往往不是好的，你很快发现她还有这样那样的问题，最后你发现还有更好的，最终你没有认清这个问题的本质，你才会有这些问题，直接去做自己应该做的事情，最后，最后我们还剩下的时间可以问问题，下周还会再来，下周的第1天还是python，然后会focus在几个重要的库上，然后有了这些基础，我们可以搞数学现在你们可以开始写code了，python的语法很详细的讲了一遍，所以你们可以开始写函数了，可以自己开始做库了，可以知道去拿查文档了，然后我们搞了一些体验什么的，搞什么题做一下，考试的时候的算法用python写一遍，看看结果是什么样的，说白了，所有的写一个函数都可以出去，然后你写出来这个函数，然后你自己调用这个函数，你测试，你把各种各样的编辑写好，让它运行一个十几套，把所有的情况都测一遍，测试都通过就算past case. 我印象中有两三个题是可以这样子做的，那些sort的题其实，大家想一想那个题可不可以不用for循环来写？可以写出给出一个on的算法，他要求是一个on. 放学。