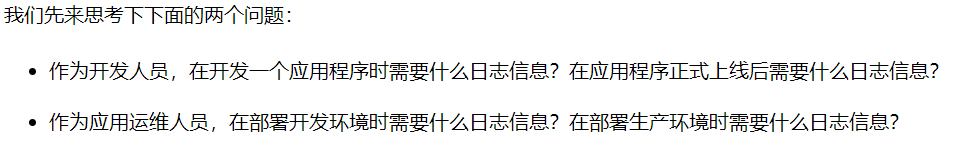
**、一、logging模块**

一、日志相关概念

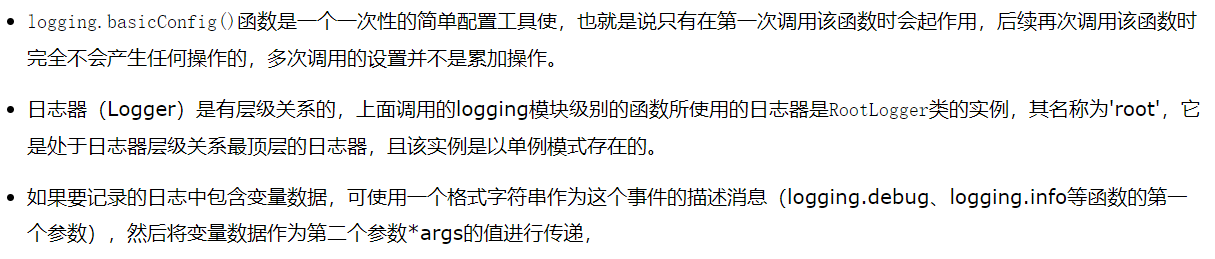
日志是一种可以追踪某些软件运行时所发生事件的方法。软件开发人员可以向他们的代码中调用日志记录相关的方法来表明发生了某些事情。一个事件可以用一个可包含可选变量数据的消息来描述。此外，事件也有重要性的概念，这个重要性也可以被称为严重性级别（level）。

二．日志等级











这次使用的是 biopython 中解析 blast 结果的功能，随着 blast 版本的不断更新， blast的输出结果的格式也在不断改变，所以这对于 biopython 解析 blast 的结果造成了很大影响，所以 biopython 中一般倾向于处理 xml 格式的 blast 输出结果，因为这种结果的格式一般不随 blast 版本的改变而改变。在进行 blast 的时候需要选择参数 -outfmt 5 即 xml 格式。在使用 biopython 时，加载包的代码是：

from Bio.Blast import NCBIXML

NCBIXML 就是 biopython 中针对解析 xml 格式 blast 结果的子包，要解析结果，同平时的读取文件一样，也需要一个 blast 结果文件的句柄，假设我们的结果文件叫’blast\_result.xml’， 那么读取就是：

result\_handle=open(‘blast\_result.xml’,’r’)

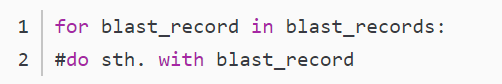
有了这个句柄，我们就可以通过 biopython 对其进行各种处理了，首先， blasts 的结果是一条一条存在这个句柄文件里的，这时就要分情况讨论，在做 blast 的时候我们是用的一条序列去 blast 还是用的多条序列去 blast，如果是一条，那么在读取这个句柄的时候就可以用：



如果是多条：



唯一的区别就是 read 和 parse 的区别，他们的原理都是一样的，即从句柄中提取结果。result\_records 对象是一个迭代器，可以允许用户一个接一个的获得 blast 的结果（因为有多条序列）。一般采用 for 循环来挨个读取比对结果：



现在，所有有关与 blast 结果的信息都被储存在一条条的 blast\_record 中，那么如何提取这些信息？

blast\_record 的结果可以分为三大类，调用不同的类可以得到相应的信息：descriptions、 alignments、 multiple\_alignment。其中 descriptions 和 alignments 都是以 list 的形式来存储数据的。

·descriptions 下面有四个信息： title、 score、 e、 num\_alignments，分别对应数据库中匹配上的序列的标题、匹配的得分、匹配的 e-value、以及多序列比对的数目。

·aligments 下面有三个信息： title、 length、 hsps，分别对应数据库中匹配上的序列的标题、匹配的长度、其中 hsps 是 list 格式的对象，里面储存了 query 和数据库中序列匹配的具体信息，包括匹配得分、 gap 等信息，详细的内容在后面的图片里。

**XML文件序列的对信息，如果是调用aligments类，则从图中可以看出，我们可以获取到的关于序列标题Hit\_id,Hit\_len,以及Hsp中的相关内容。**

**对于blast的xml文件，前面主要介绍比对的方法版本以及比对的主要信息，在Hit中，主要是序列的信息。**



·multiple\_alignment 是多序列比对的结果，里面只储存了 alignment 的信息。有了这些具体的方法，在处理 blast 的结果时就不需要写额外的脚本了，只需要提取相关的记录然后进行具体操作就可以了。

同样， biopython 的另一个子包 SearchIO 也可以完成一些类似的工作，而且对于比较简单的需求， SearchIO 的代码相对更加简洁。调用代码：



和 NCBIXML 不同的是，SearchIO 在获取 blast 结果文件的时候使用的代码是：



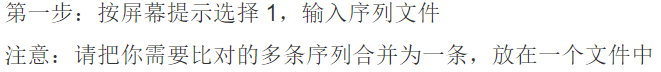
这个方法不需要先用 open 获得句柄，然后再对句柄进行处理， SearchIO 得到的对象直接就是可以进行各种操作的可迭代对象，这里， read 和 parse 的使用和 NCBIXML 中类似。有了这个对象，就可以提取对象中的各个信息，比如想知道具体是那一条序列和 blast 数据库中的序列比对上了，就可以使用 blast\_result.id， id 就指的是具体序列的名字。如果我们想知道 blast 结果中那些序列比对上了，并且把比对上的序列名称存进一个列表进行后续操作，那么代码就是：

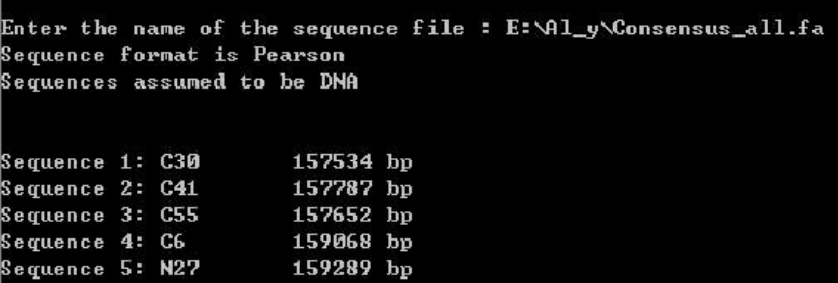


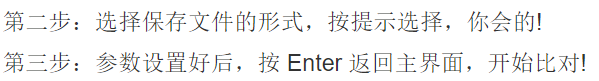
Clustalw

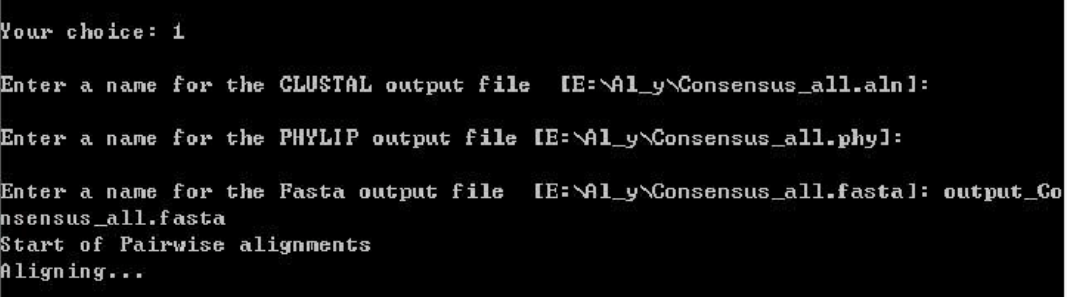
CLUSTALW是一种渐进的多序列比对方法，先将多个序列两两比对构建距离矩阵，反应序列之间两两关系；然后根据距离矩阵计算产生系统进化指导树，对关系密切的序列进行加权；然后从最紧密的两条序列开始，逐步引入临近的序列并不断重新构建比对，直到所有序列都被加入为止。

Clustalw 操作方法







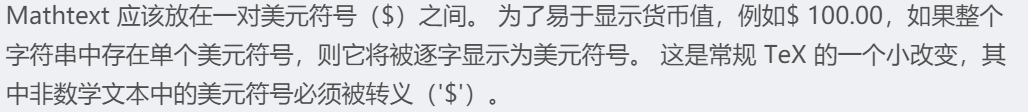


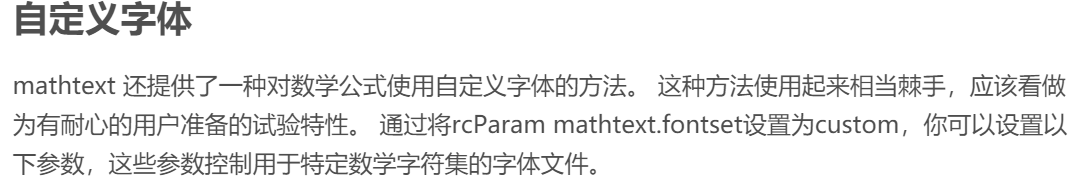
\_\_init\_\_方法使用

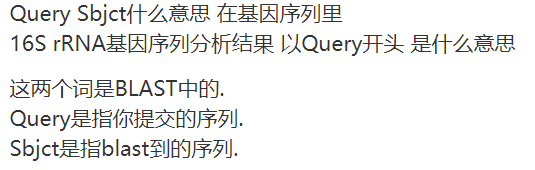
初始化 注意，这个名称的开始和结尾都是双下划线

即\_\_init\_\_作用是初始化已实例化后的对象。

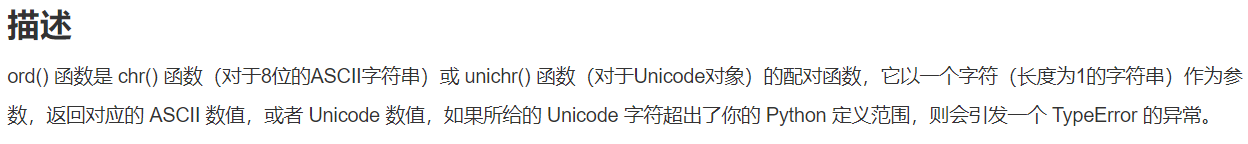
Matploitlib中的mathtext模块

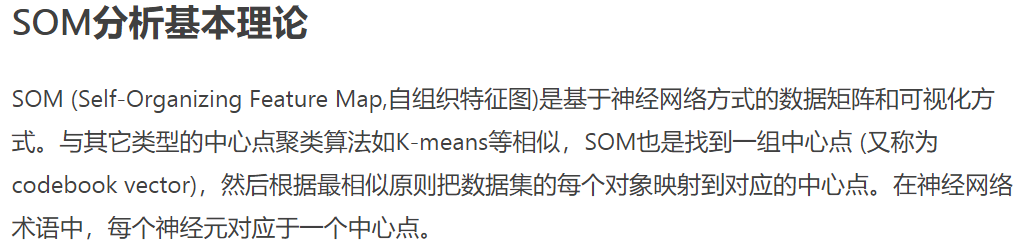


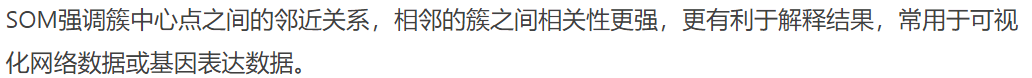






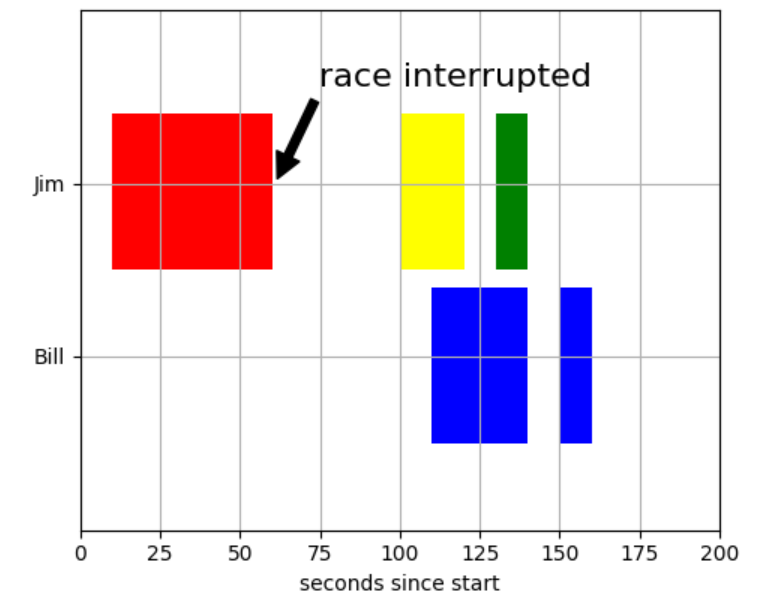












坐标值的意思为：【】内表示从【横坐标的起始位置,宽度】，（纵坐标其实位置，高度），facecolors=间断条形图颜色。

Matplotlib中的annotate（注解）的用法

