R的初级操作

> 这一部分的课程参考了Datacamp的R introdution

R语言是现在数据科学中非常流行的一种语言。虽然我个人更倾向于使用python，但是说到生物信息学的各种应用，R还是一个绕不过的东西。所以就算是看在bioconductor的份上，也应该至少学会一下R语言的基础。

如何练习R

下载安装R以后，打开R这个软件就可以进入R的命令行模式。R studio是R的一个开发环境，你可以把R studio当做界面更友好的一种R。我们接下来主要会使用R studio作为编程环境。

在R语言里（以及很多其他的语言里）行首的#表示注释。计算机不会运行注释的内容，但是这些内容对于阅读代码的人来说却很有帮助不管现在你多么确信自己明白一个代码在做什么，几个月以后你肯定不会这么想，所以给代码写注释是一个不错的习惯。至于注释应该写些什么，你可以首先把自己准备解决一个问题的步骤写成几行注释（每一行前面肯定都要有#），然后这些注释就像是文章各节的标题，提示下面的代码在做什么工作。

因为注释**不会被运行**。所以当你想让一行代码不被运行的时候，可以在这一行前面加上#。当然，如果你不小心在一行应该运行的代码前加上了#，这一行代码就没法运行了。

这个教程使用的是markdown语言书写的，具体而言是r markdown。用这种形式一方面可以利用markdown的特性，另一方面，如果用rstudio打开rmd文件的话，代码部分是可以直接运行的。下面是一个代码段，尝试点击代码段右上角的三角符号运行代码

```
# Calculate 3 + 4
3 + 4

# Calculate 6 + 12
6+12
```

代码和命令行

如果你打开 R studio 就可以看到软件界面基本上被切分成四个部分。左上角是类似文本编辑器的一个区域，这个区域可以写代码，也可以用表格的形式展示数据；右上角是当前项目中的各种变量和对象；右下角可以查看目前安装的各种包（library），也可以查看帮助文档；而左下角这个区域和R软件就很相似了，是命令行区域。

要说命令行与文本编辑的区别的话，大概就是word写信函与微信聊天的区别吧。命令行会更灵活，当你想和R做一些比较简单的沟通的时候会很方便，但是如果你想做一下需要斟酌表述的沟通的话，那恐怕在文本编辑器里面慢慢写代码会更好。

把R当做一个计算器来用

下面说到的这些就是最典型的“简单的沟通”了。在R语言环境里，做一些数学运算是非常方便的：

运算	语言符号

运算	语言符号
加法	+
减法	-
乘法	*
除法	/
幂运算	^
取余	%%

最常见的幂运算应该就是平方了，比如说3的平方是9，用代码求就是3^2。而取余的意思就是用%%左边的数字除以右边的数字，求余数。试着运行下面的代码：

```
# An addition
5 + 5

# A subtraction
5 - 5

# A multiplication
3 * 5

 # A division
(5 + 5) / 2

# Exponentiation
2 ^ 5

# Modulo
28 %% 6
```

变量声明和赋值

变量（variable）是编程的一个基本的概念。变量就是可以变的一个东西（object），因为可以变化，所以你可以为一个变量指定一个值（赋值），然后接下来通过变量的名字重新获得（get）这个值。在R里面变量的赋值的符号是<-，当然了，其他语言更通用的=也是ok的。运行一下代码：

```
# Assign the value 42 to x
x <- 42

# Print out the value of the variable x
x
```

变量的名称可以很长，一般来说变量的第一个字符使用小写字母，变量名称里面不要写空格（可以考虑用_代替），变量名最好有一定的规律，阅读代码的时候能够根据变量的名字猜出来变量表示的是什么。

下面的代码没什么特殊，就是变量名长一点：

```
# Assign the value 5 to the variable my_apples
my_apples = 5

# Print out the value of the variable my_apples
my_apples
```

刚才我们用R做计算器的时候3+4这个命令可以返回7的结果。现在能不能用两个变量相加呢？

```
my_apple = 5
my_oranges = 3
fruit_count = my_apples + my_oranges
```

变量的类型（基本类型）

像下面这些都是属于基本类型：

1. 比如像4.5这样的小数，这种成为numerics
2. 像5这样的整数，称为integer （注意，如果直接写x=5，x仍旧是numeric类型，只有明确声明x=5L才能使用integer类型）
3. 真/伪这样的二分逻辑值，称为logical。逻辑值只有两种：TRUE和FALSE。注意全是**大写**
4. 字符串内容，称为characters。字符串的两端**必**须有英文的引号"或者'

下面展示了哪三种类型的变量？

```
my_number <- 42

my_character <- "universe"

my_logical <- FALSE
```

class方法

所以如果现在你想看一个变量是什么类型，那可以用class()

```
# Declare variables of different types
my_numeric <- 42
my_character <- "universe"
my_logical <- FALSE

# Check class of my_numeric
class(my_numeric)

# Check class of my_character
```

```
class(my_character)

# Check class of my_logical
class(my_logical)
```

class和typeof好像都可以返回对象的类型。但是在某些地方还是会有微妙的区别。整体来说，class会是你比较想要用的那个。

> 下面演示程序的来源

```
x = 1:10
> x = 1:10
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> y = x/5+rnorm(10)
> y
 [1] -0.1971789  1.1195313  0.4743572 -0.2644470  0.3854243  2.0359322  2.1123241
0.5846501
 [9]  1.9206651  1.9500710
> class(y)
[1] "numeric"
> typeof(y)
[1] "double"
> g = lm(y~x)
> class(g)
[1] "lm"
> typeof(g)
[1] "list"
```

可以看到typeof倾向于返回primitive类型（double和list），而class则倾向于返回更有意义的**类型**，比如lm(linear model)和list这一对结果就是明显的例子。

向量 Vector

到现在为止，我们一个变量就只能存储一个值，对吧？

```
a = 10
print(a)
a = 12
print(a)
```

可以看到先输出了10，但是再次将a设置为12以后，a现在的值是12. 如果现在我想同时存储10和12呢？比如说现在我想声明一个变量scores，存储5个学生的小测得分10，12，11，15，10那应该怎么办呢？在R里面，这种可以存储多个同类型的值的变量，叫做向量

> 在python里面，我们把这个叫做list……

在R里面，建立一个vector的方法叫做`c()`，比如：

```
numeric_vector <- c(1, 2, 3)
character_vector <- c("a", "b", "c")
```

收支表

比如说你给自己的收支流水做了记账，现在要统计一周的收支情况。数据参见data.xlsx的01记账工作表

在excel中如果要统计10-20到10-24的所有开销，可以使用`sum(B2:B6)`公式，那么你有没有想过`B2:B6`是个什么东西呢？

在R里面，当你要对一组数据进行统计的时候，首先要用向量（vector）将这些数据存储起来

```
cash_flow = c(100, -80, -20, -12, 10, -23, 50)
```

向量的命名

以刚才的数据为例，如果直接输出的话

```
> cash_flow
[1] 100 -80 -20 -12  10 -23  50
```

这就不是很容易读，如果能将这个收支记录标记**上**星期几就会更好一点，这个操作叫做naming

```
> wday = c('sun','mon','tue','wed','thu','fri','sat')
> names(cash_flow)=wday
> cash_flow
sun mon tue wed thu fri sat
100 -80 -20 -12  10 -23  50
```

向量的运算

向量的加减是逐个元素进行的，所以有

```
> a = c(1,2,3)
> b = c(4,5,6)
> a+b
[1] 5 7 9
> a-b
[1] -3 -3 -3
```

那么a * b是什么呢？其实这里向量的乘既不是点积（dot product）也不是叉积（cross product），只是单纯的做element wise的乘法运算而已

```
> head(mtcars,2)
              mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4      21   6  160 110  3.9 2.620 16.46  0  1    4    4
Mazda RX4 Wag  21   6  160 110  3.9 2.875 17.02  0  1    4    4
> tail(mtcars,2)
               mpg cyl disp  hp drat   wt qsec vs am gear carb
Maserati Bora 15.0   8  301 335 3.54 3.57 14.6  0  1    5    8
Volvo 142E    21.4   4  121 109 4.11 2.78 18.6  1  1    4    2
```