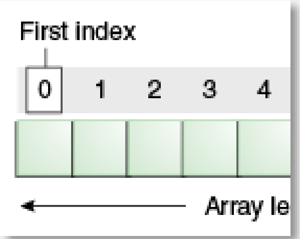
R vs Python

NEED-TO-KNOW DISTINCTIONS

★

Indexation From 1 (R) vs 0 (Python)

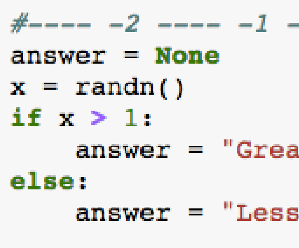


Unlike in *most* programming languages, indexation in R starts from [1]. Python, on the other hand, starts indexation at [0]. You really have to shift your mentality when you jump between the two tools.

Tabulation Matters In Python

★★

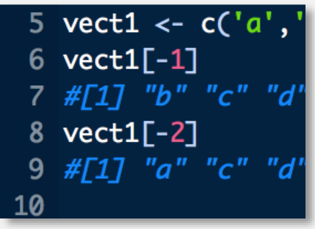
Python is created to be visually appealing even in terms of reading the code. This is one of the *few* programming languages where white spaces matter. Watch out where you place indentations! This is not the case in R.



```
#---- -2 ---- -1 -
answer = None
x = randn()
if x > 1:
    answer = "Grea
else:
    answer = "Less
```

★★★

Negative Indexation [-1]



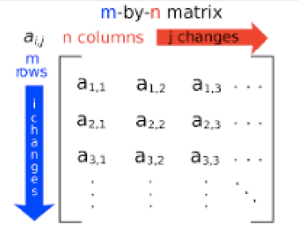
```
vect1 <- c('a','
vect1[-1]
#[1] "b" "c" "d"
vect1[-2]
#[1] "a" "c" "d"
```

In both languages you can pass a negative index [-n] to a vector / list. However, the results will be drastically different. E.g. in R [-4] excludes the 4th element from the subset, in Python [-4] returns only the 4th element from the right.

Matrix Population Behaviour

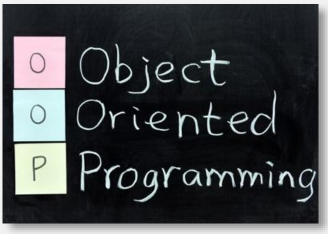
★★★★

By default, in R the matrix() function populates matrices column-by-column, whereas in Python np.reshape() does it row-by-row. This behaviour stems from the original design of the languages and can be adjusted via special parameters.


m-by-n matrix
$a_{i,j}$
n columns
j changes
m rows
i changes
$a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,1}$ $a_{3,2}$ $a_{3,3}$

★★★★★

Vectorized vs OOP



The fundamental underlying principles of the two languages are different: for R it's vectorization and for Python it's OOP. Very important to keep this distinction in mind when using either tool for Data Science. (More details in the course).

>> Click Here To Join Our Python And R Programming Courses <<