분석프로그래밍 I.02

Operations, Variables, Data types, Vectors

국민대학교 경영학부 빅데이터경영통계학과

2018. 3. 12(월)

1 데이터 타입 (Data types)

R의 기본적인 데이터 타임은 다음과 같이 구분할 수 있다. 데이터 타입 `numeric`, `character`, `factor`, `logical`, `Date`, `POSIXct` 등은 `class` 함수를 사용하여 확인할 수 있다(예. `class(TRUE)`).

데이터 타입	예
숫자(numeric)	`0, 1, -1, 0.45, 3L, 1e-7, 0xFF, pi, exp(1)`
문자(character)	`"Letter", '1', '"Hello", says he', "Hello?",`
	`'Cheer up!\r\nRight Now!', '\x41', '\101'`
범주(factor)	`factor(c('high','low','high'))`
논리(logical)	`TRUE, T, FALSE, F`
날짜(Date, POSIXct)	`as.Date("2018-03-12"), as.POSIXct('2018-03-12 10:33')`
	`as.Date("2018/03/12"), as.POSIXct('2018/03/12 10:33')`

앞쪽 테이블 '예'에서 설명이 필요한 부분은 다음과 같다.

데이터 타입	예	설명
숫자(`numeric`)	`3L`	`L`은 Long integer의 의미인 듯하다
	`2.3e-3`	$2.3 \times 10^{-3} = 0.0023$
	`0xFF`	16진수 `FF`
	`0xff`	16진수 `FF`
	`pi`	원주율 π = `3.1415926535897931`
	`exp(1)`	자연 상수 e = `2.7182818284590451` e는 Euler's constant(오일러의 상수)라는 의미에서 왔다.
문자(`character`)	`'\r'`	Carriage return을 나타내는 제어문자
	`'\n'`	new line을 나타내는 제어문자
	`'\x41'`	아스키 코드 16진수 `41`에 해당하는 문자(`A`)
	`'\101'`	아스키 코드 8진수 `101`에 해당하는 문자(`A`)
논리(`logical`)	`TRUE`	'참'을 나타내는 키워드
	`T`	`TRUE`를 나타내는 약어
	`FALSE`	'거짓'을 나타내는 키워드
	`F`	`FALSE`를 나타내는 약어

R에서 제어문자는 '\'로 시작한다. 따라서 문자 '\'을 나타내고 싶다면 `'\\'`로 써야 한다. 다음은 몇몇 제어 문자의 의미를 나타낸다.

제어문자	의미
`'\a'`	alert beep(비프 알람 소리)
`'\b'`	backspace(한 문자 앞으로)
`'\t'`	tab(탭)
`'\n'`	newline(다음 줄로)
`'\r'`	carriage return(동일한 줄의 첫 위치로)
`'\f'`	form feed(다음 페이지의 첫 위치로)

`print('abc\bde')`를 해보자. 결과는 `"abc\bde"`가 된다. 왠지 시시하다. 만약 제어문자가 그 소임을 다한 결과를 보고 싶다면 `cat` 함수를 사용한다. `cat('abc\bde')`을 해보자. 결과는 `abde`가 된다. `\b` 대신 **다른 제어문자**를 넣어 보고 그 결과를 비교해보자.

2 연산(Operations)과 함수(Functions)

2.1 숫자(numeric)

```
3 + 2
5 - 4
3 * 2
72 / 2
3 ^ 4
3 ** 4
3 ^ (1/2)
sqrt(3)
3 - 2 + 2 * 4 / 2 ^ (1 + 1)
# PEMDAS(Parenthesize, Exponentiate, Multiply, Divide, Add, Subtract)
7 / 3   # Float division
7 %/% 3 # Integer division
7 %% 3  # Remainder
```

Listing 1: 산술 연산(Arithmetic operations)

```
exp(1)
log(180, base=2)
log2(180)
log10(180)
sin(2)
```

Listing 2: 함수(functions)

```
"+"(3,2)
"*"(3,2)
"%p%" = function(x,y) { 2^x + y^2 }
3 %p% 2
"%p%"(3,2)
```

Listing 3: 연산은 함수이다.

```
7 < 3
7 > 3
```

```
7 == 3
7 != 3
sqrt(2)^2 == 2
print(sqrt(2)^2)
print(sqrt(2)^2, digits = 21)
all.equal(sqrt(2)^2, 2)
all.equal(1e-23, 1e-24)
```

Listing 4: 비교 연산(Comparison Operations)

2.2 문자(character)

```
print("Letter")
print('Letter')
print('"Hello", says he')
cat('"Hello", says he')
print('Cheer up!\r\nRight Now!')
cat('Cheer up!\r\nRight Now!')
#https://stat.ethz.ch/R-manual/R-devel/library/base/html/Quotes.html
nchar('hello?') # number of characters
paste('Here is', 'an apple.')
substring('hello?', 2, 3) # from 2nd character to 3rd character
```

2.3 날짜/시간(Date/POSIXct)

```
Sys.Date()  # Current Date
Sys.Date() - as.Date("2018-01-01")
as.POSIXct("2018/12/31 23:59:59")
Sys.time()  # Current Date/Time(POSIXct)
as.POSIXct("2018/12/31 23:59:59")-Sys.time()
as.numeric(as.POSIXct("2018/12/31 23:59:59"))-as.numeric(Sys.time()) #
    in Seconds
(as.numeric(as.POSIXct("2018/12/31 23:59:59"))-as.numeric(Sys.time()))/
    60 # in Minutes
```

2.4 논리(logical)

```r
#https://stat.ethz.ch/R-manual/R-devel/library/base/html/Logic.html
(7 < 3) & (4 > 3)
(7 < 3) | (4 > 3)
!(7 < 3)
xor(T, T) # XOR
isTRUE(x == 3)  # robust to NAs
```

R에서 논리값은 TRUE, FALSE, T, F으로 나타낼 수 있다. 이때 TRUE는 항상 '참'을 나타낸다. R에서 TRUE는 '참'을 나타내는 예약어keyword로 다른 의미가 부여될 수 없다. 반면 T와 F는 R에서 TRUE와 FALSE로 미리 값을 부여하였지만 변경될 수 있다. 다음의 코드를 통해 확인해 보자.

```r
> TRUE <- c(3,2)
Error in TRUE <- c(3, 2) : invalid (do_set) left-hand side to
    assignment
> T <- c(3,2)
> T
[1] 3 2
```

3 변수(Variables)

- R은 동적 타입의 변수를 사용한다.
- 숫자, 문자, 함수, 분석 결과 등 모든 R 객체(Object)를 담을 수 있다.

3.1 변수의 이름

```r
myAge = 22
year = 2018
day_of_Month = 3  # Alphabets, numbers, '.', '_'
stock.high = 13322
whatIGotForMy23thBirthday = "flowers"
```

3.2 변수 할당

```r
# Assign
a <- 3
b = 2
-1 -> d
e1 <- e2 <- 7

assign("var1", 3)
varname = "myVariable" # Camel case
assign(varname, 2)

# Naming Convention
myVariable = 1 # Camel case/Pascal case
varname2 = "my_Another_Variable" # Python
varname3 = "myVariable.3rd" # R
assign(varname2, 2)
assign(varname3, 2)
```

3.3 변수 관리

```r
# list variables
ls()
ls.str()

# Does a variable named "a" exist?
exists("a")

# remove all variables
rm(list=ls())

# remove all variables except for functions
rm(list = setdiff(ls(), lsf.str()))

# save my current workspace
save.image()
```

3.4 변수의 데이터 타입 확인하기

데이터 타입	확인하는 함수	문자에서 변환하는 함수
숫자(numeric)	is.numeric()	as.numeric()
정수(integer)	is.integer()	as.integer()
문자(character)	is.character()	
범주(factor)	is.factor()	as.factor()
순위범주(ordered)	is.ordered()	as.ordered()
논리(logical)	is.logical()	as.logical()
날짜(Date)	class(□) == "Date"	as.Date()
날짜시간벡터(POSIXct)	class(□)[1] == "POSIXct" "POSIXct" %in% class(□)	as.POSIXct()
날짜시간리스트(POSIXlt)	class(□)[1] == "POSIXlt" "POSIXlt" %in% class(□)	as.POSIXlt()

```
x = 23L
x = 22.3
x = "strings"
x = factor(c("Hi","Lo","Hi","Lo","Lo"))
x = as.Date("2018-01-01")
class(x)
str(x)
#typeof(x)
#storage.mode(x)
#mode(x)
```

4 벡터(Vectors)

4.1 벡터 만들기

```
# concatenate
c(1,3,2,4); c(pi, 1e-3, sin(3)); c("John","Mary")

```

```r
# with names
person = c(name = "John", likes = "Mary", loves = "Suzy")

# repetition
rep(1, 10)
rep("ABC", 10)
rep(c("A","B","C"), 3)

# sequence
1:10; seq(1,10)
seq(from=1, to=10, by=2)
seq(from=1, to=10, length.out=3)
seq(form=1, to=10, by=2, each=3)
```

4.2 벡터 내용 간단하게 확인하기

```r
vec <- sample(100,10000,replace=T)
vec <- sample(c("John","Mary","Tom"), 100, replace=T)
print(vec)
head(vec)
tail(vec)
summary(vec)
psych::describe(vec)
table(vec)

hist(vec)
hist(vec, breaks=seq(-0.5, 100.5, 1))
#cut(vec,breaks=seq(-0.5, 100.5, 2))
#table(cut(vec,breaks=seq(-0.5, 100.5, 2)))
```

4.3 벡터 내용 간단하게 확인하기

```r
vec <- sample(100,10000,replace=T)
vec <- sample(c("John","Mary","Tom"), 100, replace=T)
print(vec)
head(vec)
```

```
tail(vec)
summary(vec)
psych::describe(vec)
table(vec)

hist(vec)
hist(vec, breaks=seq(-0.5, 100.5, 1))
#cut(vec,breaks=seq(-0.5, 100.5, 2))
#table(cut(vec,breaks=seq(-0.5, 100.5, 2)))
```

4.4 벡터 연산

```
x = 1:20
x * 2
x + 5
x / 3
x^3
sqrt(x)

v1 = c(1,3,5)
v2 = c(2,4,1)
v1 + v2
v1 / v2

v1 = c(1,4,2,4)
v2 = c(1,2)
v1 / v2

length(v1)

c(1,2,3,c(4,5),c(6,7,8))

v1 < v2
all(v1 < v2)
any(v1 < v2)

```

```r
v1 = c("I", "You", "He")
v2 = c("go", "come", "climbs")
nchar(v1)
paste(v1, v2)
substring(v2, 2, 2)

vec <- sample(100,10000,replace=T)
mean(vec)
sum(vec)
sd(vec)
```

4.5 집합 연산

```r
primes <- c(2,3,5,7,11,13,17,19,7,13)
numbersILike <- c(3,7,14,3)

union(primes, numbersILike)
intersect(primes, numbersILike)
setdiff(numbersILike, primes)
numbersILike2 <- c(3, 7, 14)
setequal(numbersILike, numbersILike2)

is.element(17, primes)
is.element(18, primes)
17 %in% primes
18 %in% primes

numbersJaneLikes <- c(5, 7, 11)
setdiff(numbersJaneLikes, primes)
length(setdiff(numbersJaneLikes, primes)) # is A a subset of B?
```

Listing 5: 집합연산:`setequal, union, intersect, setdiff, %in%`

4.6 벡터 참조

- 순서로 참조

- 논리형 벡터
- 이름으로 참조
- 참조 : http://adv-r.had.co.nz/Subsetting.html

```r
# subsetting by orders
vec = 1:10
vec[c(2,3,5,7)]
index = c(2,3,5,7)
vec[index]

vec[-1]
vec[-1:-2]

vec[index][-1]

# subsetting by names
person = c(name = "John", likes = "Mary", loves = "Suzy")
person["name"]
person[c("name","loves")]

# subsetting by logical values
vec = sample(5, 10, replace=T)
biggerThan3 = vec > 3
vec[biggerThan3]
vec[vec > 3]
vec[c(T,F)] # recycling
```

4.7 데이터 정렬

```r
vec = c(1,5,6,4,2)
sort(vec)
order(vec)
vec[order(vec)]
```

- 벡터 vec와 sort(vec)

index	1	2	3	4	5
vec	1	5	6	4	2

−(sort)→

index					
vec	1	2	4	5	6

```
vec = c("Tom","Alex","John")
sort(vec)
order(vec)
vec[order(vec)]
```