

R의 데이터 형식

환경생태데이터사이언스 실습 September 10, 2019

R 패키지 설치 및 부르기

기본 사용법: `install.packages("package name")`

```
install.packages("raster")  
install.packages(c("raster", "randomForest"),  
                  repos="cran.seoul.go.kr")
```

더 많은 옵션들 확인하기

```
?install.packages()
```

R 패키지 부르기

기본 사용법: `library("package name")`

```
library("raster")
```

```
library(randomForest)
```

```
library(raster, randomForest)
```

더 많은 사용법 확인하기

```
?library
```

변수형

변수

변수명은 알파벳, 숫자, 언더스코어, 마침표로 구성

첫 글자는 알파벳 또는 마침표으로 시작

올바른 예:

```
Data_1, Data.1, D1_test
```

잘못된 예:

```
1_Data, 1.Data, .1D_test, Data-test
```

변숫값 할당

변수값을 할당 할 때는 <- 사용

```
A <- 3
```

```
B <- 5
```

```
C <- A - B
```

```
# Calculate mean values while assigning x inside the function.
```

```
mean( x <- c(1, 2, 4, 6, 8) )
```

```
## [1] 4.2
```

```
# Check x
```

```
x
```

```
## [1] 1 2 4 6 8
```

기본 변수형

1. 숫자: 3, 5.5, 1/6
2. 문자열: "choi", "kwanghun"
3. 불대수: TRUE, FALSE
4. 팩터 (요인): 범주형 데이터 (명목, 순서)
5. NA : 결측치
6. NULL: 미정값 (변수 초기화 때 사용)

숫자 다루기

```
# Calculate birth year
ThisYear <- 2019
MyAge    <- 36
BirthYear<- ThisYear - MyAge
print(BirthYear)
```

```
## [1] 1983
```

```
# Convert fractional number to decimal number
a <- 5
b <- 7
print(a/b)
```

```
## [1] 0.7142857
```

문자열 다루기

```
FamilyName <- "Choi"  
GivenName <- "Kwanghun"  
FullName <- paste(FamilyName, GivenName, sep = " ")  
print(FullName)
```

```
## [1] "Choi Kwanghun"
```

```
FamilyName + GivenName # Error: not possible
```

```
## Error in FamilyName + GivenName: non-numeric argument to binary op
```

불대수 다루기

```
FirstNumber <- 1; SecondNumber <- 2; ThirdNumber <- 1  
FirstNumber == SecondNumber # Do they have same values?
```

```
## [1] FALSE
```

```
FirstNumber == ThirdNumber # Do they have same values?
```

```
## [1] TRUE
```

```
# logical "And" operator
```

```
(FirstNumber == SecondNumber) & (FirstNumber == ThirdNumber)
```

```
## [1] FALSE
```

```
# logical "Or" operator
```

```
(FirstNumber == SecondNumber) | (FirstNumber == ThirdNumber)
```

```
## [1] TRUE
```

팩터 다루기

```
TestFactor_1 <- factor(c("KR", "JP", "KR", "DE", "KR"))
```

```
TestFactor_1
```

```
## [1] KR JP KR DE KR
```

```
## Levels: DE JP KR
```

```
TestFactor_2 <- factor(c("KR", "JP", "KR", "DE", "KR"),  
                        levels = c("KR", "DE", "JP"))
```

```
TestFactor_2
```

```
## [1] KR JP KR DE KR
```

```
## Levels: KR DE JP
```

팩터 다루기

```
TestFactor_3 <- factor(c("KR", "JP", "KR", "DE", "KR"),  
                       levels = c("KR", "DE", "JP"), ordered=T)
```

```
TestFactor_3
```

```
## [1] KR JP KR DE KR  
## Levels: KR < DE < JP
```

```
levels(TestFactor_3)
```

```
## [1] "KR" "DE" "JP"
```

```
nlevels(TestFactor_3)
```

```
## [1] 3
```

```
is.factor(TestFactor_3)
```

```
## [1] TRUE
```

변수형은 is를 이용하여 확인할 수 있다.

```
is.numeric() # Check whether the variable is numeric or not  
is.character() # Check whether the variable is character or not  
is.factor() # Check whether the variable is factor or not  
is.na() # Check whether the variable is na or not  
is.logical() # Check whether the variable is logical or not
```

변수간 변환

`as.character()`, `as.factor()`, `as.numeric()`과 같은 함수를 이용하여 변수간 속성을 변환할 수 있다 (`as`).

```
TestFactor    <- factor(c("4", "5", "7"))  
TestFactor
```

```
## [1] 4 5 7  
## Levels: 4 5 7
```

```
TestCharacter <- as.character(TestFactor)  
TestCharacter
```

```
## [1] "4" "5" "7"
```

변수간 변환

```
TestFactor    <- factor(c("4", "5", "7"))
TestNumeric   <- as.numeric(TestFactor)
TestNumeric
```

```
## [1] 1 2 3
```

```
# What's the problem?
```

```
TestCharacter  <- c("4", "", "7")
TestNumeric_2  <- as.numeric(TestCharacter)
TestNumeric_2
```

```
## [1]  4 NA  7
```

팩터는 기본적으로 레벨 뒤에 숫자를 가지고 있기 때문에 팩터를 직접 숫자로 변환해서는 안된다.

대안: factor -> character -> numeric

자료형

1. 벡터
2. 스칼라
3. 리스트
4. 행렬 (Matrix, 2-D array)
5. 배열 (N-D array)
6. 데이터 프레임

스칼라와 벡터

R의 기본 자료형 (데이터 타입): 벡터 단일 값을 갖는 변수들: 스칼라

벡터의 생성: `c()`

벡터 이름 생성: `names()`

```
TestVector <- c(1, 2, 3)
```

```
TestVector
```

```
## [1] 1 2 3
```

```
names(TestVector) <- c("First", "Second", "Third")
```

```
TestVector
```

```
##   First Second  Third
```

```
##      1      2      3
```

벡터 다루기

```
# seq is the function to generate sequence
```

```
TestSerial <- c(seq(from = 1, to = 50, by=2))
```

```
TestSerial
```

```
## [1] 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35
```

```
## [19] 37 39 41 43 45 47 49
```

```
TestSerial[10]
```

```
## [1] 19
```

```
TestSerial[-10]
```

```
## [1] 1 3 5 7 9 11 13 15 17 21 23 25 27 29 31 33 35 37
```

```
## [19] 39 41 43 45 47 49
```

벡터 다루기

```
TestSerial[1:10]
```

```
## [1] 1 3 5 7 9 11 13 15 17 19
```

rep is the function to generate vector with repeated elements

```
TestRep <- c(rep(c(1, 2, 3), times = 2))
```

```
TestRep
```

```
## [1] 1 2 3 1 2 3
```

```
TestRep[-10]
```

```
## [1] 1 2 3 1 2 3
```

```
TestRep_2 <- c(rep(c(1, 2, 3), each = 2))
```

```
TestRep_2
```

```
## [1] 1 1 2 2 3 3
```

리스트는 벡터, 스칼라 등 속성이 다른 여러 데이터 타입이 묶여진 데이터 형식이다.

```
TestList <- list(TestRep, TestFactor_1, TestNumeric)
```

```
TestList
```

```
## [[1]]
```

```
## [1] 1 2 3 1 2 3
```

```
##
```

```
## [[2]]
```

```
## [1] KR JP KR DE KR
```

```
## Levels: DE JP KR
```

```
##
```

```
## [[3]]
```

```
## [1] 1 2 3
```

```
TestVector <- c(1:15)
TestMatrix <- matrix(TestVector, nrow=3, ncol=5)
TestMatrix
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    4    7   10   13
## [2,]    2    5    8   11   14
## [3,]    3    6    9   12   15
```

```
TestMatrix_2 <- matrix(TestVector, nrow=3, ncol=5, byrow=TRUE)
TestMatrix_2
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    2    3    4    5
## [2,]    6    7    8    9   10
## [3,]   11   12   13   14   15
```

행렬 다루기

```
TestMatrix[1,1]
```

```
## [1] 1
```

```
TestMatrix[1,]
```

```
## [1] 1 4 7 10 13
```

```
TestMatrix[,1]
```

```
## [1] 1 2 3
```

```
TestMatrix[-2,]
```

```
##      [,1] [,2] [,3] [,4] [,5]
```

```
## [1,]    1    4    7   10   13
```

```
## [2,]    3    6    9   12   15
```



```
# transpose of the matrix
```

```
t(TestMatrix)
```

```
##      [,1] [,2] [,3]
```

```
## [1,]    1    2    3
```

```
## [2,]    4    5    6
```

```
## [3,]    7    8    9
```

```
## [4,]   10   11   12
```

```
## [5,]   13   14   15
```

```
# Solve the equation A %% x = B, solve(A, B) = x
```

```
A <- matrix(c(1,2,3,4), ncol=2)
```

```
solve(A)
```

```
##      [,1] [,2]
```

```
## [1,]   -2  1.5
```

```
## [2,]    1 -0.5
```

```
A%%solve(A)
```

```
##      [,1] [,2]  
## [1,]    1    0  
## [2,]    0    1
```

```
# Check dimension of the matrix  
dim(A)
```

```
## [1] 2 2
```

배열은 행렬을 일반화한 것으로 다양한 차원을 가질 수 있다.

```
array(1:12, dim=c(3, 4))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

```
array(1:12, dim=c(2, 2, 3))
```

```
## , , 1
##
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
##
## , , 2
```

1. 데이터 프레임은 엑셀 스프레드 시트처럼 정리된 표이다.
2. 각각의 열은 서로 다른 속성을 가질 수 있으나
3. 하나의 주어진 열은 단일 속성을 갖는다.
4. 데이터처리의 핵심 자료형

데이터 프레임 다루기

```
TestDataFrame <- data.frame("Number"= as.character(c(1:10)),  
                             "Name" = paste("Name", c(1:10), sep=""),  
                             "Height" = rnorm(10, mean = 160, sd=10),  
                             "Weight" = rnorm(10, mean = 50, sd=5))
```

TestDataFrame

##	Number	Name	Height	Weight
## 1	1	Name1	176.9422	52.22017
## 2	2	Name2	155.0110	58.03192
## 3	3	Name3	153.7730	55.75408
## 4	4	Name4	152.4784	44.47850
## 5	5	Name5	165.1860	47.87653
## 6	6	Name6	150.9090	53.54681
## 7	7	Name7	157.3996	55.58462
## 8	8	Name8	150.6842	44.88467
## 9	9	Name9	161.2959	54.07089
## 10	10	Name10	137.8712	40.68055

데이터 프레임 다루기

```
# Calculate BMI index
# BMI = Weight / (Height)^2
TestDataFrame$BMI <-
  TestDataFrame$Weight / (TestDataFrame$Height*0.01)^2
TestDataFrame
```

##	Number	Name	Height	Weight	BMI
## 1	1	Name1	176.9422	52.22017	16.67921
## 2	2	Name2	155.0110	58.03192	24.15138
## 3	3	Name3	153.7730	55.75408	23.57853
## 4	4	Name4	152.4784	44.47850	19.13081
## 5	5	Name5	165.1860	47.87653	17.54592
## 6	6	Name6	150.9090	53.54681	23.51275
## 7	7	Name7	157.3996	55.58462	22.43611
## 8	8	Name8	150.6842	44.88467	19.76800
## 9	9	Name9	161.2959	54.07089	20.78341
## 10	10	Name10	137.8712	40.68055	21.40129

데이터 프레임 다루기

```
# Check structure of the data frame
```

```
str(TestDataFrame)
```

```
## 'data.frame':    10 obs. of  5 variables:
```

```
## $ Number: Factor w/ 10 levels "1","10","2","3",...: 1 3 4 5 6 7 8
```

```
## $ Name : Factor w/ 10 levels "Name1","Name10",...: 1 3 4 5 6 7 8
```

```
## $ Height: num  177 155 154 152 165 ...
```

```
## $ Weight: num  52.2 58 55.8 44.5 47.9 ...
```

```
## $ BMI : num  16.7 24.2 23.6 19.1 17.5 ...
```

```
colnames(TestDataFrame)
```

```
## [1] "Number" "Name" "Height" "Weight" "BMI"
```

```
TestDataFrame[2,3]
```

```
## [1] 155.011
```

데이터 프레임 다루기

```
TestDataFrame[TestDataFrame$BMI > 25, ]
```

```
## [1] Number Name    Height Weight BMI  
## <0 rows> (or 0-length row.names)
```

```
TestDataFrame[as.character(TestDataFrame$Name) %in%  
               c("Name1", "Name6"), ]
```

```
##   Number  Name    Height    Weight      BMI  
## 1      1 Name1 176.9422 52.22017 16.67921  
## 6      6 Name6 150.9090 53.54681 23.51275
```


데이터 프레임 다루기

```
head(TestDataFrame, 3)
```

```
##      Number  Name    Height   Weight      BMI
## 1         1 Name1  176.9422  52.22017  16.67921
## 2         2 Name2  155.0110  58.03192  24.15138
## 3         3 Name3  153.7730  55.75408  23.57853
```

```
tail(TestDataFrame, 3)
```

```
##      Number  Name    Height   Weight      BMI
## 8         8 Name8  150.6842  44.88467  19.76800
## 9         9 Name9  161.2959  54.07089  20.78341
## 10        10 Name10 137.8712  40.68055  21.40129
```

```
# View(TestDataFrame)
```

데이터 프레임 다루기

```
summary(TestDataFrame)
```

```
##           Number      Name      Height      Weight
##  1           :1   Name1   :1   Min.      :137.9   Min.      :40.68
## 10          :1   Name10  :1   1st Qu.:151.3   1st Qu.:45.63
##  2           :1   Name2   :1   Median   :154.4   Median   :52.88
##  3           :1   Name3   :1   Mean      :156.2   Mean      :50.71
##  4           :1   Name4   :1   3rd Qu.:160.3   3rd Qu.:55.21
##  5           :1   Name5   :1   Max.      :176.9   Max.      :58.03
## (Other):4   (Other):4
##           BMI
## Min.      :16.68
## 1st Qu.:19.29
## Median   :21.09
## Mean      :20.90
## 3rd Qu.:23.24
## Max.      :24.15
```

is.자료형을 통해 판별

```
class(TestDataFrame)  
str(TestDataFrame)  
is.matrix()  
is.array()  
is.data.frame()
```

as.자료형을 통해 변환

```
as.matrix()
```

```
as.array()
```

```
as.data.frame()
```