01. R 기초

1. R 설치 및 기본 사용법
    - 설치 생략
    - R 기본 사용법
        - R의 데이터 타입
            - 변수의 타입은 class 함수를 이용해 확인 가능 EX) class(”abc”) → “character”

데이터 타입	의미	비고
character	문자형 타입	따옴표('), 쌍따옴표(")로 표시
numeric(숫자형) double(실수) integer(정수) complex(복소수)	숫자형 타입	double : 숫자로만 표현 가능 Inf : 무한대, -Inf : 음의 무한대
logical	논리형 타입	참 OR 거짓
NaN NA NULL	NaN : 숫자가 아님을 반환 NA, NULL : 결측값	NA : 공간을 차지하는 결측값 NULL : 존재하지 않는값

2. R 기본 문법
    - 연산자
        - 대입연산자 : 변수에 값을 할당하기 위해 사용하는 연산자

대입 연산자	내용
←, <←, =	오른쪽 값을 왼쪽에 대입
→, →>	왼쪽 값을 오른쪽에 대

```
> string1 <- 'abc'
> "data" -> string2
> number1 <<- 15
> Inf ->> number2
> logical = NA
```

        - 비교연산자 : 할당된 값과 변수를 비교하거나 임의의 숫자, 문자 혹은 논리값을 비교 가능, NA는 어떤 것과 비교를 하더라도 NA를 반환

비교 연산자	내용
==	두 값이 같은지 비교
<, >	초과, 미만을 비교
!=	두 값이 다른지를 비교
<=, ⇒	이상, 이하를 비교
is.character	문자형인지 아닌지를 비교
is.numeric	숫자형인지 아닌지를 비교
is.logical	논리형인지 아닌지를 비교
is.na	NA인지 아닌지를 비교
is.null	NULL인지 아닌지를 비교

```
> string1 == 'abc'
[1] TRUE
> string1 != 'abcd'
[1] TRUE
> string2 > 'DATA'
[1] FALSE
> number1 <= 15
[1] TRUE
> is.na(logical)
[1] TRUE
> is.null(NULL)
[1] TRUE
```

- 산술 연산자 : 두 숫자형 타입의 계산을 위한 연산자

산술 연산자	내용
+	두 숫자의 덧셈
-	두 숫자의 뺄셈
*	두 숫자의 곱셈
/	두 숫자의 나눗셈
%/%	두 숫자의 나눗셈의 몫
%%	두 숫자의 나눗셈의 나머지
^, **	거듭제곱
exp()	자연상수의 거듭제곱

- 기타 연산자 :논리값을 계산하기 위해 연산자

기타 연산자	내용
!	부정 연산자
&	AND 연산자
|	OR 연산자

```
> !TRUE
[1] FALSE
> TRUE&TRUE
[1] TRUE
> TRUE&FALSE
[1] FALSE
>!(TRUE&FALSE)
[1] TRUE
> TRUE | FALSE
[1] TRUE
```

- R 데이터 구조

1. 벡터
    - 타입이 같은 여러 데이터를 하나의 행으로 저장하는 1차원 데이터 구조, '연결한다'라는 의미의 'concatenate'의 c를 써서 데이터를 묶을 수 있음

```
> v4 <- c( 3 , TRUE , FALSE)
> v4
[1] 3 1 0
> v5 <- c( 'a' , 1 , TRUE )
> v5
[1] "a" "1" "TRUE"
```

2. 행렬
    - 2차원 구조를 가진 벡터. 행렬에 저장된 모든 데이터는 같은 타입이어야 함
    - matrix 함수 사용시 nrow → 행의 수, ncol → 열의 수

```
> m1 <- matrix( c( 1 : 6 ) , nrow = 2 )
> m1
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> m2 <- matrix( c( 1 : 6 ) , ncol = 2 )
> m2
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

    - byrow 옵션 T 지정시 값들이 열이 아닌 행으로 저장

```
> m3 <- matrix( c( 1 : 6 ) , nrow = 2 , byrow = T )
> m3
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

- dim 함수 사용시 행의 개수와 열의 개수를 지정하여 행렬로 변환 가능

```
> v1 <- c( 1 : 6 )
> v1
[1] 1 2 3 4 5 6
> dim(v1) <- c( 2 , 3 )
> v1
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

3. 배열

- 3차원 이상의 구조를 갖는 벡터, 행렬에 저장된 모든 데이터는 같은 타입이어야 함
- array 함수 사용시 dim 옵션 명시 필요, 그렇지 않으면 1차원 벡터 생성

```
> a1 <- array( c( 1 : 12 ) , dim = c( 2 , 3 , 2 ) )
> a1
, , 1
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
, , 2
     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12
```

- dim 함수 사용시

```
> a2 <- c( 1 : 12 )
 [1] 1 2 3 4 5 6 7 8 9 10 11 12
> dim(a2) <- c( 2 , 3 , 2 )
```

4. 리스트

- 데이터 타입, 데이터 구조에 상관없이 사용자가 원하는 모든 것을 저장할 수 있는 자료구조
- list()를 사용한 예시

```
> L <- list()
> L[[1]] <- 5
> L[[2]] <- c( 1 : 6 )
> L[[3]] <- matrix( c( 1 : 6 ) , nrow=2 )
> L[[4]] <- array( c( 1 : 12 ) , dim=c( 2 , 3 , 2 ) )
> L
[[1]]
[1] 5

[[2]]
[1] 1 2 3 4 5 6

[[3]]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

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

5. 데이터프레임
    - 데이터 분석을 위한 2차원 구조를 갖는 관계형 데이터 구조
    - 행렬과 같은 모양을 갖지만 여러 개의 벡터로 구성되어 있기 때문에 각 열은 서로 다른 타입의 데이터를 가질 수 있음

```
> v1 <- c( 1 , 2 , 3 )
> v2 <- c( 'a' , 'b' , 'c' )
> df1 <- data.frame( v1 , v2 )
```

```
> df1
  v1 v2
1  1  a
2  2  b
3  3  c
```

배열의 시작 인덱스 값

파이썬, 자바, C 등 많은 언어는 배열의 시작 인덱스 값을 0으로 갖지만, R의 벡터는 시작 인덱스 값을 1로 갖는다.

- R 내장 함수

1. 기본 함수

함수	내용
help() 또는 ?	함수들의 도움말을 볼 수 있음
paste()	문자열을 이어 붙임
seq()	시작값, 끝갑스, 간격으로 수열을 생성
rep()	주어진 데이터를 일정 횟수만큼 반복
rm()	대입 연산자에 의해 생성된 변수를 삭제
ls()	현재 생성된 변수들의 리스트를 보여줌
print()	값을 콘솔창에 출력

```
> help(paste)  # 명령어 입력 후 우측 하단 도움말에 결과가 출력된다.
> ?paste
> paste( 'This is' , 'a pen' )
[1] "This is a pen"
> seq( 1 , 10 , by=2 )
[1] 1 3 5 7 9
> rep( 1 , 5 )
[1] 1 1 1 1 1
> a <- 1
> a
[1] 1
> rm( a )
> a
ERROR: object 'a' not found
> ls()
[1] "a1" "a2" "m1" "m2" "v1" "v2" "df1"
> print(10)
[1] 10
```

2. 통계 함수

함수	내용
sum	입력된 값의 합
mean	입력된 값의 평균
median	입력된 값의 중앙값
var	입력된 값의 표본분산
sd	입력된 값의 표본표준편차
max	입력된 값의 최댓값
min	입력된 값의 최솟값

```
> v1 <- c( 1 : 9 )
> sum( v1 )
[1] 45
> mean( v1 )
[1] 5
> median( v1 )
[1] 5
> var( v1 )
[1] 7.5
> sd( v1 )
[1] 2.738613
> max( v1 )
[1] 9
> min( v1 )
[1] 1
> range( v1 )
[1] 1 9
> summary( v1 )
Min.  1st Qu.  Median  Mean  3rd Qu.  Max.
   1       3       5       5       7     9
# 첨도와 왜도 값 계산 함수를 사용하려면 별도의 패키지가 필요하다.
> install.packages("fBasics")
```

```
> library( fBasics )
> skewness(v1)
[1] 0
attr(,"method")
[1] "moment"
> kurtosis(v1)
[1] -1.601481
attr(,"method")
[1] "excess"
```

함수	내용
range	입력된 값의 최댓값과 최솟값
summary	입력된 값의 요약값
skewness	입력된 값의 왜도
kurtosis	입력된 값의 첨도

- R 데이터 핸들링
-

1. 데이터 이름 변경 : 2차원 이상의 데이터 구조는 colnames와 rownames 함수를 사용하여 행과 열의 이름을 알 수 있으며, 이름을 지정할 수 있음

```
> m1 <- matrix( c( 1 : 6 ) , nrow = 2 )
> colnames( m1 ) <- c( 'c1' , 'c2' , 'c3' )
> rownames( m1 ) <- c( 'r1' , 'r2' )
> m1
   c1  c2  c3
r1  1   3   5
r2  2   4   6
> colnames( m1 )
[1] "c1" "c2" "c3"
> rownames( m1 )
[1] "r1" "r2"
> df1 <- data.frame( x = c( 1 , 2 , 3 ) , y = c ( 4 , 5 , 6 ) )
> colnames( df1 ) <- c( 'c1' , 'c2' )
> rownames( df1 ) <- c( 'r1' , 'r2' , 'r3' )
> df1
   c1  c2
r1  1   4
r2  2   5
r3  3   6
> colnames( df1 )
[1] "c1" "c2"
> rownames( df1 )
[1] "r1" "r2" "r3"
```

2. 데이터 추출 : R이 보유한 여러 데이터 구조 모두 인섹싱을 지원, 대괄호 기호([,])를 사용하여 원하는 위치의 데이터 추출 가능 , 행과 열의 이름으로도 가능

```
> v1 <- c( 3 , 6 , 9 , 12 )
> v1[ 2 ]
[1] 6
> m1 <- matrix( c( 1 : 6 ) , nrow = 3 )
> m1[ 2 , 2 ]
[1] 5
> colnames( m1 ) <- c( 'c1' , 'c2' )
> m1[ , 'c1' ]
[1] 1 2 3
> rownames( m1 ) <- c( 'r1' , 'r2' , 'r3' )
> m1[ 'r3' , 'c2' ]
[1] 6
```

데이터프레임에서는 $ OR []기호 → 열의 데이터 추출

```
> v1 <- c( 1 : 6 )
> v2 <- c( 7 : 12 )
> df1 <- data.frame( v1 , v2 )
> df1$v1
[1] 1 2 3 4 5 6
> df1$v2[3]
[1] 9
```

3. 데이터 결합
    - rbind 함수 : 행으로 결합

```
> v1 <- c( 1 , 2 , 3 )
> v2 <- c( 4 , 5 , 6 )
> rbind( v1 , v2 )
   [,1] [,2] [,3]
v1   1    2    3
v2   4    5    6
```

    - cbind 함수 : 열로 결합

```
> cbind( v1 , v2 )
     v1 v2
[1,]  1  4
[2,]  2  5
[3,]  3  6
```

- 제어문
    1. 반복문 : 특정 부분의 코드가 반복적으로 수행
        - for 문

```
> for ( i in 1:3 ){
+   print( i )
+ }
# 콘솔박스에서 명령어가 완성되지 않은 채 엔터가 입력되면 자동으로 +가 입력된다. 따라서 +는 입력하지 않는다.

[1] 1
[1] 2
[1] 3

> data <- c( "a" , "b" , "c" )
> for ( i in data ){
    print( i )
  }
[1] "a"
[1] "b"
[1] "c"
```

- while 문

```
> i <- 0
> while( i < 5 ){
    print( i )
    i <- i + 1
  }
[1] 0
[1] 1
[1] 2
[1] 3
[1] 4
```

2. 조건문 : 참과 거짓에 따라 특정 코드가 수행될지 혹은 수행되지 않을지를 결정

```
> number <- 5
> if ( number < 5 ){
    print( 'number는 5보다 작다.' )
  } else if ( number > 5 ){
    print( 'number는 5보다 크다.' )
  }else{
    print( 'number는 5와 같다.' )
  }
[1] "number는 5와 같다."

> number <- 3
> if ( number < 5 ){
    print( 'number는 5보다 작다.' )
  } else if ( number > 5 ){
    print( 'number는 5보다 크다.' )
  }else{
    print( 'number는 5와 같다.' )
  }
[1] "number는 5보다 작다."

> number <- 7
> if ( number < 5 ){
    print( 'number는 5보다 작다.' )
  } else if ( number > 5 ){
    print( 'number는 5보다 크다.' )
  }else{
    print( 'number는 5와 같다.' )
  }
[1] "number는 5보다 크다."
```

3. 사용자 정의 함수 : 사용자가 임의로 하나의 함수로 명명하여 저장하였다가 필요한 경우 함수를 호출

```
> comparedTo5 <- function( number ){
>  if ( number < 5 ){
    print( 'number는 5보다 작다.' )
   } else if ( number > 5 ){
    print( 'number는 5보다 크다.' )
   }else{
    print( 'number는 5와 같다.' )
   }
  }
> comparedTo5( 10 )
[1] "number는 5보다 크다."
> comparedTo5( 3 )
[1] "number는 5보다 작다."
> comparedTo5( 5 )
[1] "number는 5와 같다."
```

4. 주석 : 실행되지 않는 문장, R코드를 설명하거나 함수를 설명할 목적으로 작성하는 글로서 #을 사용하여 표시

```
# 1+1을 계산하는 방법
> 1 + 1
[1] 2
```

- 통계분석에 자주 사용되는 R 함수

1. 숫자 연산

함수	내용
sqrt	주어진 수의 제곱근
abs	주어진 수의 절대값
exp	자연상수 e의 제곱수
log	밑이 자연상수인 로그 값
log10	밑이 10인 로그 값
pi	원주율을 의미하는 pi 값인 3.141592
round	주어진 수의 반올림 값
ceiling	주어진 수를 올림
floor	주어진 수를 내림

2. 문자 연산

함수	내용
tolower	주어진 문자열을 소문자로 바꿈
toupper	주어진 문자열을 대문자로 바꿈
nchar	주어진 문자열의 길이를 구함
substr	문자열의 일부분을 추출
strsplit	문자열을 구분자로 나눔
grepl	문자열에 주어진 문자가 있는지 확인
gsub	문자열의 일부분을 다른 문자로 대체

3. 벡터 연산

함수	내용
length	주어진 벡터의 길이를 구함
paste	주어진 벡터를 구분자를 기준으로 결합

함수	내용
cov	두 수치 벡터의 공분산
cor	두 수치 벡터의 상관계수
table	데이터의 개수
order	벡터의 순서

4. 행렬 연산

함수	내용
t	전치행렬
diag	대각행렬
%*%	두 행렬의 곱

5. 데이터 탐색

함수	내용
head	데이터의 앞 일부분
tail	데이터의 뒤 일부분
quantile	수치 벡터의 4분위

```
# 벡터 생성
> x <- c( 1 : 12 )
# 기본값은 6이지만 원하는 개수만큼 데이터를 탐색한다.
> head( x , 5 )
[1]  1  2  3  4  5
> tail( x , 5 )
[1]  8  9  10  11  12
> quantile( x )
   0%   25%   50%   75%  100%
 1.00  3.75  6.50  9.25 12.00
```

6. 데이터 전처리

함수	내용
subset	데이터에서 조건식에 맞는 데이터 추출
merge	두 데이터를 특정 공통된 열을 기준으로 병합
apply	데이터에 열(또는 행)별로 주어진 함수를 적용

7. 정규분포 (기본값은 표준 정규 분포로 mean=0, sd=1)

함수	내용
dnorm	정규 분포의 주어진 값에서 함수 값
rnorm	정규 분포에서 주어진 개수만큼 표본 추출
pnorm	정규 분포에서 주어진 값보다 작을 확률 값
qnorm	정규 분포에서 주어진 넓이 값에서 갖는 x값

8. 표본추출

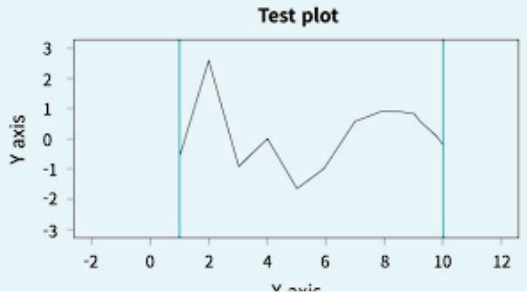
함수	내용
runif	균일 분포에서 주어진 개수만큼 표본을 추출
sample	주어진 데이터에서 주어진 개수만큼 표본을 추출

9. 날짜

함수	내용
Sys.Date	연, 월,일을 출력
Sys.time	연, 월, 일, 시간을 출력
as.Date	주어진 데이터를 날짜 형식으로 변환
format	원하는 날짜 형식으로 변경
as.POSIXct	타임스탬프를 날짜 및 시간으로 변

10. 산점도

```
# 데이터 생성
> x <- c( 1 : 10 )
> y <- rnorm( 10 )
# 파라미터 type에서 p는 점, l은 직선, b는 점과 직선, n은 아무것도 표시하지 않음을 의미
# xlim로 x축의 범위, ylim로 y축의 범위를 설정할 수 있다.
# xlab, ylab으로 각 축의 이름을 지정할 수 있다.
# main으로 산점도의 이름을 정할 수 있다.
> plot( x , y , type = 'l' , xlim = c( -2 , 12 ) , ylim = c( -3 , 3 ) , xlab = 'X axis' ,
ylab = 'Y axis' , main = 'Test plot' )
# abline의 v는 수직선, h는 수평선을 그리는 매개변수다.
# col 매개변수로 색상을 선택할 수 있다.
> abline( v = c( 1 , 10 ) , col = 'blue' )
```



11. 파일 읽기 쓰기

함수	내용
read.csv	csv 파일 불러옴
write.csv	csv 파일로 저장
saveRDS	분석 모델 및 R 파일 저장

함수	내용
readRDS	분석 모델 및 R 파일 불러옴

12. 기타

함수	내용
install.packages	패키지 설치
library	설치된 패키지 호출
getwd	작업 디렉터리 확인
setwd	작업 디렉터리 설정