**## 기본 단축기**

Ctrl + Enter : 실행하기

rm(list=ls()) : 데이터 확인 공간 초기화

Ctrl + L : 콘솔창 초기화 하기

Ctrl + Shift + H : setwd 지정하기

Ctrl + Shift + C : 주석 처리 (#은 주석)

Ctrl + 1 or 2 : 스크립트 콘솔창 변경

**### 간편함수**

**Ls()**

**: 생성된 변수의 목록을 보여줌.**

**- 목록만을 보여주며 변수의 유형 및 값은 확인할 수 없음.**

**ls.str()**

**: 생성된 변수의 목록 뿐만 아니라 각 변수의 유형 및 값 등을 확인할 수 있음**

**str()**

**: 생성된 변수 내의 값을 확인할 수 있음.**

**rm(list=ls())**

**: 오브젝트 내의 모든 변수 삭제**

**rm()**

**: 개별 변수 삭제**

**rm(x or y or z)**

**dim(mat)**

**: 행 열 확인하기**

**#### 벡터(Vector)란? ####**

하나 이상의 원소로 구성된 자료구조로서, R의 자료 객체 중에서 가장 기본이 되는 자료 객체임.

* 하나의 벡터 원소는 한 가지 형태(mode)만이 가능

**#seq (시작값, 마지막값, 증가분 \* length : 길이지정)**

증가분은 by로 표시하는데 default값이라 써도 안써도 둘 다 OK

**#rep (반복할 값, 반복횟수 \* each : 각각 \* length : 길이만큼)**

반복횟수는 **times**로 표시하는데 이 또한 default 값이라 써도 안써도 둘 다 OK

* seq(), rep() 둘 다 length 기능이 있다. (잘 까먹어서 강조)

## HELP 기능

?seq, ?rep 와 같이 함수앞에 ?를 작성하면 Help 기능이 관련 설명을 제공한다.

# 요인(범주형 자료)자료

#factor(obj, levels=변수값, labels=요인이름)

* gender, blood 변수 예시 참고

#순서형 자료

#ordered(obj, levels=변수값, labels=요인이름)

# class(x)  : 벡터의 클래스를 확인

<http://ds.sumeun.org/?p=1193> (범주형 자료와 순서형 자료의 개념)

# 벡터 접근 (개념 잊어버리지말기)

vec1 <- c(1,2,3,4,6) #1~5까지 자료를 갖는 vec1 변수 생성

vec1[5] #두 번째 자료

vec1[c(2,3,5)] #vec1의 2, 3, 5번째 값만 표현

vec1[c(-2,-3)] #vec1의 2, 3번째 자료 값 삭제

#### 행렬(Matrix) ####

▶ 행렬(matrix)은 동일한 형으로 구성된 2차원의 데이터 구조이다

▶ 행의 차원과 열의 차원을 갖는 행렬은 수학에서의 행렬과는 달리 문자형이나 논리형 등을 원소로 사용할 수 있다.

# ▶ 그러나 행렬의 원소는 한가지 형의 자료만 허락됨

# 행렬을 생성하기 위해서는 matrix() 함수를 사용

# 이외에 cbind(), rbind(), dim() 등을 이용하여 행렬을 생성시킬 수 있음

Matrix 결과값은 [,1] [,2] [,3] 와 같이 나옴

기본형식 : matrix(c(1,5,6,7,3,4,5,6,3), ncol=3, byrow=T)

# 벡터에 차원을 결정해줘서 생성 가능

dim()

m1 <- 1:9

dim(m1) <- c(3,3)

m1

**# 행렬과 관련된 여러 함수와 성분의 추출과 삭제 등에 관해 알아봄**

mat <- matrix(c(1,2,3,4,5,6,7,8,9), ncol=3, byrow=T)

mat[1,]

mat[,3]

**## row와 col에 이름 생성하기**

rownames(mat) <- c("a1", "a2", "a3")

mat

colnames(mat) <- c("z1", "z2", "z3")

mat

**# 행 열 확인하기**

dim(mat)

**#### 배열(Array) ####**

▶ 배열(Array)은 행렬을 2차원 이상으로 확장시킨 객체로써 2차원의 구조를 갖는 행렬도 배열의 한 종류이다.

▶ 그러나 일반적으로 3차원 이상의 차원을 갖는 데이터 객체를 배열이라고 부름

▶ 배열을 생성하기 위해서는 array 함수를 사용한다.

**배열을 생성하기 위한 함수 :**

1) array()

2) dim()

**## 중요한 개념 예제들**

array(1:6) **#1~6의 자료로 1차원 배열 생성**

array(1:6, c(2,3)) **#1~6의 자료로 2차원 배열 생성**

array(1:8, c(2,2,2)) **#1~8의 자료로 3차원 배열 생성**

arr <- c(1:24) **#1~24의 자료 생성**

dim(arr) <- c(3,4,2) **#dim() 함수를 이용해 3행 4열의 행렬 2개 생성**

arr

**데이터 구조 형태 한 번에 이해하기**

**Vector 벡터(1차원) < Matrix 행렬(2차원) < Arrary 배열(2,3차원) < 데이터프레임, 리스트**

**#### week 데이터 프레임(data.frame) ####**

▶ **데이터 프레임은 행렬과 비슷한 형태로 되어 있으나,**

▶ **행렬은 차원으로 표시되며 같은 형태(mode)의 객체를 가지는 반면,**

▶ **데이터 프레임은 각 열(column)들이 서로 다른 형태(mode)의 객체를 가질 수 있으므로**

▶ **범주형 변수를 가질 수도 있기 때문에 범주형 자료분석에도 유용하게 사용된다.**

**# data.frame() :** 이미 생성되어 있는 벡터들을 결합하여 데이터 프레임을 생성

char1 <- rep(LETTERS[1:3],c(2,2,1)) **#벡터**

char1

num1 <- rep(1:3,c(2,2,1)) **#벡터**

num1

test1 <- data.frame(char1, num1) **#test1 데이터 프레임 생성**

test1

**# as.data.frame() :**모든 다른 종류의 자료객체들을 데이터 프레임으로 변환

mat <- matrix(c(1,2,3,4,5,6,7,8,9), ncol=3, byrow=T) #행 기준 3열의 행렬 생성

test3 <- as.data.frame(mat) # **matrix를 데이터 프레임으로 변환**

test3

test3 <- data.frame(mat) **## 다시 데이터 프레임으로 저장**

test3

#### 리스트(list) ####

▶ 리스트(List)는 서로 다른 형태(mode)의 자료를 포함하는 하나의 객체이다.

서로다른 형태란? : list(문자, 숫자, 논리형 객체)

▶ 리스트를 생성하기 위해서는 list 함수를 사용한다.

사용함수 : list()

son <- list(son.name = c("Minsu", "Minchul"), son.cnt = 2, son.age = c(2.6))

son

son.name, son.cnt, son.age는 list의 변수가 된다.

length(son) **#son 리스트 자료의 개수**

mode(son) **#son 리스트 자료의 형태**

names(son) **#son 리스트 각 구성요소의 이름**

#예제

exm <- list(c("Abe", "Bob", "Carol", "Deb"),c("Weight","Waist")) **#exm의 이름으로 list생성**

exm[[2]] **#리스트의 2번째 성분**

exm[[2]][2] **#2번째 성분 2번째 원소**

names(exm) <- c("Rows","Columns") **#exm 리스트에 성분 이름 부여**

exm$Rows **#exm의 Rows 성분만 표현**

exm$Rows[2] **#Rows 성분 2번째 원소 표현**

exm$Columns **#exm의 Columns 성분만 표현**