Day21; 20221004

날짜	
유형	
태그	

GitHub - u8yes/R

You can't perform that action at this time. You signed in with another tab or window. You signed out in another tab or window. Reload to refresh your session. Reload to refresh your session.

https://github.com/u8yes/R

u8yes/**R**

1 Contributor 0 Issues 1 Star 0 Forks

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/45280bb7-2def-406e-b995-2e7b83f6da3d/03._%EB%8D%B0%EC%9D%B4%ED%84%B0_%EC%9E%85%EC%B6%9C%EB%A0%A5.pdf

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/765e1771-4d33-4b57-a52f-52d2a0fad046/chap03_DataIO.r

---

```
# txt 파일 이용 객체 생성
getwd()
setwd("D:/heaven_dev/workspaces/R/data")

txtemp <- read.table('emp.txt')
txtemp
class(txtemp)


# csv 파일 이용
csvtemp
```

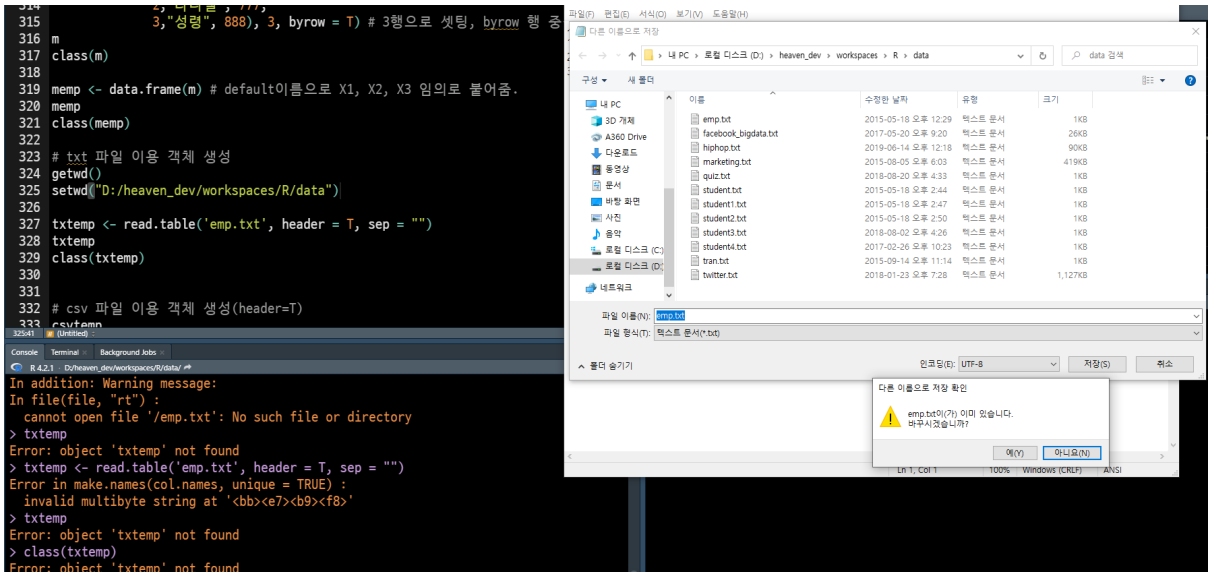
(Untitled)

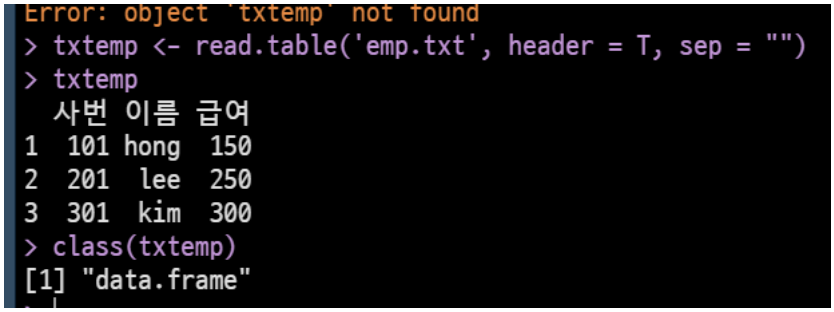
emp.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
사번 이름 급여
101 hong 150
201 lee 250
301  kim 300
```

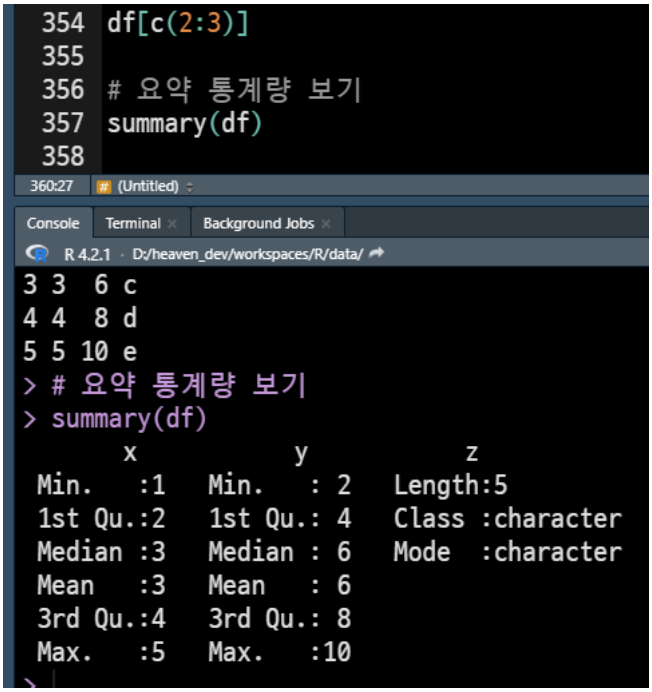
UTF-8 포맷으로 재 저장을 해줘야 함.





---

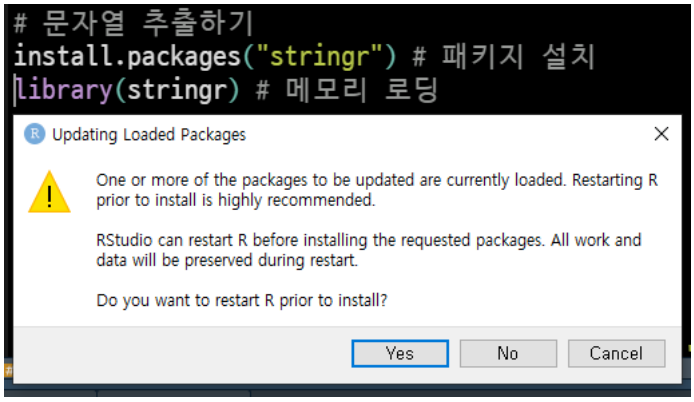
summary(df)



The Comprehensive R Archive Network

https://cran.biodisk.org/

이미 설치가 돼있을 경우 물어봄.



```
# Day19; 20220929

# chap02_DataStructure

#######################################
# chapter02. 데이터의 유형과 구조
#######################################

# 1. Vector 자료 구조
##  - C() 함수 이용 벡터 객체 생성
```

```
x <- c(1, 2.5, 3, 4, 5) # combine 함수
x

x <- c(1:20) # 콜론 : 범위
x

y <- 10:20
y

##  - seq() 함수 이용 벡터 객체 생성
x <- seq(1, 10, 2) # sequence(시작, 종료, 증감)
x

##  - rep() 함수 이용 벡터 객체 생성
help(rep) # ?rep
example(rep)
rep(1:3, 3) # replicate(대상, 반복수)
# 1 2 3 1 2 3 1 2 3
rep(1:3, each=3)
# 1 1 1 2 2 2 3 3 3

# union(), setdiff(), intersect() 함수 이용
x <- c(1, 3, 5, 7)
y <- c(3, 5)
x; y

union(x, y)     # 합집합(x+y) # 1 3 5 7
setdiff(x, y)   # 차집합(x-y) # 1 7
intersect(x, y) # 교집합(x^y) # 3 5


# 숫자형, 문자형, 논리형 벡터 생성
v1 <- c(33, -5, 20:23, 12, -2:3) # 33 -5 20 21 22 23 12 -2 -1  0  1  2  3
v1
v2 <- c(33, -5, 20:23, 12, "4") # 데이터를 모두 문자형으로 변환. # "33" "-5" "20" "21" "22" "23" "12" "4"
v2


# 한 줄에 명령문 중복 사용
v1; mode(v1) # mode()는 자료형을 반환해줌.
v2; mode(v2)

# 벡터에 컬럼명 지정
age <- c(30, 35, 40)
age
names(age) <- c("홍길동", "이순자", "강감촌치킨")
age
#    홍길동     이순자     강감촌치킨
#    30         35         40
age <- NULL # age 변수 데이터 삭제
age

# 벡터 자료 참조하기
a <- c(1:50)
a[10] # index : 1부터 시작
a[c(10:45)] # 10 ~ 45 사이의 벡터 원소 출력
a[c(10, 20, 30, 40)]
a[10:(length(a)-5)] # 모든 함수가 올 수 있다. 그것을 예상하고 있어야 한다.

# 잘못된 벡터 첨자 사용 예
a[5, 10] # Error in a[5, 10] : incorrect number of dimensions


# c() 함수에서 콤마 사용 예
v1 <- c(33, -5, 20:23, 12, -2:3)
v1
v1[3:6] # 20 21 22 23
v1[c(4, 5:8, 9)]


# 음수 값으로 첨자 지정 예
v1
v1[-1] # 첫번째 인덱스를 제외시키고 실행시켜라 = 해당 위치의 원소를 제외한 값 출력
# -5 20 21 22 23 12 -2 -1  0  1  2  3
v1
v1[-c(2,4)]


# 패키지 설치와 메모리 로딩
install.packages("RSADBE") # 패키지(데이터) 설치
library(RSADBE)            # 패키지를 메모리에 로드

data("Severity_Counts")    # RSADBE 패키지에서 제공되는 데이터 셋 가져오기.
str(Severity_Counts)# Structure(구조)

# 패키지에서 제공되는 벡터데이터 셋 보기
Severity_Counts
```

```
# 2. Matrix 자료 구조
args(matrix) # (data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)

# 벡터 이용 행렬 객체 생성
m <- matrix(c(1:5))
m  # 5행 1열

# 벡터의 열 우선으로 행렬 객체 생성
?matrix
m <- matrix(c(1:10), nrow = 2) # 2행 5열
m

# 행과 열의 수가 일치하지 않는 경우 예
m <- matrix(c(1:11), nrow = 2)
m

# 벡터의 행 우선으로 행렬 객체 생성
m <- matrix(c(1:11), nrow = 2, byrow = T) # 행 우선
m

m <- matrix(c(1:10), byrow = T) # 주의 - 여전히 10행 1열
# (한글) nrow 또는 ncol 중 하나가 주어지지 않으면 데이터 길이와 다른 매개 변수에서 추론하려고 시도합니다. 둘 다 주어지지 않으면 한 열 행렬이 반환됩니다.
# (english) If one of nrow or ncol is not given, an attempt is made to infer it from the length of data and the other parameter. If ne
# 즉 byrow 속성은 적용되지 않음.
m

m <- matrix(c(1:10), ncol = 10) # 의도한 1행 10열 # 1    2    3    4    5    6    7    8    9    10
m

# 행 묶음으로 행렬 객체 생성
x1 <- c(5, 40, 50:52)
x2 <- c(30, 5, 6:8)
mr <- rbind(x1, x2)
mr

# 열 묶음으로 행렬 객체 생성
mc <- cbind(x1, x2)
mc

# 2행으로 행렬 객체 생성
m3 <- matrix(10:19, 2)
m3

# 자료와 객체 type 보기
mode(m3) # "numeric"
class(m3) # "matrix" "array"


# 행렬 객체에 첨자로 접근
m3[2,3] # 2행 3열의 데이터 1개 : 15
m3[1,] # 1행 전체 # 10 12 14 16 18
m3[,5] # 18 19
m3[1, c(2:5)] # 1행에서 2~5열까지의 데이터터 # 12 14 16 18
m3[1, c(2, 5)] # 1행에서 2열, 5열 데이터 2개 # 12 18
m3

# 3행 3열로 행렬 객체 생성
x <- matrix(c(1:9), nrow = 3, ncol = 3) # nrow, ncol, nro, nco, nr, nc 모두 가능하다. 하지만 n, n은 안 된다.
x

# 자료의 개수 보기
length(x) # 데이터 개수
ncol(x); nrow(x) # 열 / 행 수


# apply() 함수 적용
apply(x, 1, max) # 1번은 행, 2번은 열 # 행 단위 최대값
apply(x, 1, min) # 행 단위 최소값
apply(x, 2, mean) # 열 단위 평균값
x

# Day 20; 20220930
# 사용자 정의 적용
f <- function(x){ # x : 매개변수
  x * c(1,2,3)
}

# 행(1) 우선 순서로 사용자 정의 함수 적용
result <- apply(x, 1, f)
result

# 열(2) 우선 순서로 사용자 정의 함수 적용
result <- apply(x, 2, f)
result
```

```
# 행렬 객체에 컬럼명 지정하기
colnames(x) <- c('one', 'two', 'three')
x

## 3. Array 자료 구조

# 3차원 배열 객체 생성하기
vec <- c(1:12) # 12개 벡터 객체 생성
arr <- array(vec, c(3,2,2)) # 3차원일 때는 2개의 ,, 컴마가 있다.
# 3행 2열 2면 만들라
arr

# 3차원 배열 객체 자료 조회
arr[2,1,2] # 2행1열2면
arr[,,1] # 1면
arr[,,2] # 2면
arr[2,,1] # 1면2행

# 배열 자료형과 자료 구조
mode(arr); class(arr)


# 데이터 셋 가져오기
library(RSADBE)
data(Bug_Metrics_Software)
str(Bug_Metrics_Software)

# 데이터 셋 자료보기
Bug_Metrics_Software # array 형태로 보여줌.


## 4. List 자료 구조 # 자바의 map개념과 같다

# key를 이용하여 value에 접근하기
member <- list(name=c("홍길동", "유관순"),
               age=c(35, 25),
               address=c("제주도","천국"),
               gender=c("남자","여자"),
               htype=c("아파트","왕국")) # 변수에 접근할 때 자바는 '.'으로 접근하듯이 R은 key값으로 '$'로 접근한다.
member

# key를 이용하여 value에 접근하기
member$name
member$name[1]
member$name[2]
member$name[3] <- "이명박" # 데이터 추가 가능하다.

member$age <- 45 # 데이터 수정(※주의: 하나의 값으로 수정.)
member

member$id <- c("hong","yu") # 데이터 항목 추가
member

member$age <- NULL
member

# 1개 값을 갖는 리스트 객체 생성
list <- list("lee","이명박",70)
list
"""
[[1]]         ----------------------> key(생략) [[n]]
[1] "lee"     ----------------------> value[n]

[[2]]
[1] "이명박"

[[3]]
[1] 70
""" # """ 3개는 그 안의 모든 것을 문자열로 인식하게 만들어줌. error표시는 그냥 무시하면 된다.

# 1개 이상의 값을 갖는 리스트 객체 생성
num <- list(c(1:5), c(6:10))
num

# 리스트 자료구조 -> 벡터 구조로 변경하기
unlist <- unlist(num) # 1차원의 배열로 형변환해줌.
unlist

# 리스트 객체에 함수 적용하기
# list data 처리 함수
a <- list(c(1:5))
b <- list(c(6:10))
a; b

c(a,b)

c <- lapply(c(a,b), max) # list로 결과 반환
```

```
c

mode(c); class(c)  # "list" "list" # key 중심으로 자료형을 보기 때문에 key로 접근해서 봐야 알 수 있다.

# 리스트 형식을 벡터(1차원) 형식으로 반환하기
c <- sapply(c(a,b), max)
c
mode(c); class(c) # 자료형:"numeric", 자료구조:"integer"

# 다차원 리스트 객체 생성
multi_list <- list(list(1,2,3), list(10,20,30), list(100,200,300))
multi_list
multi_list <- list(c1=list(1,2,3), c2=list(10,20,30), c3=list(100,200,300))
multi_list

multi_list$c1
multi_list$c2
multi_list$c3
mode(multi_list); class(multi_list)

# 다차원 리스트를 열 단위로 바인딩
d <- do.call(cbind, multi_list)
d

d <- do.call(rbind, multi_list)
d

class(d) # "matrix"

## 5. Data Frame 자료 구조 # 서로 다른 자료형으로 컬럼별로 가질 수 있다.

# 벡터 이용 객체 생성
no  <- c(1, 2, 3)
name <- c("베드로","요한","야고보")
pay <- c(150,250,300)
vemp <- data.frame(No=no,Name=name,Pay=pay)
vemp
class(vemp) # "data.frame"

# matrix 이용 객체 생성 # Day21; 20221004
args(matrix) # 매개변수에 대한 inform을 보여준다. # function (data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL) # byrow 행우선
m <- matrix(c(1,"베드로", 153,
              2,"다니엘", 777,
              3,"성령", 888), 3, byrow = T) # 3행으로 셋팅, byrow 행 중심으로
m
class(m)

memp <- data.frame(m) # default이름으로 X1, X2, X3 임의로 붙어줌.
memp
class(memp)

# txt 파일 이용 객체 생성
getwd()
setwd("D:/heaven_dev/workspaces/R/data")

txtemp <- read.table('emp.txt', header = T, sep = "") # header = T 이름이 있다면 col.names에 이름을 보여준다. HTML <th> 때 header같은 느낌
txtemp
class(txtemp)


# csv 파일 이용 객체 생성(header=T)
csvtemp <- read.csv('emp.csv', header = T)
csvtemp; class(csvtemp)


# csv 파일 이용 객체 생성(header=F)
name <- c("사번", "이름", "급여")
csvtemp2 <- read.csv('emp2.csv', header = F, col.names = name) # col.names에 name변수를 넣어주면 됨
csvtemp2


# 데이터프레임 만들기
df <- data.frame(x=c(1:5),y=seq(2,10,2), z=c('a', 'b', 'c', 'd', 'e')) # seq(시작, 끝, 2개씩 더해줌)
df

# 데이터프레임 컬럼명 참조
df$x; df$y; df$z

# 자료구조, 열수, 행수, 컬럼명 보기
str(df) # 5 obs. of  3 variables: # 5개의 오브젝트와 3개의 변수
ncol(df)
nrow(df)
df[c(2:3)]

# 요약 통계량 보기
summary(df) # 수치형의 값에서 필요
```

```
# 데이터프레임 자료에 함수 적용
apply(df[,c(1,2)], 2, sum) # 데이터, 행(1)·열(2), 함수를 수행
# lapply는 리스트로, sapply는 vector형태로 쭉 펼쳐서 처리.

# 데이터프레임의 부분 객체 만들기 # 전처리에서 가장 많이 사용하는 함수
# 내가 원하는 데이터를 필터하는 용도도
x1 <- subset(df, x >= 3) # x가 3이상인 레코드 대상
x1

y1 <- subset(df, y <= 8) # y가 8이하인 레코드 대상
y1

# 두 개의 조건으로 부분 객체 만들기
xyand <- subset(df, x>=2 & y<=6) # R에서는 '&&' 2개 사용 안 하고 '&' 1개만 사용
xyand

xyor <- subset(df, x>=2 | y<=6)
xyor

# student 데이터프레임 만들기
sid <- c('A','B','C','D')
score <- c(90, 80, 70, 60)
subject <- c('컴퓨터', '국어국문', '소프트웨어', '유아교육')

student <- data.frame(sid, score, subject)
student

# 자료형과 자료구조 보기
mode(student); class(student) # list라는 자료형, data.frame
str(sid); str(score); str(subject) # sid character, score number, subject character
str(student)

# 두 개 이상의 데이터프레임 병합하기
height <- data.frame(id=c(1,2), h=c(180, 175))
weight <- data.frame(id=c(1,2), w=c(80,75))
height; weight

person <- merge(height, weight, by.x = "id", by.y = "id") # id를 기준으로 병합
person


# galton 데이터 셋 가져오기
install.packages("UsingR") # 패키지 설치
library(UsingR) # 패키지 메모리에 로드
data("galton") # galton 데이터 셋 가져오기
View(galton)

# galton 데이터 셋 구조 보기
str(galton) # 928 obs. of  2 variables:
dim(galton) # 928   2
head(galton, 20) # 상위 1~20개까지
head(galton) # default 갯수는 6개
tail(galton, 20) # 하위 1~20개까지
tail(galton)

## 6. 문자열 처리

# 문자열 추출하기
install.packages("stringr") # 패키지 설치
library(stringr) # 메모리 로딩

# 형식) str_extract('문자열', '정규표현식')
str_extract("홍길동35이순신45강감찬50","[0-9]{2}") #[0-9]사이의 숫자값, {2}연속 2자리인 것만 추출하겠다.
str_extract_all("홍길동35이순신45강감찬50","[0-9]{2}") # 모든 숫자형 문자열 데이터를 추출


# 반복수를 지정하여 영문자 추출
string <- 'hongkildong105lee1002you25베드로2005'
str_extract_all(string, '[a-z]{3}') # 3자 연속된 알파벳 추출 # 소문자, 대문자 구분
str_extract_all(string, '[a-z]{3,}') # 3글자 이상 연속된 알파벳 추출
str_extract_all(string, '[a-z]{3,5}') # 3~5글자 범위 안에 속하는 알파벳 추출

# 특정 단어 추출
str_extract_all(string, '유관순')
str_extract_all(string, '베드로')


# 한글, 영문자, 숫자 추출하기
str_extract_all(string, 'hong')
str_extract_all(string, '25')
str_extract_all(string, '[가-힣]{3}') # 한글 검색
str_extract_all(string, '[a-z]{3}') # 소문자 검색
str_extract_all(string, '[A-Z]{3}') # 대문자 검색
str_extract_all(string, '[0-9]{4}') # 숫자 검색


# 한글, 영문자, 숫자를 제외한 나머지 추출하기
```

```
str_extract_all(string, '[^a-z]') # ^표시가 제외외
str_extract_all(string, '[^a-z]{4}') # 알파벳을 제외한 4개의 연속
str_extract_all(string, '[^가-힣]{5}')
str_extract_all(string, '[^0-9]{3}')


# 주민등록번호 검사하기
jumin <- '123456-3234567'
str_extract_all(jumin, '[0-9]{6}-[1234][0-9]{6}')
str_extract_all(jumin, '\\d{6}-[1234]\\d{6}') # \\두번을 써야지 문자 '\'로 인식
# 숫자는 \d
# 지정된 길이의 단어 추출하기
name <- '홍길동1234,이순신5678,강감찬1012'
str_extract_all(name, '\\w{7,}') # '\\w' 특수문자 제외한 모든 문자
# '\\w': 한글, 영문자, 숫자는 포함.
# '\\W': only 특수문자만 선택.
str_extract_all(name, '\\W')

# 문자열 위치(index) 구하기
string <- 'hongkd105leess1002you25강감찬2005'
str_locate(string, '강감찬') # '강'위치:24,  '찬'위치:26

# 문자열 길이 구하기
string <- 'hongkild105lee1002you25강감찬2005'
len <- str_length(string) # 30
len

# 부분 문자열
string_sub <- str_sub(string, 1, len-7) # 30-7 = 23개까지의 데이터만을
string_sub

string_sub <- str_sub(string, 1, 23)
string_sub

# 대문자, 소문자 변경하기
str_to_upper(string_sub)
str_to_lower(string_sub)

# 문자열 교체하기
string_rep <- str_replace(string_sub, 'hongkild105', '홍길동35,')
string_rep <- str_replace(string_rep, 'lee1002', '이순신45,')
string_rep <- str_replace(string_rep, 'you25', '유관순25,')
string_rep # "홍길동35,이순신45,유관순25,"


# 문자열 결합하기
string_c <- str_c(string_rep, '강감찬55') # 추가하고 싶을 때 str_c
string_c


# 문자열 분리하기
string_sp <- str_split(string_c, ',') # 각각의 하나의 데이터셋으로 출력
string_sp

# 문자열 합치기
string_vec <- c('홍길동35', '이순신45', '유관순25', '강감찬55')
string_vec

string_join <- paste(string_vec, collapse = ',') # 하나의 벡터로 합침.
string_join
```

```
> df[c(2:3)]
   y z
1  2 a
2  4 b
3  6 c
4  8 d
5 10 e
> # 요약 통계량 보기
> summary(df)
       x          y          z
 Min.   :1   Min.   : 2   Length:5
 1st Qu.:2   1st Qu.: 4   Class :character
 Median :3   Median : 6   Mode  :character
 Mean   :3   Mean   : 6
 3rd Qu.:4   3rd Qu.: 8
 Max.   :5   Max.   :10
> df[c(1:3)]
  x  y z
1 1  2 a
2 2  4 b
3 3  6 c
4 4  8 d
5 5 10 e
```

```
> # 요약 통계량 보기
> summary(df)
       x           y           z
 Min.   :1   Min.   : 2   Length:5
 1st Qu.:2   1st Qu.: 4   Class :character
 Median :3   Median : 6   Mode  :character
 Mean   :3   Mean   : 6
 3rd Qu.:4   3rd Qu.: 8
 Max.   :5   Max.   :10
> # 데이터프레임 자료에 함수 적용
> apply(df[,c(1,2)], 2, sum)
 x  y
15 30
> # 데이터프레임 자료에 함수 적용
> apply(df[,c(1,2)], 2, sum) # 데이터, 행(1)·열(2), 함수를 수행
 x  y
15 30
> # 데이터프레임의 부분 객체 만들기 # 전처리에서 가장 많이 사용하는 함수
> x1 <- subset(df, x >= 3) # x가 3이상인 레코드 대상
> x1
  x  y z
3 3  6 c
4 4  8 d
5 5 10 e
> # 데이터프레임의 부분 객체 만들기 # 전처리에서 가장 많이 사용하는 함수
> # 내가 원하는 데이터를 필터하는 용도도
> x1 <- subset(df, x >= 3) # x가 3이상인 레코드 대상
> x1
  x  y z
3 3  6 c
4 4  8 d
5 5 10 e
> y1 <- subset(df, y <= 8) # y가 8이하인 레코드 대상
> y1
  x y z
1 1 2 a
2 2 4 b
3 3 6 c
4 4 8 d
> # 두 개의 조건으로 부분 객체 만들기
> xyand <- subset(df, x>=2 & y<=6)
> xyand
  x y z
2 2 4 b
3 3 6 c
> xyor <- subset(df, x>=2 | y<=6)
> xyor
  x  y z
1 1  2 a
2 2  4 b
3 3  6 c
4 4  8 d
5 5 10 e
> # student 데이터프레임 만들기
> sid <- c('A','B','C','D')
> score <- c(90, 80, 70, 60)
> subject <- c('컴퓨터', '국어국문', '소프트웨어', '유아교육')
> student <- data.frame(sid, score, subject)
> student
  sid score    subject
1   A    90     컴퓨터
2   B    80   국어국문
3   C    70 소프트웨어
4   D    60   유아교육
> # 자료형과 자료구조 보기
> mode(student); class(student) # list, data.frame
[1] "list"
[1] "data.frame"
> str(sid); str(score); str(subject)
 chr [1:4] "A" "B" "C" "D"
 num [1:4] 90 80 70 60
 chr [1:4] "컴퓨터" "국어국문" "소프트웨어" "유아교육"
> str(student)
'data.frame': 4 obs. of  3 variables:
 $ sid    : chr  "A" "B" "C" "D"
 $ score  : num  90 80 70 60
 $ subject: chr  "컴퓨터" "국어국문" "소프트웨어" "유아교육"
> str(student)
'data.frame': 4 obs. of  3 variables:
 $ sid    : chr  "A" "B" "C" "D"
 $ score  : num  90 80 70 60
 $ subject: chr  "컴퓨터" "국어국문" "소프트웨어" "유아교육"
> student
  sid score    subject
1   A    90     컴퓨터
2   B    80   국어국문
3   C    70 소프트웨어
```

```
4   D    60   유아교육
> # 두 개 이상의 데이터프레임 병합하기
> height <- data.frame(id=c(1,2), h=c(180, 175))
> weight <- data.frame(id=c(1,2), w=c(80,75))
> height; weight
  id   h
1  1 180
2  2 175
  id  w
1  1 80
2  2 75
> person <- merge(id,h,w)
Error in merge(id, h, w) : object 'id' not found
> person
function (given = NULL, family = NULL, middle = NULL, email = NULL,
    role = NULL, comment = NULL, first = NULL, last = NULL)
{
    args <- list(given = given, family = family, middle = middle,
        email = email, role = role, comment = comment, first = first,
        last = last)
    if (all(vapply(args, is.null, NA)))
        return(.person())
    args <- lapply(args, .listify)
    args_length <- lengths(args)
    if (!all(args_length_ok <- args_length %in% c(1L, max(args_length))))
        warning(gettextf("Not all arguments are of the same length, the following need to be recycled: %s",
            paste(names(args)[!args_length_ok], collapse = ", ")),
            domain = NA)
    args <- lapply(args, function(x) rep_len(x, max(args_length)))
    person1 <- function(given = NULL, family = NULL, middle = NULL,
        email = NULL, role = NULL, comment = NULL, first = NULL,
        last = NULL) {
        if (!.is_not_nonempty_text(first)) {
            if (!.is_not_nonempty_text(given))
                stop(gettextf("Use either %s or %s/%s but not both.",
                  sQuote("given"), sQuote("first"), sQuote("middle")),
                  domain = NA)
            warning(gettextf("It is recommended to use %s instead of %s.",
                sQuote("given"), sQuote("first")), domain = NA)
            given <- first
        }
        if (!.is_not_nonempty_text(middle)) {
            warning(gettextf("It is recommended to use %s instead of %s.",
                sQuote("given"), sQuote("middle")), domain = NA)
            given <- c(given, unlist(strsplit(middle, "[[:space:]]+")))
        }
        if (!.is_not_nonempty_text(last)) {
            if (!.is_not_nonempty_text(family))
                stop(gettextf("Use either %s or %s but not both.",
                  sQuote("family"), sQuote("last")), domain = NA)
            warning(gettextf("It is recommended to use %s instead of %s.",
                sQuote("family"), sQuote("last")), domain = NA)
            family <- last
        }
        if (.is_not_nonempty_text(given))
            given <- NULL
        if (.is_not_nonempty_text(family))
            family <- NULL
        if (.is_not_nonempty_text(email))
            email <- NULL
        if (.is_not_nonempty_text(role)) {
            if (!is.null(role))
                warning(sprintf(ngettext(length(role), "Invalid role specification: %s.",
                  "Invalid role specifications: %s."), paste(sQuote(role),
                  collapse = ", ")), domain = NA)
            role <- NULL
        }
        if (.is_not_nonempty_text(comment))
            comment <- NULL
        if (length(role))
            role <- .canonicalize_person_role(role)
        if (length(comment)) {
            ind <- grepl(sprintf("^https?://orcid.org/%s$", tools:::.ORCID_iD_regexp),
                comment)
            if (any(ind)) {
                if (is.null(names(comment)))
                  names(comment) <- ifelse(ind, "ORCID", "")
                else names(comment)[ind] <- "ORCID"
            }
        }
        rval <- list(given = given, family = family, role = role,
            email = email, comment = comment)
        if (any(ind <- (lengths(rval) == 0L)))
            rval[ind] <- vector("list", length = sum(ind))
        if (all(vapply(rval, is.null, NA)))
            NULL
        else rval
```

```
    }
    force(person1)
    rval <- lapply(seq_along(args$given), function(i) with(args,
        person1(given = given[[i]], family = family[[i]], middle = middle[[i]],
            email = email[[i]], role = role[[i]], comment = comment[[i]],
            first = first[[i]], last = last[[i]])))
    .person(rval[!vapply(rval, is.null, NA)])
}
<bytecode: 0x000001e3962f8428>
<environment: namespace:utils>
> person <- merge(h,w)
Error in merge(h, w) : object 'h' not found
> person
function (given = NULL, family = NULL, middle = NULL, email = NULL,
    role = NULL, comment = NULL, first = NULL, last = NULL)
{
    args <- list(given = given, family = family, middle = middle,
        email = email, role = role, comment = comment, first = first,
        last = last)
    if (all(vapply(args, is.null, NA)))
        return(.person())
    args <- lapply(args, .listify)
    args_length <- lengths(args)
    if (!all(args_length_ok <- args_length %in% c(1L, max(args_length))))
        warning(gettextf("Not all arguments are of the same length, the following need to be recycled: %s",
            paste(names(args)[!args_length_ok], collapse = ", ")),
            domain = NA)
    args <- lapply(args, function(x) rep_len(x, max(args_length)))
    person1 <- function(given = NULL, family = NULL, middle = NULL,
        email = NULL, role = NULL, comment = NULL, first = NULL,
        last = NULL) {
        if (!.is_not_nonempty_text(first)) {
            if (!.is_not_nonempty_text(given))
                stop(gettextf("Use either %s or %s/%s but not both.",
                  sQuote("given"), sQuote("first"), sQuote("middle")),
                  domain = NA)
            warning(gettextf("It is recommended to use %s instead of %s.",
                sQuote("given"), sQuote("first")), domain = NA)
            given <- first
        }
        if (!.is_not_nonempty_text(middle)) {
            warning(gettextf("It is recommended to use %s instead of %s.",
                sQuote("given"), sQuote("middle")), domain = NA)
            given <- c(given, unlist(strsplit(middle, "[[:space:]]+")))
        }
        if (!.is_not_nonempty_text(last)) {
            if (!.is_not_nonempty_text(family))
                stop(gettextf("Use either %s or %s but not both.",
                  sQuote("family"), sQuote("last")), domain = NA)
            warning(gettextf("It is recommended to use %s instead of %s.",
                sQuote("family"), sQuote("last")), domain = NA)
            family <- last
        }
        if (.is_not_nonempty_text(given))
            given <- NULL
        if (.is_not_nonempty_text(family))
            family <- NULL
        if (.is_not_nonempty_text(email))
            email <- NULL
        if (.is_not_nonempty_text(role)) {
            if (!is.null(role))
                warning(sprintf(ngettext(length(role), "Invalid role specification: %s.",
                  "Invalid role specifications: %s."), paste(sQuote(role),
                  collapse = ", ")), domain = NA)
            role <- NULL
        }
        if (.is_not_nonempty_text(comment))
            comment <- NULL
        if (length(role))
            role <- .canonicalize_person_role(role)
        if (length(comment)) {
            ind <- grepl(sprintf("^https?://orcid.org/%s$", tools:::.ORCID_iD_regexp),
                comment)
            if (any(ind)) {
                if (is.null(names(comment)))
                  names(comment) <- ifelse(ind, "ORCID", "")
                else names(comment)[ind] <- "ORCID"
            }
        }
        rval <- list(given = given, family = family, role = role,
            email = email, comment = comment)
        if (any(ind <- (lengths(rval) == 0L)))
            rval[ind] <- vector("list", length = sum(ind))
        if (all(vapply(rval, is.null, NA)))
            NULL
        else rval
    }
```

```
    force(person1)
    rval <- lapply(seq_along(args$given), function(i) with(args,
        person1(given = given[[i]], family = family[[i]], middle = middle[[i]],
            email = email[[i]], role = role[[i]], comment = comment[[i]],
            first = first[[i]], last = last[[i]])))
    .person(rval[!vapply(rval, is.null, NA)])
}
<bytecode: 0x000001e3962f8428>
<environment: namespace:utils>
> ?merge
> person <- merge(height, weight, by.x = "id", by.y = "id") # id를 기준으로 병합
> person
  id   h  w
1  1 180 80
2  2 175 75
> # galton 데이터 셋 가져오기
> install.packages("UsingR") # 패키지 설치
WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of

https://cran.rstudio.com/bin/windows/Rtools/
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.2/UsingR_2.0-7.zip'
Content type 'application/zip' length 2128728 bytes (2.0 MB)
downloaded 2.0 MB

패키지 ‘UsingR’를 성공적으로 압축해제하였고 MD5 sums 이 확인되었습니다

다운로드된 바이너리 패키지들은 다음의 위치에 있습니다
  C:\Users\tjouen-jr\AppData\Local\Temp\RtmpCiIFEN\downloaded_packages
> library(UsingR) # 패키지 메모리에 로드
필요한 패키지를 로딩중입니다: MASS
필요한 패키지를 로딩중입니다: HistData
필요한 패키지를 로딩중입니다: Hmisc
필요한 패키지를 로딩중입니다: lattice
필요한 패키지를 로딩중입니다: survival
필요한 패키지를 로딩중입니다: Formula
필요한 패키지를 로딩중입니다: ggplot2

다음의 패키지를 부착합니다: ‘Hmisc’

The following objects are masked from ‘package:base’:

    format.pval, units


다음의 패키지를 부착합니다: ‘UsingR’

The following object is masked from ‘package:survival’:

    cancer

> data("galton") # galton 데이터 셋 가져오기
> person <- merge(height, weight, by.x = "id", by.y = "id") # id를 기준으로 병합
> person
  id   h  w
1  1 180 80
2  2 175 75
> hist(person)
> hist(person)
> data("galton") # galton 데이터 셋 가져오기
> # galton 데이터 셋 가져오기
> install.packages("UsingR") # 패키지 설치
Error in install.packages : Updating loaded packages

Restarting R session...

Error: Unable to establish connection with R session when executing 'console_input'
> data("galton") # galton 데이터 셋 가져오기
Warning message:
In data("galton") : 데이터셋 ‘galton’을 찾을 수 없습니다
> install.packages("UsingR")
WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of

https://cran.rstudio.com/bin/windows/Rtools/
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.2/UsingR_2.0-7.zip'
Content type 'application/zip' length 2128728 bytes (2.0 MB)
downloaded 2.0 MB

패키지 ‘UsingR’를 성공적으로 압축해제하였고 MD5 sums 이 확인되었습니다

다운로드된 바이너리 패키지들은 다음의 위치에 있습니다
  C:\Users\tjouen-jr\AppData\Local\Temp\Rtmpecb7uM\downloaded_packages
> # galton 데이터 셋 구조 보기
> str(galton)
'data.frame': 928 obs. of  2 variables:
 $ child : num  61.7 61.7 61.7 61.7 61.7 62.2 62.2 62.2 62.2 62.2 ...
 $ parent: num  70.5 68.5 65.5 64.5 64 67.5 67.5 67.5 66.5 66.5 ...
> dim(galton)
```

```
[1] 928   2
> head(galton, 20)
   child parent
1   61.7   70.5
2   61.7   68.5
3   61.7   65.5
4   61.7   64.5
5   61.7   64.0
6   62.2   67.5
7   62.2   67.5
8   62.2   67.5
9   62.2   66.5
10  62.2   66.5
11  62.2   66.5
12  62.2   64.5
13  63.2   70.5
14  63.2   69.5
15  63.2   68.5
16  63.2   68.5
17  63.2   68.5
18  63.2   68.5
19  63.2   68.5
20  63.2   68.5
> head(galton) # default 갯수:6
  child parent
1  61.7   70.5
2  61.7   68.5
3  61.7   65.5
4  61.7   64.5
5  61.7   64.0
6  62.2   67.5
> View(galton)
> View(galton)
> dim(galton) # 928   2
[1] 928   2
> head(galton, 20)
   child parent
1   61.7   70.5
2   61.7   68.5
3   61.7   65.5
4   61.7   64.5
5   61.7   64.0
6   62.2   67.5
7   62.2   67.5
8   62.2   67.5
9   62.2   66.5
10  62.2   66.5
11  62.2   66.5
12  62.2   64.5
13  63.2   70.5
14  63.2   69.5
15  63.2   68.5
16  63.2   68.5
17  63.2   68.5
18  63.2   68.5
19  63.2   68.5
20  63.2   68.5
> head(galton) # default 갯수:6
  child parent
1  61.7   70.5
2  61.7   68.5
3  61.7   65.5
4  61.7   64.5
5  61.7   64.0
6  62.2   67.5
> # 문자열 추출하기
> install.packages("stringr") # 패키지 설치
WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of

https://cran.rstudio.com/bin/windows/Rtools/
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.2/stringr_1.4.1.zip'
Content type 'application/zip' length 218949 bytes (213 KB)
downloaded 213 KB

패키지 ‘stringr’를 성공적으로 압축해제하였고 MD5 sums 이 확인되었습니다

다운로드된 바이너리 패키지들은 다음의 위치에 있습니다
  C:\Users\tjouen-jr\AppData\Local\Temp\RtmpecB7uM\downloaded_packages
> library(stringr) # 메모리 로딩
> # 형식) str_extract('문자열', '정규표현식')
> str_extract("홍길동35이순신45강감찬50","")
[1] "홍"
> str_extract_all("홍길동35이순신45강감찬50","")
[[1]]
 [1] "홍" "길" "동" "3"  "5"  "이" "순" "신" "4"  "5"  "강" "감" "찬" "5"  "0"

> tail(galton)
```

```
    child parent
923  73.7   70.5
924  73.7   69.5
925  73.7   69.5
926  73.7   69.5
927  73.7   69.5
928  73.7   69.5
> tail(galton, 20)
    child parent
909  73.2   69.5
910  73.2   69.5
911  73.2   69.5
912  73.2   68.5
913  73.2   68.5
914  73.2   68.5
915  73.7   72.5
916  73.7   72.5
917  73.7   72.5
918  73.7   72.5
919  73.7   71.5
920  73.7   71.5
921  73.7   70.5
922  73.7   70.5
923  73.7   70.5
924  73.7   69.5
925  73.7   69.5
926  73.7   69.5
927  73.7   69.5
928  73.7   69.5
> tail(galton)
    child parent
923  73.7   70.5
924  73.7   69.5
925  73.7   69.5
926  73.7   69.5
927  73.7   69.5
928  73.7   69.5
> # 문자열 추출하기
> install.packages("stringr") # 패키지 설치
Error in install.packages : Updating loaded packages
> library(stringr) # 메모리 로딩
> # 형식) str_extract('문자열', '정규표현식')
> str_extract("홍길동35이순신45강감찬50","")
[1] "홍"
> str_extract_all("홍길동35이순신45강감찬50","")
[[1]]
 [1] "홍" "길" "동" "3"  "5"  "이" "순" "신" "4"  "5"  "강" "감" "찬" "5"  "0"

> # 반복수를 지정하여 영문자 추출
> string <- 'hongkildong105lee1002you25강감찬2005'
> str_extract_all(string, '')
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> str_extract_all(string, '')
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> str_extract_all(string, '')
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> # 특정 단어 추출
> str_extract_all(string, '유관순')
[[1]]
character(0)

> str_extract_all(string, '강감찬')
[[1]]
[1] "강감찬"

> # 한글, 영문자, 숫자 추출하기
> str_extract_all(string, 'hong')
[[1]]
[1] "hong"

> str_extract_all(string, '25')
[[1]]
[1] "25"

> str_extract_all(string, '') # 한글 검색
```

```
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> str_extract_all(string, '') # 소문자 검색
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> str_extract_all(string, '') # 대문자 검색
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> str_extract_all(string, '') # 숫자 검색
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> # 한글, 영문자, 숫자를 제외한 나머지 추출하기
> str_extract_all(string, '')
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> str_extract_all(string, '')
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> str_extract_all(string, '')
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> str_extract_all(string, '')
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "강" "감" "찬" "2"
[31] "0"  "0"  "5"

> # 주민등록번호 검사하기
> jumin <- '123456-3234567'
> str_extract_all(jumin, '')
[[1]]
 [1] "1" "2" "3" "4" "5" "6" "-" "3" "2" "3" "4" "5" "6" "7"

> str_extract_all(jumin, '')
[[1]]
 [1] "1" "2" "3" "4" "5" "6" "-" "3" "2" "3" "4" "5" "6" "7"

> # 지정된 길이의 단어 추출하기
> name <- '홍길동1234,이순신5678,강감찬1012'
> str_extract_all(name, '')
[[1]]
 [1] "홍" "길" "동" "1"  "2"  "3"  "4"  ","  "이" "순" "신" "5"  "6"  "7"  "8"
[16] ","  "강" "감" "찬" "1"  "0"  "1"  "2"

> str_extract_all(name, '')
[[1]]
 [1] "홍" "길" "동" "1"  "2"  "3"  "4"  ","  "이" "순" "신" "5"  "6"  "7"  "8"
[16] ","  "강" "감" "찬" "1"  "0"  "1"  "2"

> # 문자열 위치(index) 구하기
> string <- 'hongkd105leess1002you25강감찬2005'
> str_locate(string, '강감찬')
     start end
[1,]    24  26
> # 문자열 길이 구하기
> string <- 'hongkild105lee1002you25강감찬2005'
> len <- str_length(string) # 30
> len
[1] 30
> # 부분 문자열
> string_sub <- str_sub(string, 1, len-7)
> string_sub
[1] "hongkild105lee1002you25"
> string_sub <- str_sub(string, 1, 23)
> string_sub
[1] "hongkild105lee1002you25"
```

```
> # 대문자, 소문자 변경하기
> str_to_upper(string_sub)
[1] "HONGKILD105LEE1002YOU25"
> str_to_lower(string_sub)
[1] "hongkild105lee1002you25"
> # 문자열 교체하기
> string_rep <- str_replace(string_sub, 'hongkild105', '홍길동35,')
> string_rep <- str_replace(string_rep, 'lee1002', '이순신45,')
> string_rep <- str_replace(string_rep, 'you25', '유관순25,')
> string_rep
[1] "홍길동35,이순신45,유관순25,"
> # 문자열 결합하기
> string_c <- str_c(string_rep, '강감찬55')
> string_c
[1] "홍길동35,이순신45,유관순25,강감찬55"
> # 문자열 분리하기
> string_sp <- str_split(string_c, ',')
> string_sp
[[1]]
[1] "홍길동35" "이순신45" "유관순25" "강감찬55"

> # 문자열 합치기
> string_vec <- c('홍길동35', '이순신45', '유관순25', '강감찬55')
> string_vec
[1] "홍길동35" "이순신45" "유관순25" "강감찬55"
> string_join <- paste(string_vec, collapse = ',')
> string_join
[1] "홍길동35,이순신45,유관순25,강감찬55"
>
>
>
>
>
>
> # 형식) str_extract('문자열', '정규표현식')
> str_extract("홍길동35이순신45강감찬50","[0-9]")
[1] "3"
> # 형식) str_extract('문자열', '정규표현식')
> str_extract("홍길동35이순신45강감찬50","[0-9]{2}") #[0-9]사이의 숫자값, {2}연속 2자리인 것만 추출하겠다.
[1] "35"
> str_extract_all("홍길동35이순신45강감찬50","[0-9]{2}")
[[1]]
[1] "35" "45" "50"

> str_extract_all(string, '[0-9]{3}') # 3자 연속된 알파벳 추출
[[1]]
[1] "105" "100" "200"

> str_extract_all(string, '[a-z]{3}') # 3자 연속된 알파벳 추출
[[1]]
[1] "hon" "gki" "lee" "you"

> str_extract_all(string, '[a-z]{3}') # 3자 연속된 알파벳 추출 # 소문자, 대문자 구분
[[1]]
[1] "hon" "gki" "lee" "you"

> # 반복수를 지정하여 영문자 추출
> string <- 'hongkildong105lee1002you25강감찬2005'
> str_extract_all(string, '[a-z]{3}') # 3자 연속된 알파벳 추출 # 소문자, 대문자 구분
[[1]]
[1] "hon" "gki" "ldo" "lee" "you"

> str_extract_all(string, '[a-z]{3,}') # 3글자 이상 연속된 알파벳 추출
[[1]]
[1] "hongkildong" "lee"         "you"

> str_extract_all(string, '[a-z]{3,5}') # 3~5글자 범위 안에 속하는 알파벳 추출
[[1]]
[1] "hongk" "ildon" "lee"   "you"

> # 특정 단어 추출
> str_extract_all(string, '유관순')
[[1]]
character(0)

> str_extract_all(string, '강감찬')
[[1]]
[1] "강감찬"

> # 반복수를 지정하여 영문자 추출
> string <- 'hongkildong105lee1002you25베드로로2005'
> str_extract_all(string, '베드로')
[[1]]
[1] "베드로"

> str_extract_all(string, '[a-z]{3}') # 3자 연속된 알파벳 추출 # 소문자, 대문자 구분
[[1]]
```

```
[1] "hon" "gki" "ldo" "lee" "you"

> str_extract_all(string, '[a-z]{3,}') # 3글자 이상 연속된 알파벳 추출
[[1]]
[1] "hongkildong" "lee"         "you"

> str_extract_all(string, '[a-z]{3,5}') # 3~5글자 범위 안에 속하는 알파벳 추출
[[1]]
[1] "hongk" "ildon" "lee"   "you"

> # 특정 단어 추출
> str_extract_all(string, '유관순')
[[1]]
character(0)

> str_extract_all(string, '베드로')
[[1]]
[1] "베드로"

> # 한글, 영문자, 숫자 추출하기
> str_extract_all(string, 'hong')
[[1]]
[1] "hong"

> str_extract_all(string, '25')
[[1]]
[1] "25"

> str_extract_all(string, '') # 한글 검색
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "베" "드" "로" "로"
[31] "2"  "0"  "0"  "5"

> str_extract_all(string, '') # 소문자 검색
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "베" "드" "로" "로"
[31] "2"  "0"  "0"  "5"

> str_extract_all(string, '') # 대문자 검색
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "베" "드" "로" "로"
[31] "2"  "0"  "0"  "5"

> str_extract_all(string, '[가-힣]') # 한글 검색
[[1]]
[1] "베" "드" "로" "로"

> str_extract_all(string, '') # 소문자 검색
[[1]]
 [1] "h"  "o"  "n"  "g"  "k"  "i"  "l"  "d"  "o"  "n"  "g"  "1"  "0"  "5"  "l"
[16] "e"  "e"  "1"  "0"  "0"  "2"  "y"  "o"  "u"  "2"  "5"  "베" "드" "로" "로"
[31] "2"  "0"  "0"  "5"

> # 반복수를 지정하여 영문자 추출
> string <- 'hongkildong105lee1002you25베드로2005'
> str_extract_all(string, '[a-z]{3}') # 3자 연속된 알파벳 추출 # 소문자, 대문자 구분
[[1]]
[1] "hon" "gki" "ldo" "lee" "you"

> str_extract_all(string, '[a-z]{3,}') # 3글자 이상 연속된 알파벳 추출
[[1]]
[1] "hongkildong" "lee"         "you"

> str_extract_all(string, '[a-z]{3,5}') # 3~5글자 범위 안에 속하는 알파벳 추출
[[1]]
[1] "hongk" "ildon" "lee"   "you"

> # 특정 단어 추출
> str_extract_all(string, '유관순')
[[1]]
character(0)

> str_extract_all(string, '베드로')
[[1]]
[1] "베드로"

> str_extract_all(string, '[가-힣]{3}') # 한글 검색
[[1]]
[1] "베드로"

> str_extract_all(string, '[a-z]{3}') # 소문자 검색
[[1]]
[1] "hon" "gki" "ldo" "lee" "you"
```

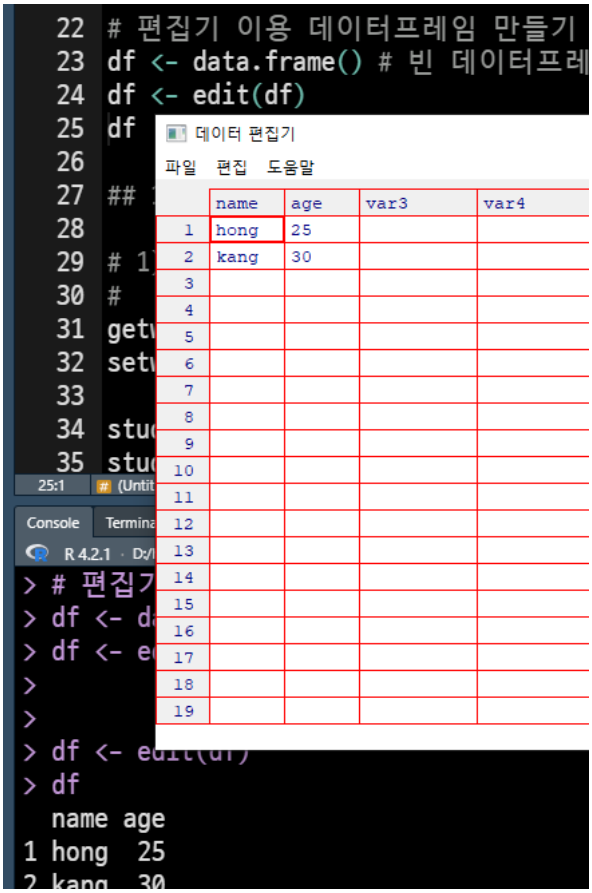
```
> str_extract_all(string, '[A-Z]{3}') # 대문자 검색
[[1]]
character(0)

> str_extract_all(string, '[0-9]{4}') # 숫자 검색
[[1]]
[1] "1002" "2005"

> str_extract_all(string, '[가-힣]{3}') # 한글 검색
[[1]]
[1] "베드로"

> str_extract_all(string, '[a-z]{3}') # 소문자 검색
[[1]]
[1] "hon" "gki" "ldo" "lee" "you"

> str_extract_all(string, '[A-Z]{3}') # 대문자 검색
[[1]]
character(0)

> str_extract_all(string, '[0-9]{4}') # 숫자 검색
[[1]]
[1] "1002" "2005"

> # 한글, 영문자, 숫자를 제외한 나머지 추출하기
> str_extract_all(string, '[^a-z]') # ^표시가 제외외
[[1]]
 [1] "1"  "0"  "5"  "1"  "0"  "0"  "2"  "2"  "5"  "베" "드" "로" "2"  "0"  "0"
[16] "5"

> str_extract_all(string, '[^a-z]{4}')
[[1]]
[1] "1002"   "25베드" "로200"

> str_extract_all(string, '[^a-z]{4}') # 알파벳을 제외한 4개의 연속
[[1]]
[1] "1002"   "25베드" "로200"

> str_extract_all(string, '[^가-힣]')
[[1]]
 [1] "h" "o" "n" "g" "k" "i" "l" "d" "o" "n" "g" "1" "0" "5" "l" "e" "e" "1" "0"
[20] "0" "2" "y" "o" "u" "2" "5" "2" "0" "0" "5"

> str_extract_all(string, '[^가-힣]{5}')
[[1]]
[1] "hongk" "ildon" "g105l" "ee100" "2you2"

> str_extract_all(string, '[^0-9]{3}')
[[1]]
[1] "hon"    "gki"    "ldo"    "lee"    "you"    "베드로"

> # 주민등록번호 검사하기
> jumin <- '123456-3234567'
> str_extract_all(jumin, '')
[[1]]
 [1] "1" "2" "3" "4" "5" "6" "-" "3" "2" "3" "4" "5" "6" "7"

> str_extract_all(jumin, '')
[[1]]
 [1] "1" "2" "3" "4" "5" "6" "-" "3" "2" "3" "4" "5" "6" "7"

> str_extract_all(string, '[^0-9]{3}')
[[1]]
[1] "hon"    "gki"    "ldo"    "lee"    "you"    "베드로"

> # 주민등록번호 검사하기
> jumin <- '123456-3234567'
> str_extract_all(jumin, '')
[[1]]
 [1] "1" "2" "3" "4" "5" "6" "-" "3" "2" "3" "4" "5" "6" "7"

> str_extract_all(jumin, '')
[[1]]
 [1] "1" "2" "3" "4" "5" "6" "-" "3" "2" "3" "4" "5" "6" "7"

> str_extract_all(jumin, '[0-9]{6}')
[[1]]
[1] "123456" "323456"

> str_extract_all(jumin, '[0-9]{7}')
[[1]]
[1] "3234567"

> str_extract_all(jumin, '[0-9]{6}-{1234}')
[[1]]
character(0)
```

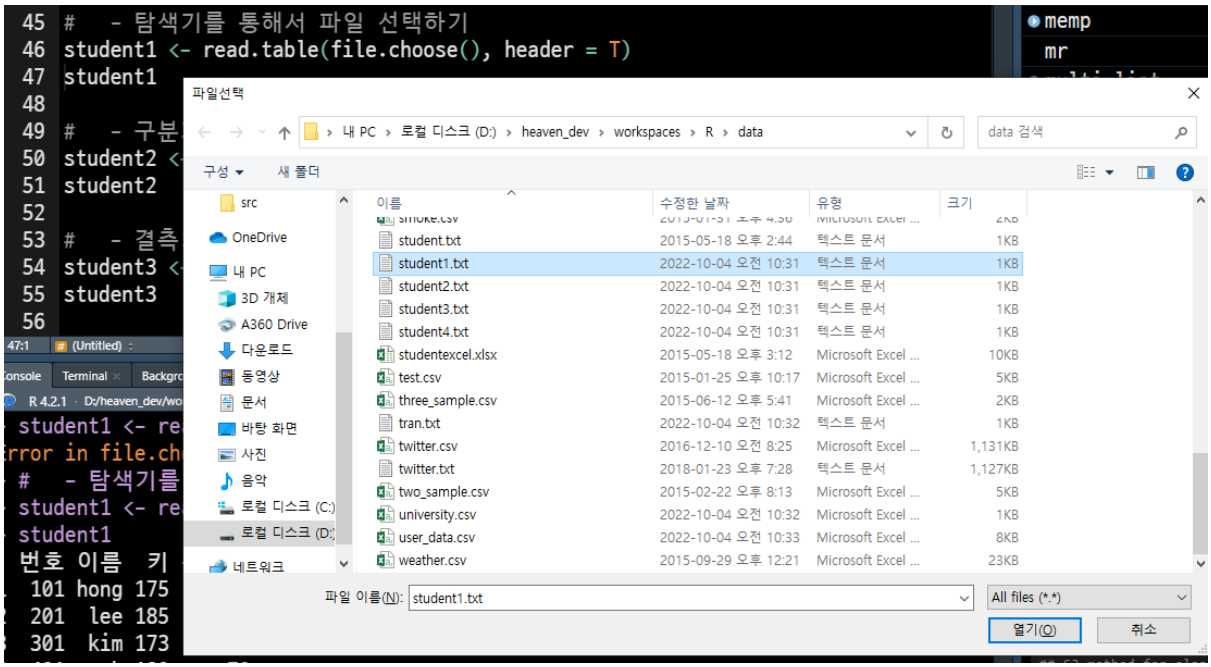
```
> str_extract_all(jumin, '[0-9]{6}-{1234}[0-9]{6}')
[[1]]
character(0)

> # 주민등록번호 검사하기
> jumin <- '123456-3234567'
> str_extract_all(jumin, '[0-9]{6}-{1234}[0-9]{6}')
[[1]]
character(0)

> str_extract_all(jumin, '[0-9]{6}-[1234][0-9]{6}')
[[1]]
[1] "123456-3234567"

> str_extract_all(jumin, '\d')
Error: '\d' is an unrecognized escape in character string starting "'\d"
> str_extract_all(jumin, '[0-9]{6}-[1234]{1}[0-9]{6}')
[[1]]
[1] "123456-3234567"

> str_extract_all(jumin, '\\d{6}-[1234]\\d{6}') # \\두번을 써야지 문자 '\'로 인식
[[1]]
[1] "123456-3234567"

> str_extract_all(jumin, '\d{6}-[1234]\d{6}') # \\두번을 써야지 문자 '\'로 인식
Error: '\d' is an unrecognized escape in character string starting "'\d"
> str_extract_all(jumin, '\\d{6}-[1234]\\d{6}') # \\두번을 써야지 문자 '\'로 인식
[[1]]
[1] "123456-3234567"

> # 지정된 길이의 단어 추출하기
> name <- '홍길동1234,이순신5678,강감찬1012'
> str_extract_all(name, '\\w{7,}') # \w 소문자 word
[[1]]
[1] "홍길동1234" "이순신5678" "강감찬1012"

> str_extract_all(name, '\\w{7,}') # \w 특수문자 제외한 모든 문자
[[1]]
[1] "홍길동1234" "이순신5678" "강감찬1012"

> # '\\w': 한글, 영문자, 숫자는 포함.
> # '\\W': 특수문자 선택.
> str_extract_all(name, '\\W')
[[1]]
[1] "," ","

> # 문자열 위치(index) 구하기
> string <- 'hongkd105leess1002you25강감찬2005'
> str_locate(string, '강감찬')
     start end
[1,]    24  26
> # 문자열 길이 구하기
> string <- 'hongkild105lee1002you25강감찬2005'
> len <- str_length(string) # 30
> len
[1] 30
> str_locate(string, '강감찬')
     start end
[1,]    24  26
> # 문자열 길이 구하기
> string <- 'hongkild105lee1002you25강감찬2005'
> len <- str_length(string) # 30
> len
[1] 30
> # 부분 문자열
> string_sub <- str_sub(string, 1, len-7) # 30-7
> string_sub
[1] "hongkild105lee1002you25"
> # 부분 문자열
> string_sub <- str_sub(string, 1, len-7) # 30-7 = 23개까지의 데이터만을
> string_sub
[1] "hongkild105lee1002you25"
> string_sub <- str_sub(string, 1, 23)
> string_sub
[1] "hongkild105lee1002you25"
> # 대문자, 소문자 변경하기
> str_to_upper(string_sub)
[1] "HONGKILD105LEE1002YOU25"
> str_to_lower(string_sub)
[1] "hongkild105lee1002you25"
> # 문자열 교체하기
> string_rep <- str_replace(string_sub, 'hongkild105', '홍길동35,')
> string_rep <- str_replace(string_rep, 'lee1002', '이순신45,')
> string_rep <- str_replace(string_rep, 'you25', '유관순25,')
> string_rep
[1] "홍길동35,이순신45,유관순25,"
> # 문자열 결합하기
```

```
> string_c <- str_c(string_rep, '강감찬55')
> string_c
[1] "홍길동35,이순신45,유관순25,강감찬55"
> # 문자열 분리하기
> string_sp <- str_split(string_c, ',')
> string_sp
[[1]]
[1] "홍길동35" "이순신45" "유관순25" "강감찬55"

> string_sp
[[1]]
[1] "홍길동35" "이순신45" "유관순25" "강감찬55"

> # 문자열 합치기
> string_vec <- c('홍길동35', '이순신45', '유관순25', '강감찬55')
> string_vec
[1] "홍길동35" "이순신45" "유관순25" "강감찬55"
> string_join <- paste(string_vec, collapse = ',')
> string_join
[1] "홍길동35,이순신45,유관순25,강감찬55"
>
```

[]	문자 클래스	"["과 "]" 사이의 문자 중 하나를 선택한다. "¦"를 여러 개 쓴 것과 같은 의미이다. 예를 들면 [abc]d는 ad, bd, cd를 뜻한다. 또한, "-" 기호와 함께 쓰면 범위를 지정할 수 있다. "[a-z]"는 a부터 z까지 중 하나, "[1-9]"는 1부터 9까지 중의 하나를 의미한다.

JAVA에서는 여러분 "안녕" System.out.println("\"\"); 문자열로 인식(") \", \t, \n, \\ 이러한 것들이 글자 그대로 인식하게 만들어줬음.


22 # 편집기 이용 데이터프레임 만들기
23 df <- data.frame() # 빈 데이터프레
24 df <- edit(df)
25 df
데이터 편집기
파일 편집 도움말
name age var3 var4
1 hong 25
2 kang 30
Console
R 4.2.1
> df <- edit(df)
> df
name age
1 hong 25
2 kang 30


45 # - 탐색기를 통해서 파일 선택하기
46 student1 <- read.table(file.choose(), header = T)
47 student1
파일선택
내 PC › 로컬 디스크 (D:) › heaven_dev › workspaces › R › data
data 검색
구성
새 폴더
src
OneDrive
내 PC
3D 개체
A360 Drive
다운로드
동영상
문서
바탕 화면
사진
음악
로컬 디스크 (C:)
로컬 디스크 (D:)
네트워크
이름
수정한 날짜
유형
크기
student.txt 2015-05-18 오후 2:44 텍스트 문서 1KB
student1.txt 2022-10-04 오전 10:31 텍스트 문서 1KB
student2.txt 2022-10-04 오전 10:31 텍스트 문서 1KB
student3.txt 2022-10-04 오전 10:31 텍스트 문서 1KB
student4.txt 2022-10-04 오전 10:31 텍스트 문서 1KB
studentexcel.xlsx 2015-05-18 오후 3:12 Microsoft Excel ... 10KB
test.csv 2015-01-25 오후 10:17 Microsoft Excel ... 5KB
three_sample.csv 2015-06-12 오후 5:41 Microsoft Excel ... 2KB
tran.txt 2022-10-04 오전 10:32 텍스트 문서 1KB
twitter.csv 2016-12-10 오전 8:25 Microsoft Excel ... 1,131KB
twitter.txt 2018-01-23 오후 7:28 텍스트 문서 1,127KB
two_sample.csv 2015-02-22 오후 8:13 Microsoft Excel ... 5KB
university.csv 2022-10-04 오전 10:32 Microsoft Excel ... 1KB
user_data.csv 2022-10-04 오전 10:33 Microsoft Excel ... 8KB
weather.csv 2015-09-29 오후 12:21 Microsoft Excel ... 23KB
파일 이름(N): student1.txt
All files (*.*)
열기(O)
취소
memp
mr
student1 <- re
# - 탐색기를
student1 <- re
student1
번호 이름 키
101 hong 175
201 lee 185
301 kim 173

```
# chap03_DataIO

#####################################
# chapter03. 데이터 입출력
#####################################

# 1. 데이터 불러오기

## 1-1. 키보드 입력

# 키보드로 숫자 입력하기
num <- scan()
num

# 합계 구하기
sum(num)

# 키보드로 문자 입력하기
name <- scan(what = character())
name

# 편집기 이용 데이터프레임 만들기
df <- data.frame() # 빈 데이터프레임 생성성
df <- edit(df)
df

## 1-2. 로컬 파일 가져오기

# 1) read.table() 함수 이용
#   - 컬럼명이 없는 파일 불러오기
getwd()
setwd("D:/heaven_dev/workspaces/R/data")

student <- read.table(file = "student.txt")
student
mode(student); class(student)

names(student) <- c('번호', '이름', '키', '몸무게')
student

#   - 컬럼명이 있는 파일 불러오기
student1 <- read.table(file = "student1.txt", header = T )
student1

#   - 탐색기를 통해서 파일 선택하기
student1 <- read.table(file.choose(), header = T)
student1

#   - 구분자가 있는 경우(세미콜론, 탭)
student2 <- read.table(file = "student2.txt", sep = ";", header = T)
student2

#   - 결측치를 처리하여 파일 불러오기
student3 <- read.table(file = "student3.txt", header = T, na.strings = "-")
student3

#   - csv 파일 형식 불러오기
student4 <- read.csv()
student4

# read.xlsx() 함수 이용 - 엑셀데이터 읽어오기
# 패키지 설치와 java 실행 환경 설정
install.packages("rJava") # rJava 패키지 설치
install.packages("xlsx")  # xlsx 패키지 설치
Sys.setenv()

# 관련 패키지 메모리 로드
library(rJava)
library(xlsx)

# 엑셀 파일 가져오기
studentex <- read.xlsx()
studentex


## 1-3. 인터넷에서 파일 가져오기

# 단계1 : 세계 GDP 순위 데이터 가져오기
GDP_ranking <- read.csv("https://databank.worldbank.org/data/download/GDP.csv", encoding = "UTF-8")
GDP_ranking
head(GDP_ranking, 20)
```

```
dim(GDP_ranking)

# 데이터를 가공하기 위해 불필요한 행과 열을 제거한다.
GDP_ranking2 <- GDP_ranking[-c(1:4), c(1,2,4,5)]
head(GDP_ranking2)

# 상위 16개 국가 선별한다.
GDP_ranking16 <- head(GDP_ranking2, 16) # 상위 16개 국가
GDP_ranking16

# 데이터프레임을 구성하는 4개의 열에 대한 이름을 지정한다.
names(GDP_ranking16) <- c('Code', 'Ranking', 'Nation', 'GDP')
GDP_ranking16
dim(GDP_ranking16)

# 단계2 : 세계 GDP 상위 16위 국가 막대 차트 시각화
gdp <- GDP_ranking16$GDP
nation <- GDP_ranking16$Nation
gdp

num_gdp <- as.numeric(str_replace_all(gdp, ',', ''))
num_gdp

GDP_ranking16$GDP <- num_gdp


# 막대차트 시각화
barplot(GDP_ranking16$GDP, col = rainbow(16),
        xlab = '국가(nation)', ylab='단위(달러)', names.arg=nation)

# 1,000단위 축소
num_gdp2 <- num_gdp / 1000
GDP_ranking16$GDP2 <- num_gdp2
barplot(GDP_ranking16$GDP2, col = rainbow(16),
        main = "2020년도 GDP 세계 16위 국가",
        xlab = '국가(nation)', ylab='단위(천달러)', names.arg=nation)

GDP_ranking16


## 1-4. 웹문서 가져오기

# 2010년 ~ 2015년도 미국의 주별 1인당 소득 자료 가져오기.
# "https://ssti.org/blog/useful-stats-capita-personal-income-state-2010-2015"

# 단계1 : XML / httr 패키지 설치
install.packages("XML")
install.packages("httr")

library(XML)
library(httr)

# 단계2 : 미국의 주별 1인당 소득 자료 가져오기.
url <- "https://ssti.org/blog/useful-stats-capita-personal-income-state-2010-2015"




# 단계4 : 컬럼명을 수정한 후 뒷부분 6개 관측치 보기



# 2. 데이터 저장하기
# 2-1. 화면(콘솔) 출력
#  1) cat() 함수



#  2) print() 함수




# 2-2. 파일에 데이터 저장
#  1) sink() 함수를 이용한 파일 저장
getwd()
setwd("C:/workspaces/Rwork/output")

library(RSADBE)
data("Severity_Counts") # Severity_Counts 데이터 셋 가져오기
Severity_Counts

sink("severity.txt") # 저장할 파일 open
severity <- Severity_Counts # 데이터 셋을 변수에 저장.
severity  # 콘솔에 출력되지 않고, 파일에 저장
sink()    # 오픈된 파일 close
```

```
#  2) write.table() 함수 이용 파일 저장
# 탐색기를 이용하여 데이터 가져오기


# 기본 속성으로 저장 - 행이름과 따옴표가 붙는다.


# 'row.names=F' 속성을 이용하여 행이름 제거하여 저장한다.


# 'quote=F' 속성을 이용하여 따옴표를 제거하여 저장한다.


# 행이름 제거 + 따옴표 제거




#  3) write.xlsx() 함수 이용 파일 저장 - 엑셀 파일로 저장
library(rJava)
library(xlsx) # excel data 입출력 함수 제공



#  4) write.csv() 함수 이용 파일 저장
#     - data.frame 형식의 데이터를 csv 형식으로 저장.
```

```
> # 키보드로 숫자 입력하기
> num <- scan()
1: num
1: 10 20 30
Error in scan() : scan() expected 'a real', got 'num'
>
> # 키보드로 숫자 입력하기
> num <- scan()
1: 10 20 30
4:
Read 3 items
> num
[1] 10 20 30
> # 합계 구하기
> sum(num)
[1] 60
> # 키보드로 문자 입력하기
> name <- scan(what = character())
1: 홍 강 특
4:
Read 3 items
> name
[1] "홍" "강" "특"
> # 편집기 이용 데이터프레임 만들기
> df <- data.frame() # 빈 데이터프레임 생성성
> df <- edit(df)
> df
  var1 var2
1  asd asdf
2 asdf asdf
> # 1) read.table() 함수 이용
> #   - 컬럼명이 없는 파일 불러오기
> getwd()
[1] "D:/heaven_dev/workspaces/R/data"
> setwd("D:/heaven_dev/workspaces/R/data")
> student <- read.table(file = "student.txt")
> student
   V1   V2  V3 V4
1 101 hong 175 65
2 201  lee 185 85
3 301  kim 173 60
4 401 park 180 70
> mode(student); class(student)
[1] "list"
[1] "data.frame"
> names(student) <- c('번호', '이름', '키', '몸무게')
> student
  번호 이름  키 몸무게
1  101 hong 175     65
2  201  lee 185     85
3  301  kim 173     60
4  401 park 180     70
> #   - 컬럼명이 있는 파일 불러오기
> student1 <- read.table(file = "student1.txt", header = T )
```

```
> student1
  번호 이름  키 몸무게
1  101 hong 175     65
2  201  lee 185     85
3  301  kim 173     60
4  401 park 180     70
> #   - 탐색기를 통해서 파일 선택하기
> student1 <- read.table(file.choose(), header = T)
> student1
  번호 이름  키 몸무게
1  101 hong 175     65
2  201  lee 185     85
3  301  kim 173     60
4  401 park 180     70
> #   - 구분자가 있는 경우(세미콜론, 탭)
> student2 <- read.table(file = "student2.txt", sep = ";", header = T)
> student2
  번호 이름  키 몸무게
1  101 hong 175     65
2  201  lee 185     85
3  301  kim 173     60
4  401 park 180     70
> #   - 결측치를 처리하여 파일 불러오기
> student3 <- read.table(file = "student3.txt", header = T, na.strings = "-")
> student3
  번호 이름  키 몸무게
1  101 hong 175     65
2  201  lee 185     85
3  301  kim 173     NA
4  401 park  NA     70
>
```