PR5\_201823869\_조성우

조성우

2020년 4월 17일

# Dataframe

## 1.벡터를 이용해 데이터프레임 만들기

name <- c("boil","Tom","Ravindra","Bob","Sobia")  
gender <- c('M','M','F','M','F')  
age <- c(17,21,33,12,37)  
marriage <- c(F,T,F,F,T)  
  
#stringsAsFactors 인수 없이 만들기  
customer <- data.frame(name,gender,age,marriage)  
str(customer)

## 'data.frame': 5 obs. of 4 variables:  
## $ name : Factor w/ 5 levels "Bob","boil","Ravindra",..: 2 5 3 1 4  
## $ gender : Factor w/ 2 levels "F","M": 2 2 1 2 1  
## $ age : num 17 21 33 12 37  
## $ marriage: logi FALSE TRUE FALSE FALSE TRUE

#stringsAsFactors=F 사용해서 만들기  
customer=data.frame(name,gender,age,marriage,stringsAsFactors = F)  
str(customer)

## 'data.frame': 5 obs. of 4 variables:  
## $ name : chr "boil" "Tom" "Ravindra" "Bob" ...  
## $ gender : chr "M" "M" "F" "M" ...  
## $ age : num 17 21 33 12 37  
## $ marriage: logi FALSE TRUE FALSE FALSE TRUE

#data.frame 함수와 관련된 다양한 함수 사용하기  
str(customer) #데이터프레임의 구조 확인

## 'data.frame': 5 obs. of 4 variables:  
## $ name : chr "boil" "Tom" "Ravindra" "Bob" ...  
## $ gender : chr "M" "M" "F" "M" ...  
## $ age : num 17 21 33 12 37  
## $ marriage: logi FALSE TRUE FALSE FALSE TRUE

names(customer) #데이터프레임의 열이름을 확인

## [1] "name" "gender" "age" "marriage"

rownames(customer) #데이터프레임의 행 이름을 확인

## [1] "1" "2" "3" "4" "5"

## 2. Data Frame 변수명 바꾸기

# colnames,rownames 함수로 변수명 반환 및 확인  
  
colnames(customer)

## [1] "name" "gender" "age" "marriage"

rownames(customer)

## [1] "1" "2" "3" "4" "5"

colnames(customer) <- c("cust\_name","cust\_gend","cust\_age","cust\_mrg")  
rownames(customer) <- c('a','b','c','d','e')  
customer

## cust\_name cust\_gend cust\_age cust\_mrg  
## a boil M 17 FALSE  
## b Tom M 21 TRUE  
## c Ravindra F 33 FALSE  
## d Bob M 12 FALSE  
## e Sobia F 37 TRUE

## Data Frame 데이터 추출

#접근 방식은 matrix와 동일  
#[행,열]연산자 및 $연산자 활용하여 데이터에 접근하기  
customer[1,] ; customer["a",] #첫번째 행 숫자 및 rowname 으로 추출

## cust\_name cust\_gend cust\_age cust\_mrg  
## a boil M 17 FALSE

## cust\_name cust\_gend cust\_age cust\_mrg  
## a boil M 17 FALSE

customer[customer$cust\_name=="Tom",] #cust\_name 컬럼이 Tom인 row만 추출

## cust\_name cust\_gend cust\_age cust\_mrg  
## b Tom M 21 TRUE

customer[2:5,] ; customer[-1,] #2~5 행

## cust\_name cust\_gend cust\_age cust\_mrg  
## b Tom M 21 TRUE  
## c Ravindra F 33 FALSE  
## d Bob M 12 FALSE  
## e Sobia F 37 TRUE

## cust\_name cust\_gend cust\_age cust\_mrg  
## b Tom M 21 TRUE  
## c Ravindra F 33 FALSE  
## d Bob M 12 FALSE  
## e Sobia F 37 TRUE

customer[customer$cust\_name!="Tom",] #cust\_name이 컬럼이 Tom이 아닌 row

## cust\_name cust\_gend cust\_age cust\_mrg  
## a boil M 17 FALSE  
## c Ravindra F 33 FALSE  
## d Bob M 12 FALSE  
## e Sobia F 37 TRUE

customer[c("b","c"),]

## cust\_name cust\_gend cust\_age cust\_mrg  
## b Tom M 21 TRUE  
## c Ravindra F 33 FALSE

## 4. Data Frame 에 데이터추가

#이름으로 추가,  
customer$cust\_height <- c("185","165","156","174","155")  
customer["f",] <- list("Jack","M",50,T,"167")  
customer

## cust\_name cust\_gend cust\_age cust\_mrg cust\_height  
## a boil M 17 FALSE 185  
## b Tom M 21 TRUE 165  
## c Ravindra F 33 FALSE 156  
## d Bob M 12 FALSE 174  
## e Sobia F 37 TRUE 155  
## f Jack M 50 TRUE 167

#cbind, rbind 로 추가   
customer <- cbind(customer,weight = c(80,70,65,48,55,100))  
customer <- rbind(customer,g=list("Merry","F",42,F,"172",60))  
customer <- rbind(customer,h=c("Merry",F,42,F,"172",60))  
customer

## cust\_name cust\_gend cust\_age cust\_mrg cust\_height weight  
## a boil M 17 FALSE 185 80  
## b Tom M 21 TRUE 165 70  
## c Ravindra F 33 FALSE 156 65  
## d Bob M 12 FALSE 174 48  
## e Sobia F 37 TRUE 155 55  
## f Jack M 50 TRUE 167 100  
## g Merry F 42 FALSE 172 60  
## h Merry FALSE 42 FALSE 172 60

## 5. Data Frame 에 데이터삭제

customer <- customer[,-5] #1 번째 컬럼을 빼고 나머지만 다시 할당  
customer <- customer[-7,] #7 번째 로우를 빼고 나머지만 다시 할당  
customer$weight <- NULL #weight 컬럼 삭제

## 6. Data 조건문을 활용해 조작하기

# 이부분은 모든 코드에 주석달것!  
# &와 |연산자로 여러개의 조건을 사용할 수 있음  
  
customer[customer$cust\_gend=="M",]

## cust\_name cust\_gend cust\_age cust\_mrg  
## a boil M 17 FALSE  
## b Tom M 21 TRUE  
## d Bob M 12 FALSE  
## f Jack M 50 TRUE

customer[customer$cust\_gend!="F",]

## cust\_name cust\_gend cust\_age cust\_mrg  
## a boil M 17 FALSE  
## b Tom M 21 TRUE  
## d Bob M 12 FALSE  
## f Jack M 50 TRUE  
## h Merry FALSE 42 FALSE

nrow(customer[customer$cust\_gend=="m"]) #nrow는 행의 개수를 보여줌

## [1] 7

customer[customer$cust\_name =="Bob", c("cust\_age","cust\_mrg")]

## cust\_age cust\_mrg  
## d 12 FALSE

customer[customer$cust\_name =="Tom" | customer$cust\_name=="Ravindra",]

## cust\_name cust\_gend cust\_age cust\_mrg  
## b Tom M 21 TRUE  
## c Ravindra F 33 FALSE

customer[customer$cust\_gend=="M" & customer$cust\_age>24, ]

## cust\_name cust\_gend cust\_age cust\_mrg  
## f Jack M 50 TRUE

## 7. Data frame 정렬하기

#order 함수를 활용해 순서를 구하여, row 조건에 넣어서 정렬  
#decreasing=T 인수를 활용하여 오름차순, 내림차순 변경 가능  
  
order(customer$cust\_age) #order 함수로 age에 대한 순서를 구함

## [1] 4 1 2 3 5 7 6

customer[order(customer$cust\_age),] #row의 조건에 위에서 구한 순서를 넣음

## cust\_name cust\_gend cust\_age cust\_mrg  
## d Bob M 12 FALSE  
## a boil M 17 FALSE  
## b Tom M 21 TRUE  
## c Ravindra F 33 FALSE  
## e Sobia F 37 TRUE  
## h Merry FALSE 42 FALSE  
## f Jack M 50 TRUE

q

## function (save = "default", status = 0, runLast = TRUE)   
## .Internal(quit(save, status, runLast))  
## <bytecode: 0x00000000139d2e48>  
## <environment: namespace:base>

order(customer$cust\_age, decreasing=F) #오름차순

## [1] 4 1 2 3 5 7 6

customer[order(customer$cust\_age, decreasing=F),]

## cust\_name cust\_gend cust\_age cust\_mrg  
## d Bob M 12 FALSE  
## a boil M 17 FALSE  
## b Tom M 21 TRUE  
## c Ravindra F 33 FALSE  
## e Sobia F 37 TRUE  
## h Merry FALSE 42 FALSE  
## f Jack M 50 TRUE

## Data frame 기타 함수

#head, tail 함수는 데이터프레임이 상위,하위 row를 출력함  
# 기본 6개를 출력하며, row 수를 지정할수 있음  
  
head(customer) #상우 6개 row

## cust\_name cust\_gend cust\_age cust\_mrg  
## a boil M 17 FALSE  
## b Tom M 21 TRUE  
## c Ravindra F 33 FALSE  
## d Bob M 12 FALSE  
## e Sobia F 37 TRUE  
## f Jack M 50 TRUE

head(customer,2) #상위 2개 row

## cust\_name cust\_gend cust\_age cust\_mrg  
## a boil M 17 FALSE  
## b Tom M 21 TRUE

tail(customer,2) #하위 2개 row

## cust\_name cust\_gend cust\_age cust\_mrg  
## f Jack M 50 TRUE  
## h Merry FALSE 42 FALSE

# 파일 입출력

## 1. 내장데이터 불러오기

#MASS 패키지에는 다양한 데이터가 들어있음  
  
#install.packages("MASS")  
library(MASS)  
#Iris 데이터셋  
#붓꽃의 종과 sepal 과 petal 의 너비와 길이에 대한 데이터  
head(iris)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
## 1 5.1 3.5 1.4 0.2 setosa  
## 2 4.9 3.0 1.4 0.2 setosa  
## 3 4.7 3.2 1.3 0.2 setosa  
## 4 4.6 3.1 1.5 0.2 setosa  
## 5 5.0 3.6 1.4 0.2 setosa  
## 6 5.4 3.9 1.7 0.4 setosa

str(iris)

## 'data.frame': 150 obs. of 5 variables:  
## $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...  
## $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...  
## $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...  
## $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...  
## $ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

# mtcars 데이터셋  
# 자동차 차종별 상세스펙에 대한 데이터  
head(mtcars)

## mpg cyl disp hp drat wt qsec vs am gear carb  
## Mazda RX4 21.0 6 160 110 3.90 2.620 16.46 0 1 4 4  
## Mazda RX4 Wag 21.0 6 160 110 3.90 2.875 17.02 0 1 4 4  
## Datsun 710 22.8 4 108 93 3.85 2.320 18.61 1 1 4 1  
## Hornet 4 Drive 21.4 6 258 110 3.08 3.215 19.44 1 0 3 1  
## Hornet Sportabout 18.7 8 360 175 3.15 3.440 17.02 0 0 3 2  
## Valiant 18.1 6 225 105 2.76 3.460 20.22 1 0 3 1

str(mtcars)

## 'data.frame': 32 obs. of 11 variables:  
## $ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...  
## $ cyl : num 6 6 4 6 8 6 8 4 4 6 ...  
## $ disp: num 160 160 108 258 360 ...  
## $ hp : num 110 110 93 110 175 105 245 62 95 123 ...  
## $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...  
## $ wt : num 2.62 2.88 2.32 3.21 3.44 ...  
## $ qsec: num 16.5 17 18.6 19.4 17 ...  
## $ vs : num 0 0 1 1 0 1 0 1 1 1 ...  
## $ am : num 1 1 1 0 0 0 0 0 0 0 ...  
## $ gear: num 4 4 4 3 3 3 3 4 4 4 ...  
## $ carb: num 4 4 1 1 2 1 4 2 2 4 ...

# USArrests 데이터셋  
# 1973년도 50 개 주에서 수집된 범죄기록 데이터  
head(USArrests)

## Murder Assault UrbanPop Rape  
## Alabama 13.2 236 58 21.2  
## Alaska 10.0 263 48 44.5  
## Arizona 8.1 294 80 31.0  
## Arkansas 8.8 190 50 19.5  
## California 9.0 276 91 40.6  
## Colorado 7.9 204 78 38.7

str(USArrests)

## 'data.frame': 50 obs. of 4 variables:  
## $ Murder : num 13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...  
## $ Assault : int 236 263 294 190 276 204 110 238 335 211 ...  
## $ UrbanPop: int 58 48 80 50 91 78 77 72 80 60 ...  
## $ Rape : num 21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...

## 2. file로 저장된 데이터 불러오기

getwd()

## [1] "C:/Users/JSW/Desktop/강의자료/R프로그래밍/R 실습 및 과제/과제"

# read.csv()함수  
# header = T (첫행 컬럼명으로 사용)  
# row.names = 1 (첫열 로우명으로 사용)  
# sep = "," (입력된 데이터를 구분해주는 기호)  
# na.strings = c("Na","nan") (NA값으로 처리할 문자열 정의)  
# fileEncoding = "UTF-8" (문자열을 특정 형식으로 재인코딩)  
# encoding = "UTF-8" (불러들일 file의 인코딩을 미리 선언)  
# stringsAsFactors = F  
  
#그냥 읽어오기  
csv <- read.csv("read\_csv.csv") ; csv

## X..연습.테이블.입니다. X X.1 X.2  
## 1 1 Daredevil Hawkeye Loki  
## 2 2 Deadpool Hulk Luke Cage  
## 3 3 Doctor Strange Human Torch .  
## 4 6 Invisible Woman Ms. Marvel  
## 5 5 Iron Man Nightcrawler  
## 6 7 Ghost Rider Jean Grey Psylocke  
## X.3 X.4  
## 1 Punisher Storm  
## 2 Rocket Raccoon Taskmaster  
## 3 Scarlet Witch Thing  
## 4 Silver Surfer Thor  
## 5 N.A. Wolverine  
## 6 Squirrel Girl Barricade

str(csv)

## 'data.frame': 6 obs. of 6 variables:  
## $ X..연습.테이블.입니다.: int 1 2 3 6 5 7  
## $ X : Factor w/ 5 levels "","Daredevil",..: 2 3 4 1 1 5  
## $ X.1 : Factor w/ 6 levels "Hawkeye","Hulk",..: 1 2 3 4 5 6  
## $ X.2 : Factor w/ 6 levels ".","Loki","Luke Cage",..: 2 3 1 4 5 6  
## $ X.3 : Factor w/ 6 levels "N.A.","Punisher",..: 2 3 4 5 1 6  
## $ X.4 : Factor w/ 6 levels "Barricade","Storm",..: 2 3 4 5 6 1

getwd()

## [1] "C:/Users/JSW/Desktop/강의자료/R프로그래밍/R 실습 및 과제/과제"

# header, stringsAsFactors 사용  
# 불러온 데이터가 어떻게 바뀌는지 확인해보세요  
csv2 <- read.csv("read\_csv.csv", header=F,stringsAsFactors =F) ; csv2

## V1 V2 V3 V4  
## 1 # 연습 테이블 입니다.   
## 2 1 Daredevil Hawkeye Loki  
## 3 2 Deadpool Hulk Luke Cage  
## 4 3 Doctor Strange Human Torch .  
## 5 6 Invisible Woman Ms. Marvel  
## 6 5 Iron Man Nightcrawler  
## 7 7 Ghost Rider Jean Grey Psylocke  
## V5 V6  
## 1   
## 2 Punisher Storm  
## 3 Rocket Raccoon Taskmaster  
## 4 Scarlet Witch Thing  
## 5 Silver Surfer Thor  
## 6 N.A. Wolverine  
## 7 Squirrel Girl Barricade

str(csv2)

## 'data.frame': 7 obs. of 6 variables:  
## $ V1: chr "# 연습 테이블 입니다." "1" "2" "3" ...  
## $ V2: chr "" "Daredevil" "Deadpool" "Doctor Strange" ...  
## $ V3: chr "" "Hawkeye" "Hulk" "Human Torch" ...  
## $ V4: chr "" "Loki" "Luke Cage" "." ...  
## $ V5: chr "" "Punisher" "Rocket Raccoon" "Scarlet Witch" ...  
## $ V6: chr "" "Storm" "Taskmaster" "Thing" ...

#결측값 처리하기  
#(".","N.A.","") 3 가지문자를 모두 NA 로 인식하도록 함  
csv3 <- read.csv("csv\_NA.csv", header=F, stringsAsFactors=F,na.strings=c(".","N.A.","")) ; csv3

## V1 V2  
## 1 癤\xbf#\xec뿰\xec뒿 \xed뀒\xec씠釉\x94 \xec엯\xeb땲\xeb떎. <NA>  
## 2 1 Daredevil  
## 3 2 Deadpool  
## 4 3 Doctor Strange  
## 5 6 <NA>  
## 6 5 <NA>  
## 7 7 Ghost Rider  
## V3 V4 V5 V6  
## 1 <NA> <NA> <NA> <NA>  
## 2 Hawkeye Loki Punisher Storm  
## 3 Hulk Luke Cage Rocket Raccoon Taskmaster  
## 4 Human Torch <NA> Scarlet Witch Thing  
## 5 Invisible Woman Ms. Marvel Silver Surfer Thor  
## 6 Iron Man Nightcrawler <NA> Wolverine  
## 7 Jean Grey Psylocke Squirrel Girl Barricade

str(csv3)

## 'data.frame': 7 obs. of 6 variables:  
## $ V1: chr "癤\xbf#\xec뿰\xec뒿 \xed뀒\xec씠釉\x94 \xec엯\xeb땲\xeb떎." "1" "2" "3" ...  
## $ V2: chr NA "Daredevil" "Deadpool" "Doctor Strange" ...  
## $ V3: chr NA "Hawkeye" "Hulk" "Human Torch" ...  
## $ V4: chr NA "Loki" "Luke Cage" NA ...  
## $ V5: chr NA "Punisher" "Rocket Raccoon" "Scarlet Witch" ...  
## $ V6: chr NA "Storm" "Taskmaster" "Thing" ...

#인코딩 문제 해결하기  
#불러올 파일의 인코딩을 UTF-8로 지정  
csv4 <- read.csv("csv\_NA.csv", header=F,stringsAsFactors=F,  
 encoding="UTF-8") ; csv4

## V1 V2 V3 V4  
## 1 <U+FEFF>#연습 테이블 입니다.   
## 2 1 Daredevil Hawkeye Loki  
## 3 2 Deadpool Hulk Luke Cage  
## 4 3 Doctor Strange Human Torch .  
## 5 6 Invisible Woman Ms. Marvel  
## 6 5 Iron Man Nightcrawler  
## 7 7 Ghost Rider Jean Grey Psylocke  
## V5 V6  
## 1   
## 2 Punisher Storm  
## 3 Rocket Raccoon Taskmaster  
## 4 Scarlet Witch Thing  
## 5 Silver Surfer Thor  
## 6 N.A. Wolverine  
## 7 Squirrel Girl Barricade

str(csv4)

## 'data.frame': 7 obs. of 6 variables:  
## $ V1: chr "<U+FEFF>#연습 테이블 입니다." "1" "2" "3" ...  
## $ V2: chr "" "Daredevil" "Deadpool" "Doctor Strange" ...  
## $ V3: chr "" "Hawkeye" "Hulk" "Human Torch" ...  
## $ V4: chr "" "Loki" "Luke Cage" "." ...  
## $ V5: chr "" "Punisher" "Rocket Raccoon" "Scarlet Witch" ...  
## $ V6: chr "" "Storm" "Taskmaster" "Thing" ...

# read.table()함수  
# table 형태로 저장된 2차원의 데이터를 불러옴  
# txt 파일이나 csv 파일을 볼러올수 있음  
#불러온 데이터는 데이터프레임으로 생성  
#read.csv()함수와 동일하게 인수를 사용  
table <- read.table("read\_csv.csv", header=F,sep=",",stringsAsFactors=F)  
table

## V1 V2 V3 V4 V5 V6  
## 1 1 Daredevil Hawkeye Loki Punisher Storm  
## 2 2 Deadpool Hulk Luke Cage Rocket Raccoon Taskmaster  
## 3 3 Doctor Strange Human Torch . Scarlet Witch Thing  
## 4 6 Invisible Woman Ms. Marvel Silver Surfer Thor  
## 5 5 Iron Man Nightcrawler N.A. Wolverine  
## 6 7 Ghost Rider Jean Grey Psylocke Squirrel Girl Barricade

## 3. 웹에 있는 표를 읽어오기 readHTMLTable()

#install.packages(c("XML","httr"))  
library(XML)  
library(httr)

## Warning: package 'httr' was built under R version 3.6.3

url <- "http://www.worldometers.info/world-population/"  
  
html\_source <- GET(url) #html 전체 소스를 받아옴  
tabs <- readHTMLTable(rawToChar(html\_source$content), stringsAsFactors =F)  
  
world\_pop <- tabs$popbycountry # 추출된 테이블 들 중에서 원하는 테이블 선택 및 저장  
head(world\_pop)

## # Country (or dependency) Population(2020) YearlyChange NetChange  
## 1 1 China 1,439,323,776 0.39 % 5,540,090  
## 2 2 India 1,380,004,385 0.99 % 13,586,631  
## 3 3 United States 331,002,651 0.59 % 1,937,734  
## 4 4 Indonesia 273,523,615 1.07 % 2,898,047  
## 5 5 Pakistan 220,892,340 2 % 4,327,022  
## 6 6 Brazil 212,559,417 0.72 % 1,509,890  
## Density (P/Km짼) Land Area (Km짼) Migrants(net) Fert.Rate Med.Age UrbanPop %  
## 1 153 9,388,211 -348,399 1.69 38 60.8 %  
## 2 464 2,973,190 -532,687 2.2402 28 35 %  
## 3 36 9,147,420 954,806 1.7764 38 82.8 %  
## 4 151 1,811,570 -98,955 2.3195 30 56.4 %  
## 5 287 770,880 -233,379 3.55 23 35.1 %  
## 6 25 8,358,140 21,200 1.74 33 87.6 %  
## WorldShare  
## 1 18.5 %  
## 2 17.7 %  
## 3 4.2 %  
## 4 3.5 %  
## 5 2.8 %  
## 6 2.7 %

## 4.데이터 저장하기

# write.table 또는 write.csv 함수 사용  
# row.names=F 는 ,해당 인수를 T로 줄 경우 행 이름이 첫 열로 이동하여 저장되기 때문  
table

## V1 V2 V3 V4 V5 V6  
## 1 1 Daredevil Hawkeye Loki Punisher Storm  
## 2 2 Deadpool Hulk Luke Cage Rocket Raccoon Taskmaster  
## 3 3 Doctor Strange Human Torch . Scarlet Witch Thing  
## 4 6 Invisible Woman Ms. Marvel Silver Surfer Thor  
## 5 5 Iron Man Nightcrawler N.A. Wolverine  
## 6 7 Ghost Rider Jean Grey Psylocke Squirrel Girl Barricade

write.table(table,"PR\_table.csv")  
write.table(table,"PR\_table1.csv",row.names=F)  
write.csv(table,"PR\_table2.csv",row.names=F)

# PR5 연습문제

###### 다음은 전세계 covid-19 확진자에 대한 정보를 제공하는 웹사이트입니다.

###### <https://www.worldometers.info/coronavirus/>

###### 각 열의 데이터들의 의미이며 다음에 제시된 열들만 사용하도록 하겠습니다.

###### 1열 : 국가명

###### 2열 : 총 확진자 수

###### 3열 : 추가 확진자 수

###### 4열 : 총 사망자 수

###### 5열 : 추가 사망자 수

###### 6열 : 총 완치자 수

###### 9열 : 인구 100만명 당 확진자 수

###### 13열: 대륙명

## 문제 1

###### 해당 웹사이트에 있는 COVID-19에 대한 테이블을 읽어오고, 하루 전의 데이터를 covid\_yesterday에 할당하세요.

library(XML)  
library(httr)  
url <- "https://www.worldometers.info/coronavirus/"  
  
html\_source <- GET(url) #html 전체 소스를 받아옴  
tabs <- readHTMLTable(rawToChar(html\_source$content), stringsAsFactors = F)  
covid\_yesterday <- tabs$main\_table\_countries\_yesterday   
  
head(covid\_yesterday,20)

## Country,Other TotalCases NewCases TotalDeaths NewDeaths TotalRecovered  
## 1 Asia 358,184 +12,188 14,066 +1,640 171,127  
## 2 North America 759,755 +35,195 39,422 +2,711 74,274  
## 3 Europe 1,029,214 +32,674 96,228 +4,004 289,063  
## 4 South America 72,531 +4,940 3,318 +261 26,856  
## 5 Oceania 8,039 +73 76 +4 4,651  
## 6 Africa 20,419 +1,426 1,020 +52 4,958  
## 7 721 15 644  
## 8 World 2,248,863 +86,496 154,145 +8,672 571,573  
## 9 China 82,692 +325 4,632 +1,290 77,994  
## 10 USA 709,735 +32,165 37,154 +2,535 60,510  
## 11 Spain 190,839 +5,891 20,002 +687 74,797  
## 12 Italy 172,434 +3,493 22,745 +575 42,727  
## 13 France 147,969 +1,909 18,681 +761 34,420  
## 14 Germany 141,397 +3,699 4,352 +300 83,114  
## 15 UK 108,692 +5,599 14,576 +847 N/A  
## 16 Iran 79,494 +1,499 4,958 +89 54,064  
## 17 Turkey 78,546 +4,353 1,769 +126 8,631  
## 18 Belgium 36,138 +1,329 5,163 +306 7,961  
## 19 Brazil 33,682 +2,999 2,141 +194 14,026  
## 20 Russia 32,008 +4,070 273 +41 2,590  
## ActiveCases Serious,Critical Tot혻Cases/1M pop Deaths/1M pop TotalTests  
## 1 172,991 6,403   
## 2 646,059 14,626   
## 3 643,923 28,102   
## 4 42,357 7,574   
## 5 3,312 64   
## 6 14,441 187   
## 7 62 7   
## 8 1,523,145 56,963 289 19.8   
## 9 66 85 57 3   
## 10 612,071 13,509 2,144 112 3,572,257  
## 11 96,040 7,371 4,082 428 930,230  
## 12 106,962 2,812 2,852 376 1,244,108  
## 13 94,868 6,027 2,267 286 463,662  
## 14 53,931 5,013 1,688 52 1,728,357  
## 15 93,772 1,559 1,601 215 438,991  
## 16 20,472 3,563 946 59 319,879  
## 17 68,146 1,845 931 21 558,413  
## 18 23,014 1,140 3,118 445 139,387  
## 19 17,515 6,634 158 10 62,985  
## 20 29,145 8 219 2 1,718,019  
## Tests/1M pop Continent  
## 1 Asia  
## 2 North America  
## 3 Europe  
## 4 South America  
## 5 Australia/Oceania  
## 6 Africa  
## 7   
## 8 All  
## 9 Asia  
## 10 10,792 North America  
## 11 19,896 Europe  
## 12 20,577 Europe  
## 13 7,103 Europe  
## 14 20,629 Europe  
## 15 6,467 Europe  
## 16 3,808 Asia  
## 17 6,621 Asia  
## 18 12,027 Europe  
## 19 296 South America  
## 20 11,773 Europe

###### 문제 1.1. str 함수를 사용해 저장된 변수의 구조를 출력해보고 저장된 형식의 문제점을 파악해 보세요.

str(covid\_yesterday)

## 'data.frame': 220 obs. of 13 variables:  
## $ Country,Other : chr "Asia" "North America" "Europe" "South America" ...  
## $ TotalCases : chr "358,184" "759,755" "1,029,214" "72,531" ...  
## $ NewCases : chr "+12,188" "+35,195" "+32,674" "+4,940" ...  
## $ TotalDeaths : chr "14,066" "39,422" "96,228" "3,318" ...  
## $ NewDeaths : chr "+1,640" "+2,711" "+4,004" "+261" ...  
## $ TotalRecovered : chr "171,127" "74,274" "289,063" "26,856" ...  
## $ ActiveCases : chr "172,991" "646,059" "643,923" "42,357" ...  
## $ Serious,Critical : chr "6,403" "14,626" "28,102" "7,574" ...  
## $ Tot혻Cases/1M pop: chr "" "" "" "" ...  
## $ Deaths/1M pop : chr "" "" "" "" ...  
## $ TotalTests : chr "" "" "" "" ...  
## $ Tests/1M pop : chr "" "" "" "" ...  
## $ Continent : chr "Asia" "North America" "Europe" "South America" ...

# 1. 해당 covid\_yesterday 데이터 프레임의 NewCasessk NewDeaths 변수의 데이터값에는 값에 특수문자 +(플러스)가 붙어있고 모든 값은 천단위로 절사되어 ,(쉼표)로 끊어져있는 char(문자열)데이터로 구성되어있습니다. 이후 문제 4.1에서와 같이 데이터를 '수(numeric)'로 변환하여 처리해 주어야 하는 경우를 위해 우선 이러한 character형식의 데이터들의 특수문자를 지워주는 전처리가 필요할 것이라고 생각합니다.  
  
  
#3. 해당 데이터프레임의 1~7행까지의 데이터는 대륙별 데이터를 8행은 전세계 데이터를 9행 부터는 국가별 데이터를 포함하고 있는데 이 자료가 모두 한가지 데이터프레임으로 구성되어있어 이후의 자료처리에 혼선이 생길것입니다.

###### 문제 1.2. 왜 그런 문제가 생겼을지 유추해보세요.

# 데이터를 불러들여온 사이트에 게시된 데이터가 사용자의 가독성을 위해 자동적으로 천단위의 절사가 ,(쉼표)로 표기되는 데이터 처리 응용프로그램(ex : excel)을 기반으로 작성/게시 되었고. 그것을 r에 크롤링해 올 때 데이터가 character로 변환된 채로 넘어와 그런것이 아닐까 생각합니다.

###### 문제 1.3. 문제점을 나름대로 해결해보고 해결과정을 서술하세요.

###### HINT: gsub(“패턴”,“패턴을 대체할 내용”, dataframe$column) 함수는 문자열들에서 특정 문자를 찾고, 이를 명시한 내용으로 대체해 줍니다. 예를 들어 문자열에서 공백을 제거하기 위해 data <- gsub(" “,”",data)와 같이 사용하여 변수에 수정된 데이터를 할당해 줍니다.

#편의를 위해 이름을 미리 변경해주도록 한다.   
colnames(covid\_yesterday) <-c("Country","TotalCases","NewCases","TotalDeaths","NewDeaths","TotalRecovered","ActiveCases","Serious","TotCases\_1MPop","Deaths\_1MPop","TotalTests","Tests\_1MPop","Continent")  
  
covid\_yesterday <- covid\_yesterday[covid\_yesterday$Country!="Diamond Princess",] #업데이트에 제시된대로 Diamond Princess 제외  
  
covid\_yesterday$TotalCases <- gsub(",","",covid\_yesterday$TotalCases)  
covid\_yesterday$NewCases <- gsub(",","",covid\_yesterday$NewCases)  
covid\_yesterday$TotalDeaths <- gsub(",","",covid\_yesterday$TotalDeaths)  
covid\_yesterday$NewDeaths <- gsub(",","",covid\_yesterday$NewDeaths)  
covid\_yesterday$TotalRecovered <- gsub(",","",covid\_yesterday$TotalRecovered)  
covid\_yesterday$ActiveCases <- gsub(",","",covid\_yesterday$ActiveCases)  
covid\_yesterday$Serious <- gsub(",","",covid\_yesterday$Serious)  
covid\_yesterday$TotCases\_1MPop <- gsub(",","",covid\_yesterday$TotCases\_1MPop)  
covid\_yesterday$Deaths\_1MPop <- gsub(",","",covid\_yesterday$Deaths\_1MPop)  
covid\_yesterday$TotalTests <- gsub(",","",covid\_yesterday$TotalTests)  
covid\_yesterday$Tests\_1MPop <- gsub(",","",covid\_yesterday$Tests\_1MPop)

###### 문제 1.4. 수정된 데이터를 world\_covid\_19.csv로 저장해 주세요. (row.names=F)

write.csv(covid\_yesterday,"world\_covid19.csv",row.names=F)

## 문제 2

###### 저장된 world\_covid\_19.csv의 데이터를 world\_covid 란 변수에 읽어오세요.

world\_covid <- read.csv("world\_covid19.csv");head(world\_covid,10) #가독성을 위해 head처리하겠습니다.

## Country TotalCases NewCases TotalDeaths NewDeaths TotalRecovered  
## 1 Asia 358184 12188 14066 1640 171127  
## 2 North America 759755 35195 39422 2711 74274  
## 3 Europe 1029214 32674 96228 4004 289063  
## 4 South America 72531 4940 3318 261 26856  
## 5 Oceania 8039 73 76 4 4651  
## 6 Africa 20419 1426 1020 52 4958  
## 7 721 NA 15 NA 644  
## 8 World 2248863 86496 154145 8672 571573  
## 9 China 82692 325 4632 1290 77994  
## 10 USA 709735 32165 37154 2535 60510  
## ActiveCases Serious TotCases\_1MPop Deaths\_1MPop TotalTests Tests\_1MPop  
## 1 172991 6403 NA NA NA NA  
## 2 646059 14626 NA NA NA NA  
## 3 643923 28102 NA NA NA NA  
## 4 42357 7574 NA NA NA NA  
## 5 3312 64 NA NA NA NA  
## 6 14441 187 NA NA NA NA  
## 7 62 7 NA NA NA NA  
## 8 1523145 56963 289 19.8 NA NA  
## 9 66 85 57 3.0 NA NA  
## 10 612071 13509 2144 112.0 3572257 10792  
## Continent  
## 1 Asia  
## 2 North America  
## 3 Europe  
## 4 South America  
## 5 Australia/Oceania  
## 6 Africa  
## 7   
## 8 All  
## 9 Asia  
## 10 North America

###### 문제 2.1. 해당 데이터 프레임의 열이름 중 1열을 Country, 9열을 CasePer1M로 수정하고, 서두에서 사용한다고 언급한 열들만 서브셋팅 하여 동일한 변수에 저장해 주세요.

names(world\_covid)[1] <-"Country"  
names(world\_covid)[9] <-"CasePer1M"  
  
world\_covid <- world\_covid[,c(1,2,3,4,5,6,9,13)]

###### 문제 2.2. world\_covid에는 각 대륙별 합산 정보가 섞여 있습니다. 이 문제를 해결할 수 있도록 대륙별 합산 정보가 담긴 데이터 프레임과 각 국가별 정보가 담긴 데이터 프레임을 만들어보세요. (모든 것을 합산한 All 정보는 포함하지 않아도 됩니다.)

#대륙별 합산 정  
Continent\_covid <- world\_covid[c(1:7),] # world\_covid 데이터 프레임의 7행에 존재하는 값이 무엇인지 언급되지 않아있고, 문제 3.1의 백분율 분석결과 해당 행의 데이터가 포함되어야 정확한 100%의 값을 가지기 때문에 대륙에 포함되지 않은 기타값의 데이터라고 판단하여 7행의 데이터까지 Continent\_covid에 할당하였습니다.  
Country\_covid <- world\_covid[-c(1:8),]

###### 문제 2.3. 두개의 서브셋팅된 데이터프레임을 continent\_covid\_19.csv와 country\_covid19.csv로 저장하세요.

write.csv(Continent\_covid ,"continent\_covid\_19.csv",row.names=F)  
write.csv(Country\_covid,"country\_covid.csv",row.names=F)

## 문제 3

###### 문제 3.1. 각 대륙별 확진자가 전세계 확진자 대비하여 차지하는 비율을 분석해보세요.

WorldPerConVec = Continent\_covid[1:7,2]/world\_covid[8,2] \*100 #백분율 공식  
WorldPerConVec1 = order(WorldPerConVec,decreasing=T) #전세계 확진자 대비 차지하는 비율이 가장 높은 대륙부터 내림차순으로 정렬  
WorldPerConVec = WorldPerConVec[c(WorldPerConVec1)] #""  
cat("차례대로 \n 유럽,북아메리카,아시아,남아메리카,아프리카,오세아니아,기타대륙 순으로 비중순 정렬 : \n",WorldPerConVec[1:7])

## 차례대로   
## 유럽,북아메리카,아시아,남아메리카,아프리카,오세아니아,기타대륙 순으로 비중순 정렬 :   
## 45.76597 33.78396 15.92734 3.22523 0.9079699 0.3574695 0.03206065

###### 문제 3.2. 아시아에서 한국,중국,일본의 확진자가 차지하는 비율을 분석해 보세요.

###### HINT: S.Korea , China ,Japan

Asia\_covid = Country\_covid[Country\_covid$Continent == "Asia",] #변수 continent의 값이 "Asia"인 것만 선언   
as.numeric(Asia\_covid[,2])

## [1] 82692 79494 78546 14352 12982 10635 9787 7142 7025 6302 5923 5878  
## [13] 5251 5050 4663 2700 1838 1740 1658 1546 1482 1405 1340 1201  
## [25] 1069 1022 906 750 668 489 407 402 395 370 268 244  
## [37] 136 122 88 45 38 31 30 29 19 18 5 1

T\_Asia\_covid = sum(Asia\_covid[,2])  
  
  
AsiaPerKor = Asia\_covid[6,2]/T\_Asia\_covid \*100  
AsiaPerJap = Asia\_covid[7,2]/T\_Asia\_covid \*100  
AsiaPerChi = Asia\_covid[1,2]/T\_Asia\_covid \*100  
  
cat("아시아 대비 차지하는 비율 내림차순 정렬\n 중국,대한민국,일본 순 : \n",AsiaPerChi, AsiaPerKor, AsiaPerJap)

## 아시아 대비 차지하는 비율 내림차순 정렬  
## 중국,대한민국,일본 순 :   
## 23.08646 2.969144 2.732395

## 문제 4

###### 문제3의 데이터 프레임을 활용하세요.

###### 문제 4.1. 현재 country\_covid에는 인구정보가 없습니다. 주어진 열들에 대한 정보를 통해 각 나라별 인구 수를 계산하고,Population 열로 추가하시오.

###### HINT : Population = TotalCases/CasePer1M X 1,000,000

Country\_covid$Population <- Country\_covid$TotalCases/Country\_covid$CasePer1M \* 1000000  
head(Country\_covid,3)

## Country TotalCases NewCases TotalDeaths NewDeaths TotalRecovered CasePer1M  
## 9 China 82692 325 4632 1290 77994 57  
## 10 USA 709735 32165 37154 2535 60510 2144  
## 11 Spain 190839 5891 20002 687 74797 4082  
## Continent Population  
## 9 Asia 1450736842  
## 10 North America 331033116  
## 11 Europe 46751347

###### 문제 4.2. 현재 continent\_covid에는 CasePer1M 값이 NA입니다. country\_covid를 활용하여 이 값을 채우시오.

#인구 100만명당 확진자 수  
Country\_covid$Population <- as.numeric(Country\_covid$Population)  
  
a <- Asia\_covid$Population  
a <- sum(a) #아시아 인구 총합  
  
  
#Country\_covid의 데이터프레임에서 각 대륙별로 CasePer1M의 평균을 구한 후 Continent\_covid의 CasePer1M에 할당  
Continent\_covid[1,7] <- mean(Asia\_covid$CasePer1M)  
  
Europe\_covid = Country\_covid[Country\_covid$Continent == "Europe",]   
Continent\_covid[3,7] <- mean(Europe\_covid$CasePer1M)  
  
N.America\_covid = Country\_covid[Country\_covid$Continent == "North America",]   
Continent\_covid[2,7] <- mean(N.America\_covid$CasePer1M)  
  
S.America\_covid = Country\_covid[Country\_covid$Continent == "South America",]  
Continent\_covid[4,7] <- mean(S.America\_covid$CasePer1M)  
  
AO\_covid = Country\_covid[Country\_covid$Continent == "Australia/Oceania",]  
Continent\_covid[5,7] <- mean(AO\_covid$CasePer1M)  
  
Africa\_covid = Country\_covid[Country\_covid$Continent == "Africa",]  
Continent\_covid[6,7] <- mean(Africa\_covid$CasePer1M)  
  
  
Continent\_covid

## Country TotalCases NewCases TotalDeaths NewDeaths TotalRecovered  
## 1 Asia 358184 12188 14066 1640 171127  
## 2 North America 759755 35195 39422 2711 74274  
## 3 Europe 1029214 32674 96228 4004 289063  
## 4 South America 72531 4940 3318 261 26856  
## 5 Oceania 8039 73 76 4 4651  
## 6 Africa 20419 1426 1020 52 4958  
## 7 721 NA 15 NA 644  
## CasePer1M Continent  
## 1 236.75063 Asia  
## 2 421.10256 North America  
## 3 2115.66667 Europe  
## 4 389.71429 South America  
## 5 137.80000 Australia/Oceania  
## 6 60.72545 Africa  
## 7 NA

# 앞서 문제 2.2에서 언급했던대로 , world\_covid 데이터 프레임의 7행에 존재하는 값이 무엇인지 언급되지 않아있고, 문제 3.1의 백분율 분석결과 해당 행의 데이터가 포함되어야 정확한 100%의 값을 가지기 때문에 대륙에 포함되지 않은 기타값의 데이터라고 판단하여 무명의 7행 데이터까지 Continent\_covid에 할당한 상태입니다.