PR4 - R Factor, Matrix, List, Array

김서준

07 10월, 2023

1. Factor

1.1. 명목형 자료 만들기

```
score <- factor(c(3, 2, 3, 4, 3, 1, 1, 3, 2, 2, 1, 1, 5),
                levels = c(1, 2, 3, 4),
                labels = c("A", "B", "C", "D"),
                ordered = T)
score
```

```
##  [1] C    B    C    D    C    A    A    C    B    B    A    A    <NA>
## Levels: A < B < C < D
```

1.2. 명목형 자료로 변환하기

```
#문자를 벡터에 입력하였을 때
fac_char <- c("포도", "키위", "메론", "바나나", "딸기")
attributes(fac_char)
```

```
## NULL
```

```
#문자벡터를 명목형 자료로 변경하였을 때
fac_char <- as.factor(fac_char)
attributes(fac_char)
```

```
## $levels
## [1] "딸기"   "메론"   "바나나" "키위"   "포도"
##
## $class
## [1] "factor"
```

```
#숫자를 벡터에 입력하며 명목형으로 변경도 가능
fac_num <- 1:4
attributes(fac_num)
```

```
## NULL
```

```
fac_num <- as.factor(fac_num)
attributes(fac_num)
```

```
## $levels
## [1] "1" "2" "3" "4"
##
## $class
## [1] "factor"
```

1.3. 팩터형 자료 빈도 파악

```
table(score)
```

```
## score
## A B C D
## 4 3 4 1
```

```
#빈도가 3 이상인 데이터만 출력
tb <- table(score)
tb[tb>3]
```

```
## score
## A C
## 4 4
```

1.4. 서수형 자료와 명목형 자료의 차이

```
score[1] > score[3] #(1)
```

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

```
fac_char[1] > fac_char[2] #(2)
```

```
## Warning in Ops.factor(fac_char[1], fac_char[2]): 요인(factors)에 대하여
## 의미있는 '>'가 아닙니다.
```

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

```
#이곳에 주석으로 (1) 과 (2)의 차이를 서술하고 왜 그런 차이가 생기는지 각자 분석해보세요.
#(1)은 순서형 요인으로 값들 간 순서를 고려한 비교 가능, (2) 는 명목형 요인으로 순서를 고려하지 않아 순서를 고려한 비교가 불가능하다.
```

2. Matrix

2.1. matrix 생성

```
mat <- matrix(1:8, nrow=2, ncol=4, byrow=T) #1~8 의 숫자로 2행 4 열의 행렬 생성, 가로(열)로 배열
dim(mat); length(mat)
```

```
## [1] 2 4
```

```
## [1] 8
```

```
matrix(1:8, nrow=2, ncol=4, byrow=F)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    5    7
## [2,]    2    4    6    8
```

```
matrix(1:8, nrow=2)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    5    7
## [2,]    2    4    6    8
```

```
matrix(1:8, ncol=2)
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    6
## [3,]    3    7
## [4,]    4    8
```

```
matrix(1:8, ncol=4, byrow=T)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    5    6    7    8
```

```
matrix(1:9, nrow=3, ncol=3,
       dimnames = (list(c("r1", "r2","r3"), c("c1", "c2", "c3"))))
```

```
##    c1 c2 c3
## r1  1  4  7
## r2  2  5  8
## r3  3  6  9
```

2.2. matrix 각 차원에 이름 부여

```
mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    5    6    7    8
```

```
rownames(mat)<-c("행 1","행 2")
colnames(mat)<-c("열 1","열 2","열 3","열 4")
mat
```

```
##      열 1 열 2 열 3 열 4
## 행 1    1    2    3    4
## 행 2    5    6    7    8
```

2.3. rbind() 와 cbind()를 사용한 매트릭스 생성

```
x <- 2:4;x
```

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

```
y <- 9:11; y
```

```
## [1]  9 10 11
```

```
cbind(x, y)
```

```
##      x  y
## [1,] 2  9
## [2,] 3 10
## [3,] 4 11
```

```
rbind(x, y)
```

```
##   [,1] [,2] [,3]
## x    2    3    4
## y    9   10   11
```

2.4. rbind() 와 cbind()를 사용한 데이터 추가

```
mat
```

```
##      열 1 열 2 열 3 열 4
## 행 1    1    2    3    4
## 행 2    5    6    7    8
```

```
cbind(mat,10:11)
```

```
##      열 1 열 2 열 3 열 4
## 행 1    1    2    3    4 10
## 행 2    5    6    7    8 11
```

```
rbind(mat, 20:23)
```

```
##      열 1 열 2 열 3 열 4
## 행 1    1    2    3    4
## 행 2    5    6    7    8
##        20   21   22   23
```

2.5. matrix 데이터 접근과 변환

```
x <- 1:3 ;x
```

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

```
y <- 10:12 ; y
```

```
## [1] 10 11 12
```

```
mat <- cbind(x, y) ; mat
```

```
##      x  y
## [1,] 1 10
## [2,] 2 11
## [3,] 3 12
```

```
mat[1,1] <- 100; mat
```

```
##        x  y
## [1,] 100 10
## [2,]   2 11
## [3,]   3 12
```

```
mat[2,] <- mat[2,] / 4 ; mat
```

```
##          x     y
## [1,] 100.0 10.00
## [2,]   0.5  2.75
## [3,]   3.0 12.00
```

```
mat[,2] <- mat[,2] - mat[,1] * 3 ; mat
```

```
##          x       y
## [1,] 100.0 -290.00
## [2,]   0.5    1.25
## [3,]   3.0    3.00
```

3. List

3.1. 여러 벡터를 이용해 리스트 만들기

```
str_vec <- c("korea", "USA", "Japan")
num_vec <- c(100, 200, 300, 400, 500)
mat <- matrix(2:9, 2, 4)
list_tot <- list(str_vec, num_vec, mat)
print(list_tot)
```

```
## [[1]]
## [1] "korea" "USA"   "Japan"
##
## [[2]]
## [1] 100 200 300 400 500
##
## [[3]]
##      [,1] [,2] [,3] [,4]
## [1,]    2    4    6    8
## [2,]    3    5    7    9
```

```
names(list_tot) <- c('str_vec', 'num_vec', 'mat')
print(list_tot)
```

```
## $str_vec
## [1] "korea" "USA"   "Japan"
##
## $num_vec
## [1] 100 200 300 400 500
##
## $mat
##      [,1] [,2] [,3] [,4]
## [1,]    2    4    6    8
## [2,]    3    5    7    9
```

3.2 list 함수 내에서 성분의 이름 지정하여 리스트 만들기

```
list_tot2= list(seq = seq(1,10,2),
                str = c("토끼", "사자", "코끼리", "양"),
                plus = rep(c('고구마', '감자', '옥수수'),2))
print (list_tot2)
```

```
## $seq
## [1] 1 3 5 7 9
##
## $str
## [1] "토끼"   "사자"   "코끼리" "양"
##
## $plus
## [1] "고구마" "감자"   "옥수수" "고구마" "감자"   "옥수수"
```

3.3. 데이터 속성을 화인하는 다양한 함수

```
class (list_tot)
```

```
## [1] "list"
```

```
length(list_tot)
```

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

```
attributes (list_tot)
```

```
## $names
## [1] "str_vec" "num_vec" "mat"
```

```
str(list_tot)
```

```
## List of 3
##  $ str_vec: chr [1:3] "korea" "USA" "Japan"
##  $ num_vec: num [1:5] 100 200 300 400 500
##  $ mat    : int [1:2, 1:4] 2 3 4 5 6 7 8 9
```

3.4. 리스트의 성분에 접근하기

```
list_tot2 [1]
```

```
## $seq
## [1] 1 3 5 7 9
```

```
list_tot2 [3]
```

```
## $plus
## [1] "고구마" "감자"   "옥수수" "고구마" "감자"   "옥수수"
```

```
list_tot2 [1:2]
```

```
## $seq
## [1] 1 3 5 7 9
##
## $str
## [1] "토끼"   "사자"   "코끼리" "양"
```

```
list_tot2$seq
```

```
## [1] 1 3 5 7 9
```

```
list_tot2$str
```

```
## [1] "토끼"   "사자"   "코끼리" "양"
```

3.5. 리스트의 성분안에 있는 원소에 접근하기

```
list_tot[[3]][1]
```

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

```
list_tot2$seq[3]
```

```
## [1] 5
```

```
list_tot2$str[1:2]
```

```
## [1] "토끼" "사자"
```

3.6. 리스트의 성분이나 원소 조작하기

```
list_tot[1] <- NULL
str(list_tot)
```

```
## List of 2
##  $ num_vec: num [1:5] 100 200 300 400 500
##  $ mat    : int [1:2, 1:4] 2 3 4 5 6 7 8 9
```

```
list_tot2$str [1] <- "고양이"
str(list_tot2)
```

```
## List of 3
##  $ seq : num [1:5] 1 3 5 7 9
##  $ str : chr [1:4] "고양이" "사자" "코끼리" "양"
##  $ plus: chr [1:6] "고구마" "감자" "옥수수" "고구마" ...
```

```
list_tot$NEW <- 2:5
str(list_tot)
```

```
## List of 3
##  $ num_vec: num [1:5] 100 200 300 400 500
##  $ mat    : int [1:2, 1:4] 2 3 4 5 6 7 8 9
##  $ NEW    : int [1:4] 2 3 4 5
```

3.7. 리스트의 성분에 하위 리스트 추가하여 계층적으로 리스트 만들기

```
list_tot$hierarchy[[1]] <- list_tot2
str(list_tot)
```

```
## List of 4
##  $ num_vec  : num [1:5] 100 200 300 400 500
##  $ mat      : int [1:2, 1:4] 2 3 4 5 6 7 8 9
##  $ NEW      : int [1:4] 2 3 4 5
##  $ hierarchy:List of 1
##   ..$ :List of 3
##   .. ..$ seq : num [1:5] 1 3 5 7 9
##   .. ..$ str : chr [1:4] "고양이" "사자" "코끼리" "양"
##   .. ..$ plus: chr [1:6] "고구마" "감자" "옥수수" "고구마" ...
```

4. Array

4.1. Array 생성하기

```
arr <- array(1:18, dim=c(3,3,2),
dimnames=list(c("KOR","CHI","JAP"),
              c("GDP.R", "USD. R", "Cuur. Acc"),
              c("2011Y", "2012Y")))
arr
```

```
## , , 2011Y
##
##     GDP.R USD. R Cuur. Acc
## KOR     1      4         7
## CHI     2      5         8
## JAP     3      6         9
##
## , , 2012Y
##
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## CHI    11     14        17
## JAP    12     15        18
```

```
#벡터 생성후 차원을 부여하며 array 로 변환하기
arr1 <- 1:18
dim(arr1) <- c(3,3,2)
dimnames (arr1) <- list(c("KOR","CHI", "JAP"),
                        c("GDP.R", "USD. R", "Cuur. Acc"),
                        c("2011Y", "2012Y"))
arr1
```

```
## , , 2011Y
##
##     GDP.R USD. R Cuur. Acc
## KOR     1      4         7
## CHI     2      5         8
## JAP     3      6         9
##
## , , 2012Y
##
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## CHI    11     14        17
## JAP    12     15        18
```

```
arr1 == arr
```

```
## , , 2011Y
##
##     GDP.R USD. R Cuur. Acc
## KOR  TRUE   TRUE      TRUE
## CHI  TRUE   TRUE      TRUE
## JAP  TRUE   TRUE      TRUE
##
## , , 2012Y
##
##     GDP.R USD. R Cuur. Acc
## KOR  TRUE   TRUE      TRUE
## CHI  TRUE   TRUE      TRUE
## JAP  TRUE   TRUE      TRUE
```

4. Array 조작 방법

4.2.1. [,,] 인덱싱으로 각 원소에 접근하기

```
arr
```

```
## , , 2011Y
##
##     GDP.R USD. R Cuur. Acc
## KOR     1      4         7
## CHI     2      5         8
## JAP     3      6         9
##
## , , 2012Y
##
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## CHI    11     14        17
## JAP    12     15        18
```

```
arr[1,,]#한국의 연도별 자료
```

```
##           2011Y 2012Y
## GDP.R         1    10
## USD. R        4    13
## Cuur. Acc     7    16
```

```
arr[,-2,]
```

```
## , , 2011Y
##
##     GDP.R Cuur. Acc
## KOR     1         7
## CHI     2         8
## JAP     3         9
##
## , , 2012Y
##
##     GDP.R Cuur. Acc
## KOR    10        16
## CHI    11        17
## JAP    12        18
```

```
arr[,,2]#3개국의 2012 년자료
```

```
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## CHI    11     14        17
## JAP    12     15        18
```

```
arr[,,"2012Y"] #이름으로 추출 (3 개국의 2012 년자료)
```

```
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## CHI    11     14        17
## JAP    12     15        18
```

```
arr[c(T,T,F),,2] #한국, 중국의 2012년 자료 - 일본제외
```

```
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## CHI    11     14        17
```

```
arr[-2,,2] #한국, 일본의 2012년 자료
```

```
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## JAP    12     15        18
```

4.2.2. 배열 원소의 추출 및 수정

```
arr.tmp <- arr
arr.tmp
```

```
## , , 2011Y
##
##     GDP.R USD. R Cuur. Acc
## KOR     1      4         7
## CHI     2      5         8
## JAP     3      6         9
##
## , , 2012Y
##
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## CHI    11     14        17
## JAP    12     15        18
```

```
arr.tmp[,,1] <- c(5,6,4)
arr.tmp
```

```
## , , 2011Y
##
##     GDP.R USD. R Cuur. Acc
## KOR     5      5         5
## CHI     6      6         6
## JAP     4      4         4
##
## , , 2012Y
##
##     GDP.R USD. R Cuur. Acc
## KOR    10     13        16
## CHI    11     14        17
## JAP    12     15        18
```

```
arr.tmp[,1,2] <- NA
arr.tmp
```

```
## , , 2011Y
##
##     GDP.R USD. R Cuur. Acc
## KOR     5      5         5
## CHI     6      6         6
## JAP     4      4         4
##
## , , 2012Y
##
##     GDP.R USD. R Cuur. Acc
## KOR    NA     13        16
## CHI    NA     14        17
## JAP    NA     15        18
```

```
arr.tmp[is.na(arr.tmp)] <- c(8,1)
```

```
## Warning in arr.tmp[is.na(arr.tmp)] <- c(8, 1): number of items to replace is
## not a multiple of replacement length
```

```
arr.tmp
```

```
## , , 2011Y
##
##     GDP.R USD. R Cuur. Acc
## KOR     5      5         5
## CHI     6      6         6
## JAP     4      4         4
##
## , , 2012Y
##
##     GDP.R USD. R Cuur. Acc
## KOR     8     13        16
## CHI     1     14        17
## JAP     8     15        18
```

5. 기타

5.1. NA 값 다루기

```
x <- c(1, 2, NA, 4, NA, 5)
is.na(x)
```

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

```
bad <- is.na(x)
y <- x[!bad]
mean(y)
```

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

```
x <- c(1, 2, NA, 4, NA, 5)
good <- complete.cases(x)
x[good]
```

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

PR4 연습문제

문제 1

```
#문제 1.1.
load('satellite.RData')
length(paper[paper==paper$위성영상])
```

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

```
# 문제 1.2.
paper[paper == paper$'&'] <- NULL
length(paper[paper==paper$'&'])
```

```
## [1] 0
```

```
# 문제 1.3.
paper[19]
```

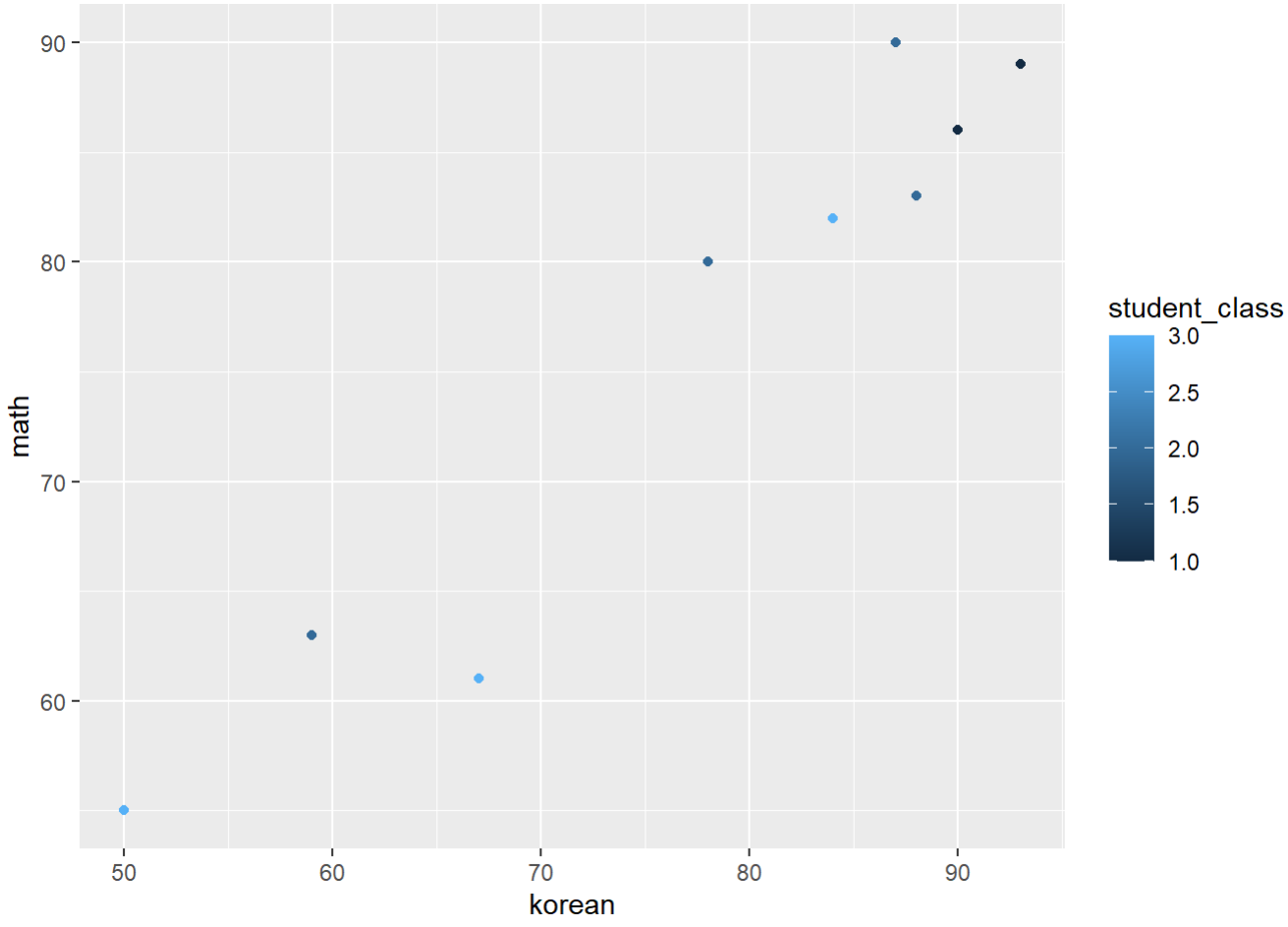
```
## $방법론의
## [1] "방법론/NC+의/JC"
```

```
paper$방법론의[1] <- "방법론/NC"
paper$방법론의[2] <- "의/JC"
paper[19]
```

```
## $방법론의
## [1] "방법론/NC" "의/JC"
```

문제 2

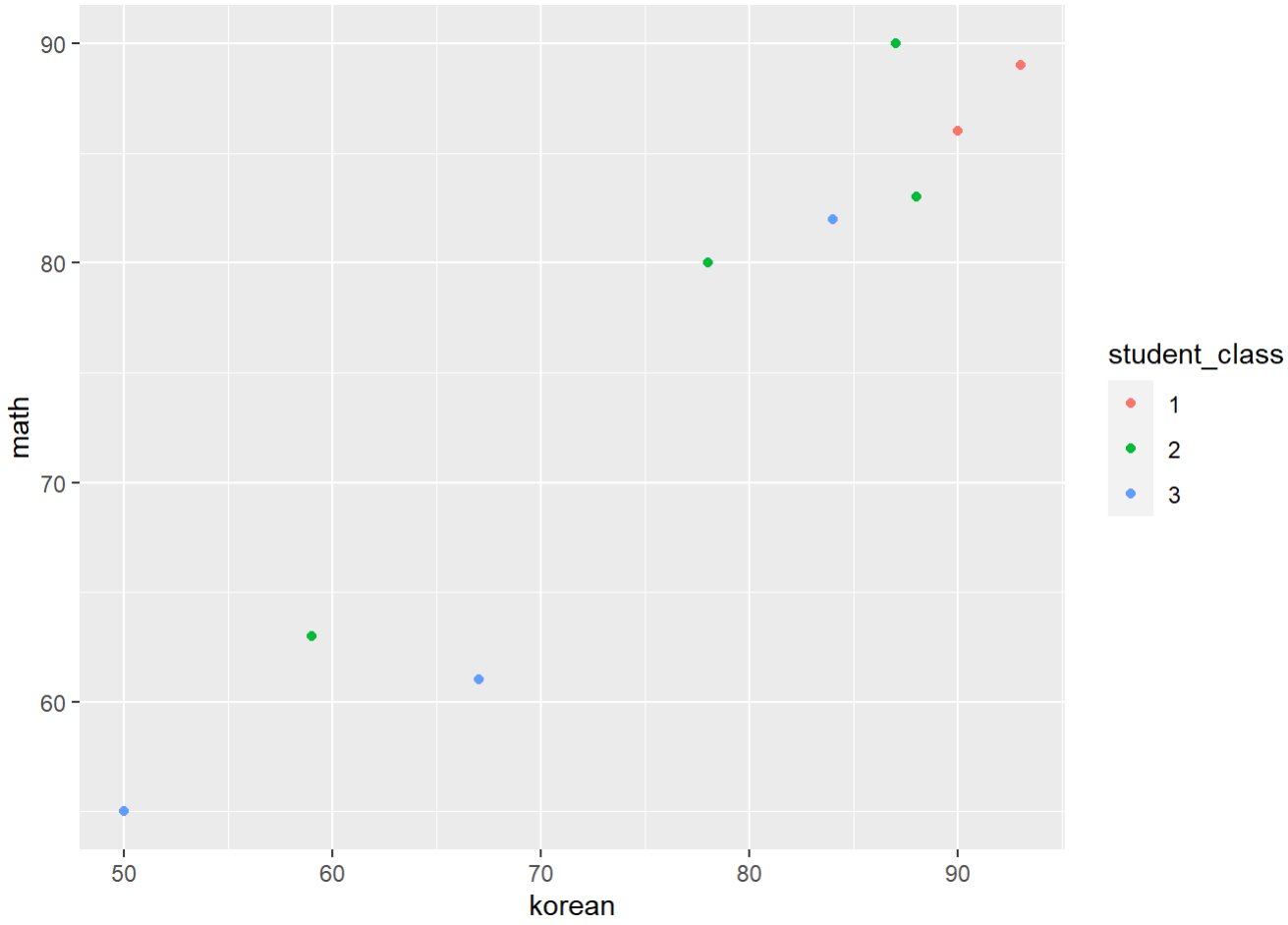
```
#원래 그래프
korean <- c(93, 78, 50, 90, 88, 87, 67, 59, 84)
math <- c(89, 80, 55, 86, 83, 90, 61, 63, 82)
student_class <- c(1, 2, 3, 1, 2, 2, 3, 2, 3)

library(ggplot2)
ggplot() + geom_point(aes(x=korean, y=math, colour=student_class))
```




```
print("student_class는 범주형 자료이며 factor 함수를 통해 생성해야 한다. 원래 그래프의 문제점은 첫 번째 코드는 student_class 변수를 요인(factor)으로 변환하지 않았다. 따라서 student_class 변수는 숫자형 벡터로 처리되고 student_class 변수를 벡터로 처리하기에 그래프의 색상은 학생 클래스별로 다른 색으로 설정되지 않는다.")
```

```
## [1] "student_class는 범주형 자료이며 factor 함수를 통해 생성해야 한다. 원래 그래프의 문제점은 첫 번째 코드는 student_class 변수를 요인(factor)으로 변환하지 않았다. 따라서 student_class 변수는 숫자형 벡터로 처리되고 student_class 변수를 벡터로 처리하기에 그래프의 색상은 학생 클래스별로 다른 색으로 설정되지 않는다."
```

```
#수정된 그래프
korean <- c(93, 78, 50, 90, 88, 87, 67, 59, 84)
math <- c(89, 80, 55, 86, 83, 90, 61, 63, 82)
student_class <- factor(student_class,levels = c("1", "2", "3"))
library(ggplot2)
ggplot() + geom_point(aes(x=korean, y=math, colour=student_class))
```



도전문제