

‘기초 통계량’과 ‘시각화’ 기초

비타민 2조

김재승 노신희 우현우 이지선

기초 통계량과 시각화의 목적

주어진 데이터의 대표적 성질을 파악 & 공유

- 데이터 전체를 대표하는 값은 무엇인가?
- 데이터가 어떻게 분포하고 있는가?
- 데이터 전체에서 유의미한 범위는 어느 정도인가?
- 데이터의 성질을 다른 사람에게 가장 잘 표현하는 방법은?

목차

1 평균값과 중앙값

2 사분위수

3 상자그림

4 히스토그램

5 분산과 표준편차

6 파이차트

7 막대차트

1 CHAPTER 3.1-3.2

평균과 중간값

설명

평균

자료 전체의 합을 자료 개수로 나눈 산술평균을 의미하며, 직관적이고 이해하기 쉬워서 대표하는 지표로 많이 사용됨

mean() : 평균을 구하기 위한 함수

코드

[edit](#) [fork](#) [download](#) [copy](#)

```
1. # A식당의 배달시간 벡터 생성
2. storeA <- c(15,25,70,100)
3. mean(storeA)
```

Success #stdin #stdout 0.24s 38844KB [comments \(0\)](#)

stdin [copy](#)

Standard input is empty

stdout [copy](#)

[1] 52.5

1 CHAPTER 3.1-3.2

평균과 중간값

설명

왜 평균만으로는 집단의 상태를 표현하는데 부족할까?

코드

</> source code

```
1 # 재승(J)과 현우(H)의 시험성적
2 J <- c(1,1,100,100)
3 H <- c(30,40,40,50)
4 mean(J)
5 mean(H)
```

input Output

Success #stdin #stdout 0.22s 39336KB

[1] 50.5

[1] 40

평균은 특이한 경우의 상황을
고려해야 하지 못함

1 CHAPTER 3.1-3.2

평균과 중간값

설명

중앙값 숫자를 크기순으로 정렬했을 때, 정가운데 숫자를 의미함

median() : 중앙값을 구하기 위한 함수

코드

</> source code

close shortcuts fullscreen ↗

```
1 # A식당의 배달시간 벡터 생성
2 storeA <- c(15,25,70,100)
3
4 # A식당의 배달시간 중앙값 확인
5 median (storeA)
6
7 # B식당의 배달시간 벡터 생성 및 중앙값 확인
8 storeB <- c(10,20,50,70,100)
9 median (storeB)
```

input

Output

clear the output

☒ syntax highlight

Success #stdin #stdout 0.24s 39192KB

[1] 47.5

[1] 50

2 CHAPTER 3.3

사분위수

설명

사분위수

데이터를 크기순으로 정렬한 뒤 0%, 25%, 50%, 75%, 100% 지점의 숫자를 의미함

quantile() : 사분위수를 구하기 위한 함수

코드

</> source code

```
1 # 벡터 선언
2 nums <- c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17)
3
4 # nums의 사분위수 산출
5 quantile(nums)
```

input Output

Success #stdin #stdout 0.2s 39368KB

0%	25%	50%	75%	100%
1	5	9	13	17

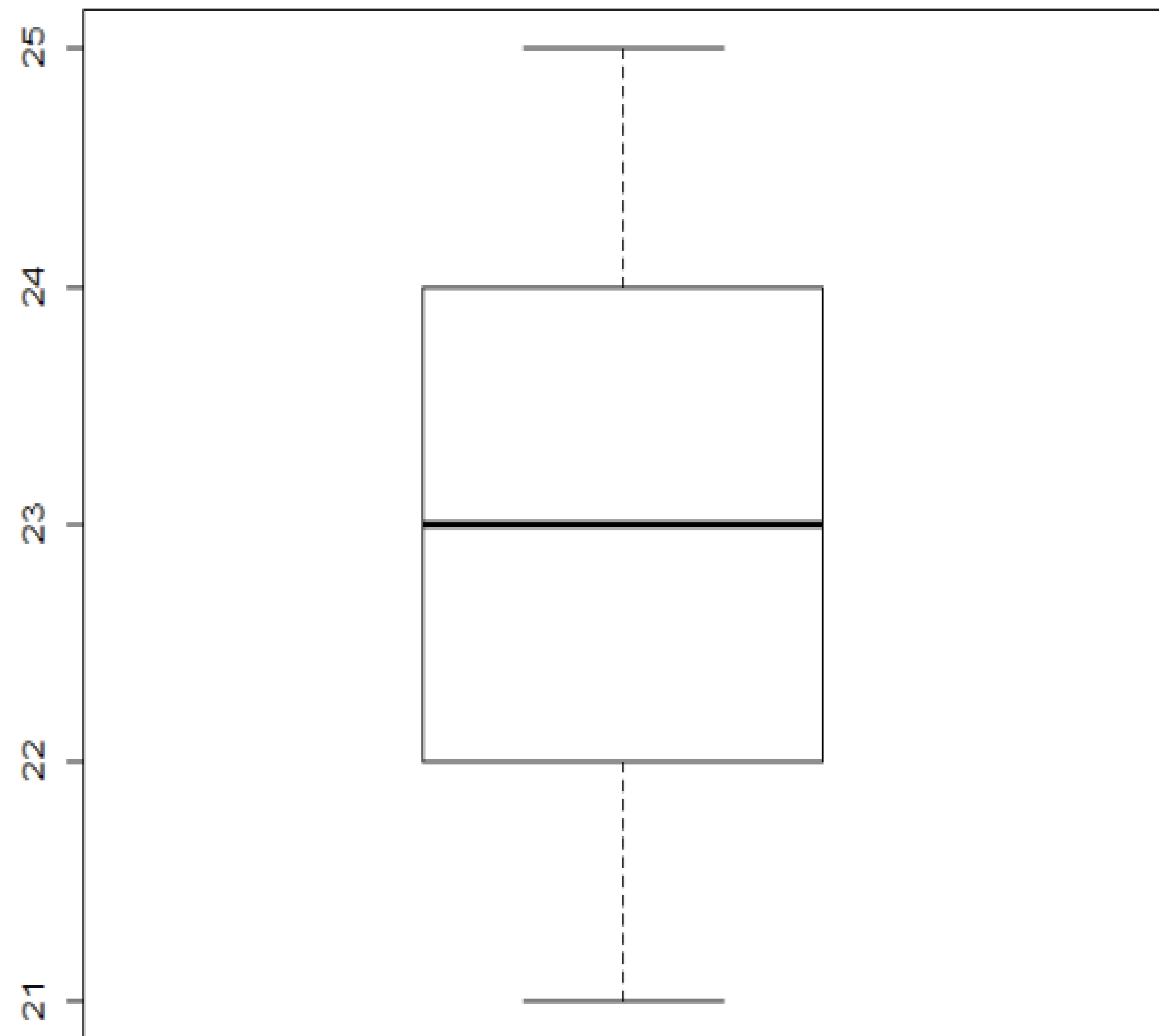
CHAPTER 3.4 상자그림

boxplot(**데이터**)

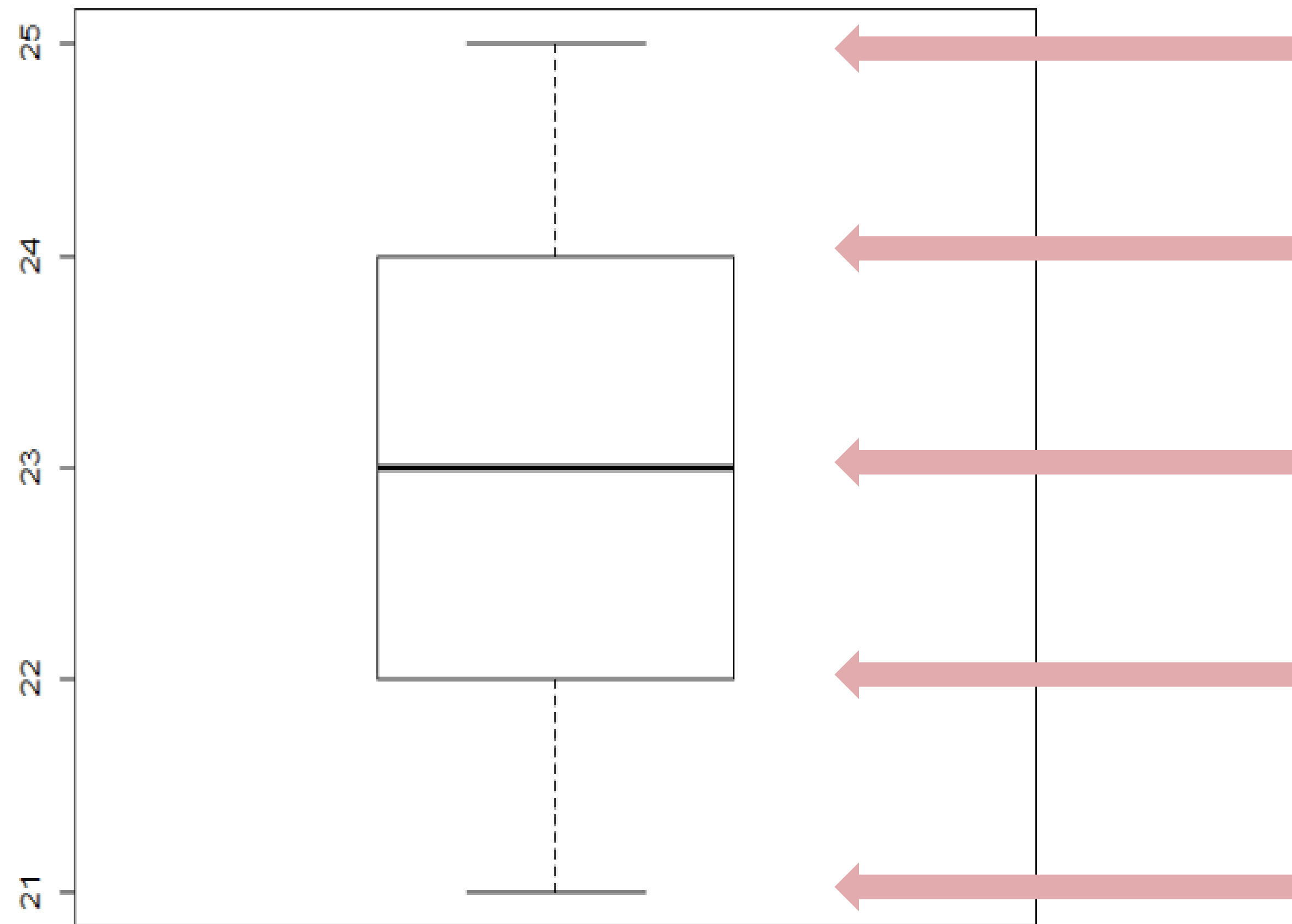
```
1 # 벡터 생성
2 boxNums <- c(21,22,23,24,25)
3
4 # 사분위수 확인
5 quantile(boxNums)
6
7 # 상자그림 그리기
8 boxplot(boxNums)
```

```
> # 벡터 생성
> boxNums <- c(21,22,23,24,25)
> # 사분위수 확인
> quantile(boxNums)
 0%  25%  50%  75% 100%
21   22   23   24   25
> # 상자그림 그리기
> boxplot(boxNums)
```

< 오른쪽 plots 창에 상자그림이 표시됨 >



3 CHAPTER 3.4 상자그림 특징



이상치를 제외한 최댓값

3사분위수

중간 값(2사분위수)

1사분위수

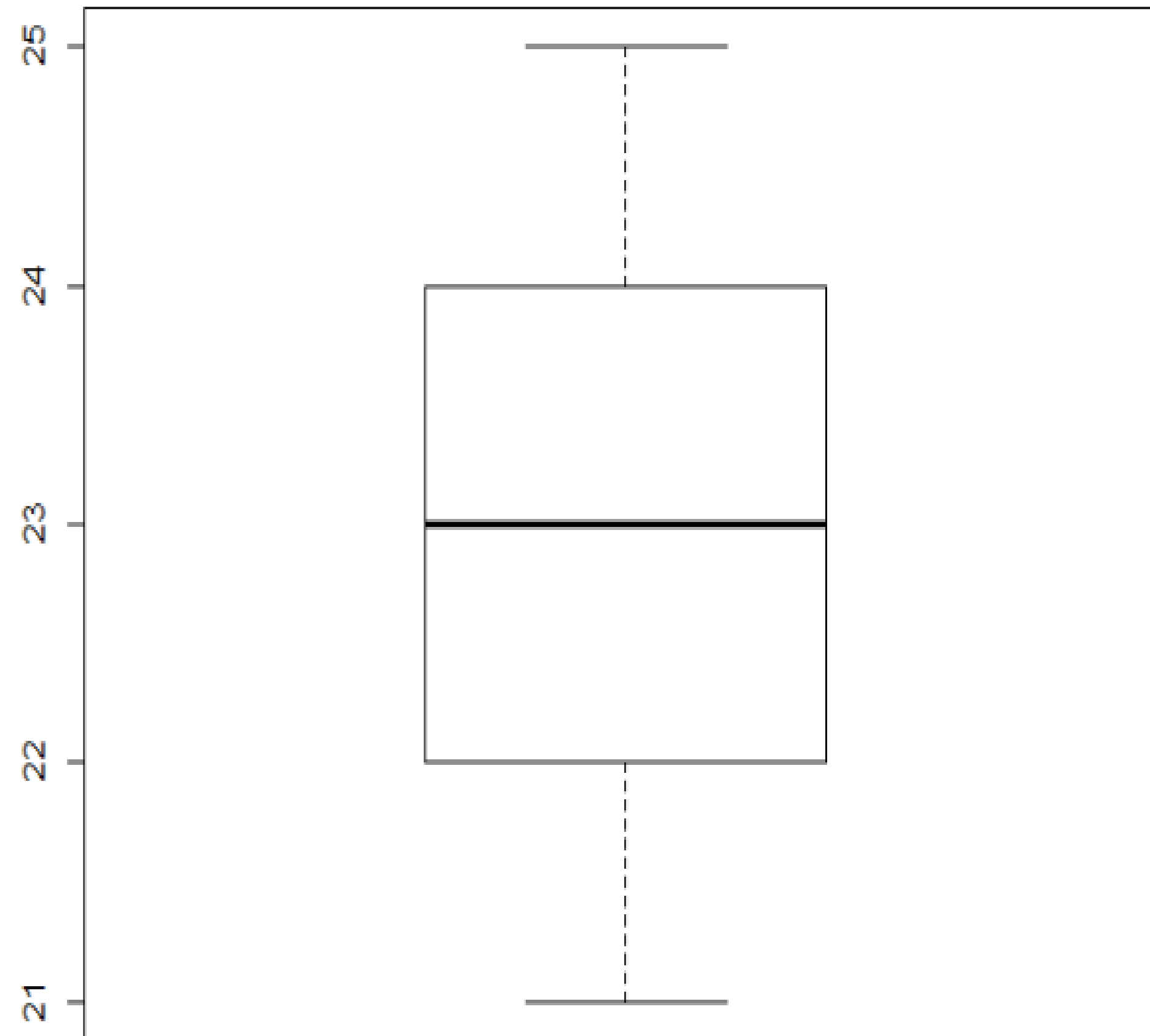
이상치를 제외한 최솟값

< 상자 그림 그리는 순서 >

1. 1&3 사분위수를 아래위로 하는 박스 그리기
2. 박스 중간에 중간 값을 표기
3. 이상치 범위 표시
4. 이상치 범위와 박스를 연결
5. 이상치는 별도 표시



알아서 이 순서로 그려 줌



CHAPTER 3.4 이상치

설명

이상치(outlier) : 데이터 분포상 정상범위를 벗어난 값

정상치의 기준 : $IQR(3사분위수 - 1사분위수) * 1.5$

이상치

: 정상치의 범위를 벗어나는 값

(상자 그림 박스의 높이의 1.5배를 벗어나는 값)

< 주의할 점 >

- 이상치는 별도 기호로 표현
- 분석 목적에 따라 정상치의 기준 변경 가능

CHAPTER 3.4 이상치

이상치(Outlier) 포함해서 상자그림 그리기

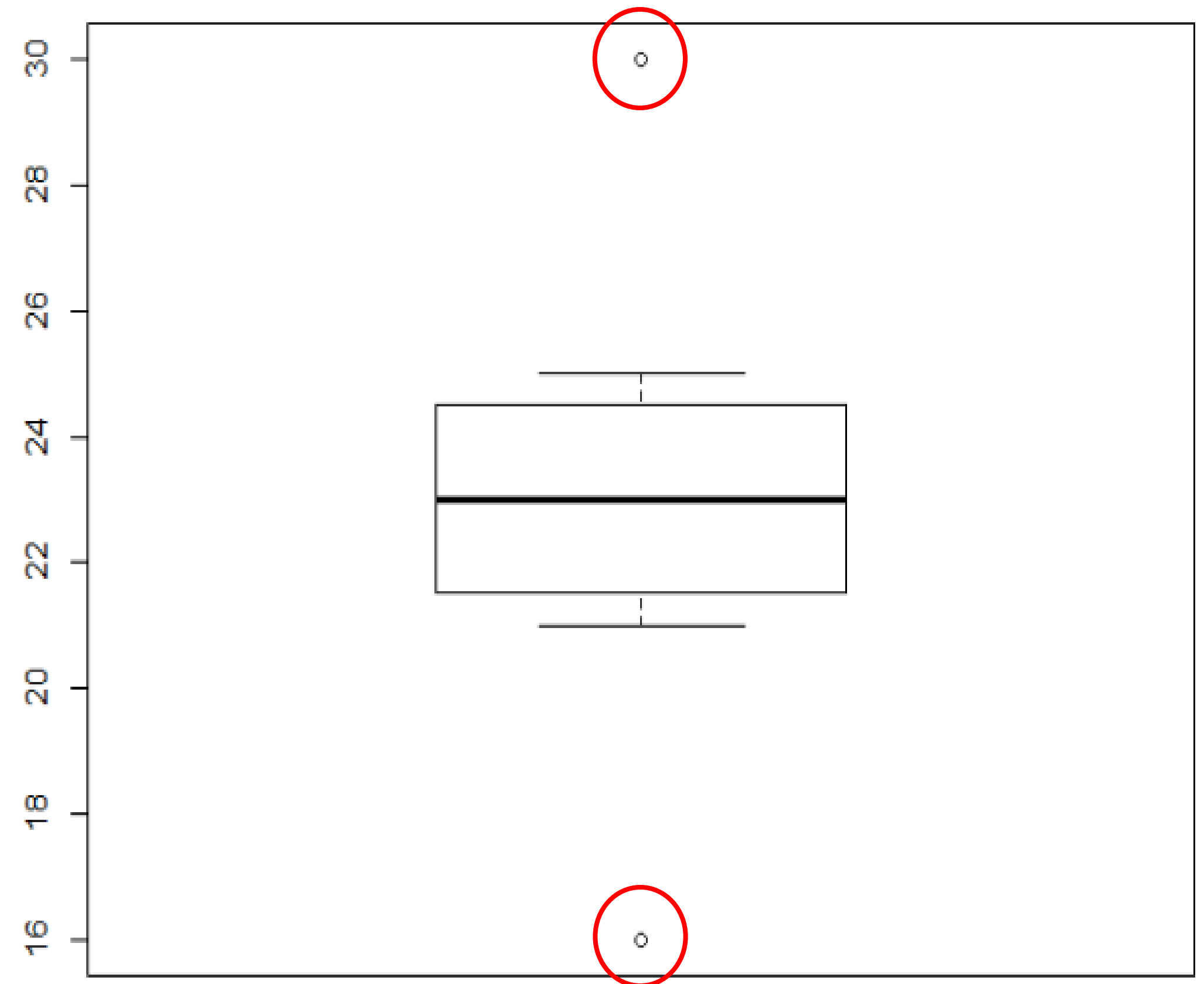
```
# 이상치 추가해서 상자그림 그리기
boxNums_1 <- c(16,21,22,23,24,25,30)

# 사분위수 확인
quantile(boxNums_1)

# 상자그림 그리기
boxplot(boxNums_1)
```

```
> # 이상치 추가해서 상자그림 그리기
> boxNums_1 <- c(16,21,22,23,24,25,30)
> # 사분위수 확인
> quantile(boxNums_1)
 0%  25%  50%  75% 100%
16.0 21.5 23.0 24.5 30.0
> # 상자그림 그리기
> boxplot(boxNums_1)
```

< 이상치(outlier)가 동그라미로 표시됨 >



3 CHAPTER 3.4 이상치

< 앞 명령문의 결과 창 >

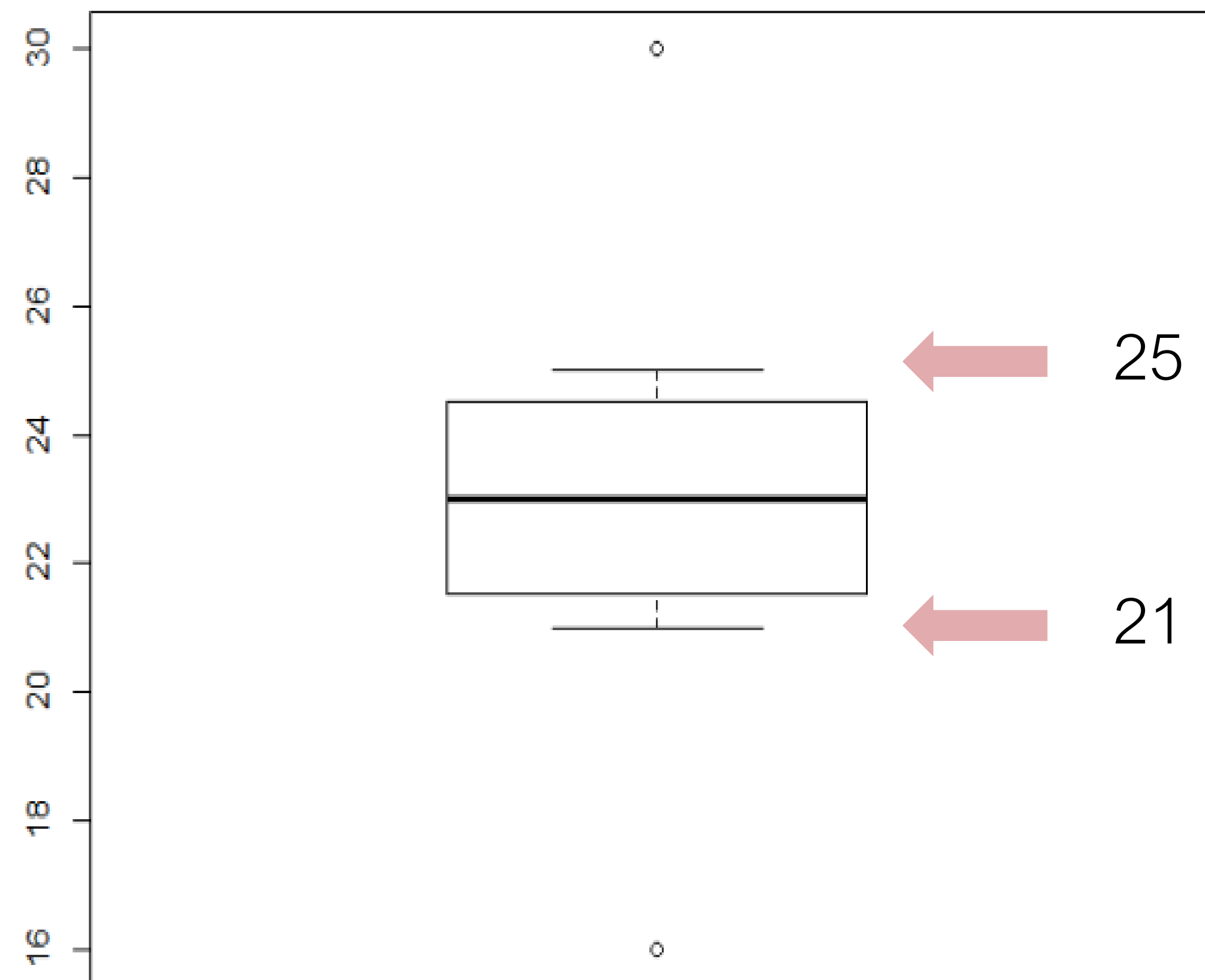
```
> # 이상치 추가해서 상자그림 그리기
> boxNums_1 <- c(16,21,22,23,24,25,30)
> # 사분위수 확인
> quantile(boxNums_1)
 0%  25%  50%  75% 100%
16.0 21.5 23.0 24.5 30.0
> # 상자그림 그리기
> boxplot(boxNums_1)
```

< 정상치 기준 구하기 >

$$IQR * 1.5 = (24.5 - 21.5) * 1.5 = 4.5$$

정상치 범위 : 17이상 29이하

(계산 과정 : $21.5 - 4.5 = 17$ / $24.5 + 4.5 = 29$)



< 박스와 연결된 위, 아래의 실선 >

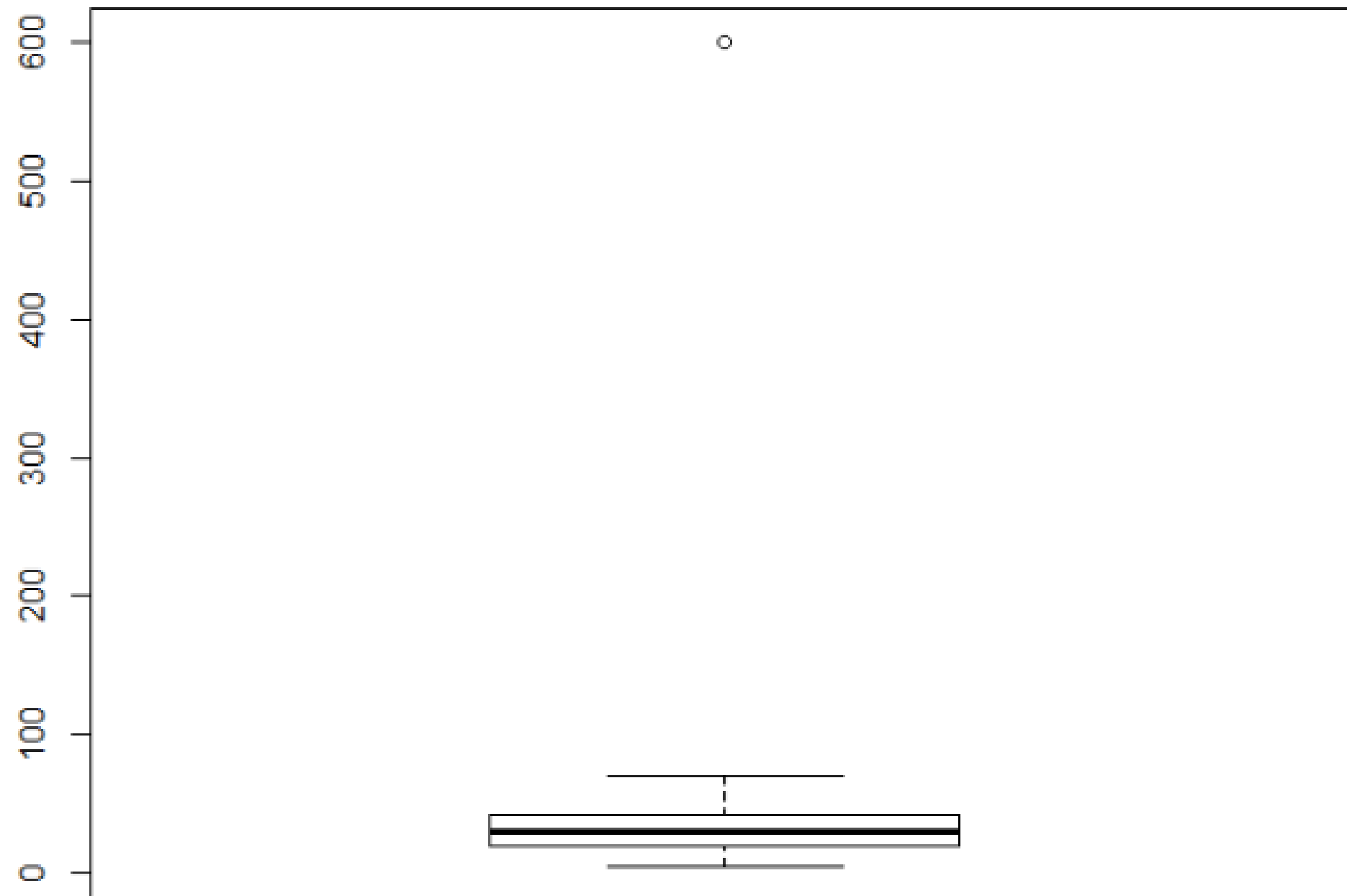
이상치를 제외한 최솟값 : 21

이상치를 제외한 최댓값 : 25

< 예시 : B식당의 배달시간
(이상치 제거 전) >

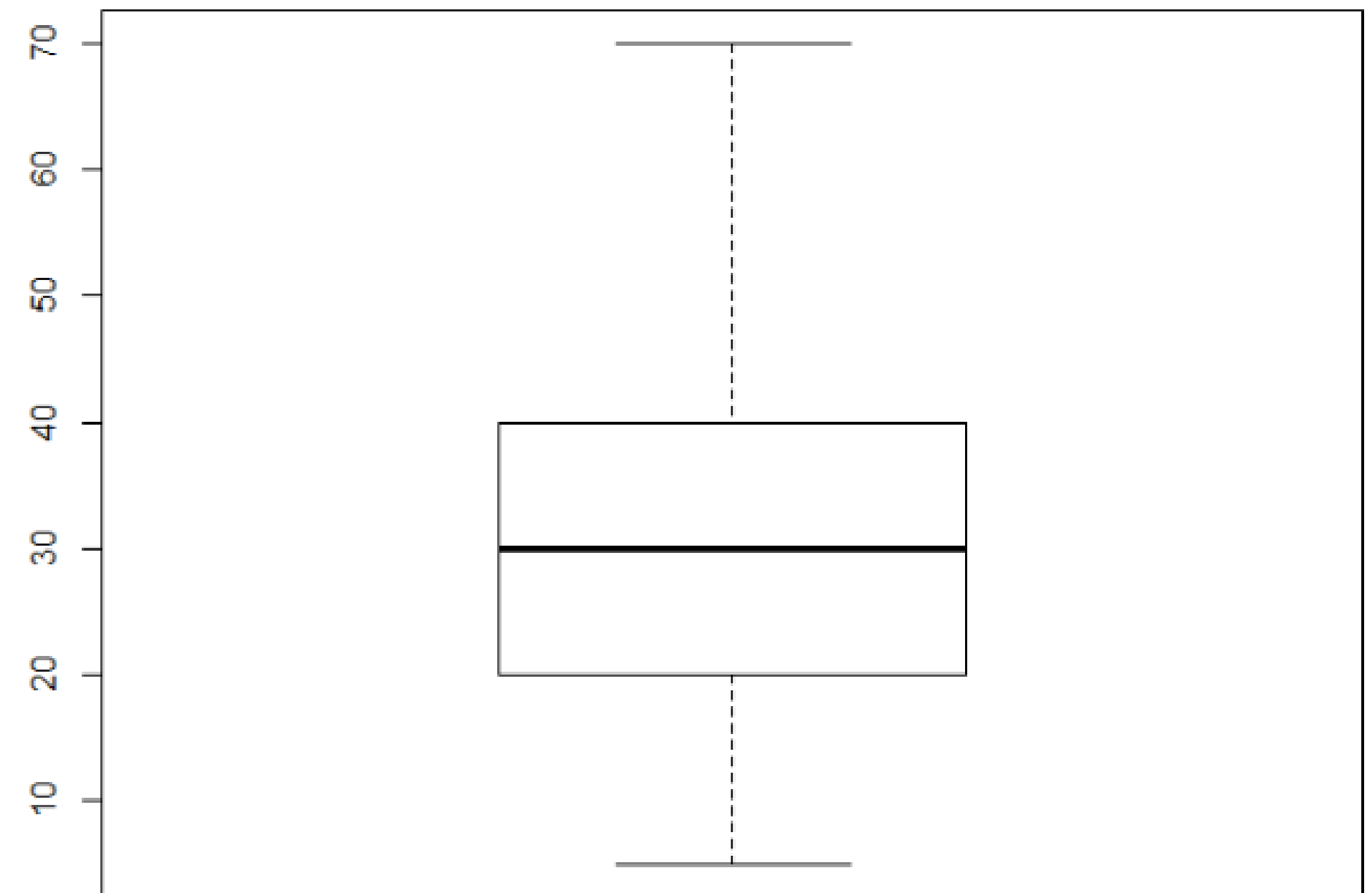
```
# storeB 상자그림 그리기  
boxplot(storeB)
```

```
> # storeB 상자그림 그리기  
> boxplot(storeB)
```



< 이상치 제거한 B식당의 배달시간 >

```
# 이상값 제외하고 상자그림 그리기  
# 600 미만 배달시간만 감안  
storeB<-storeB[storeB<600]  
boxplot(storeB)
```



< A식당 VS 이상치 제거한 B식당 배달시간 >

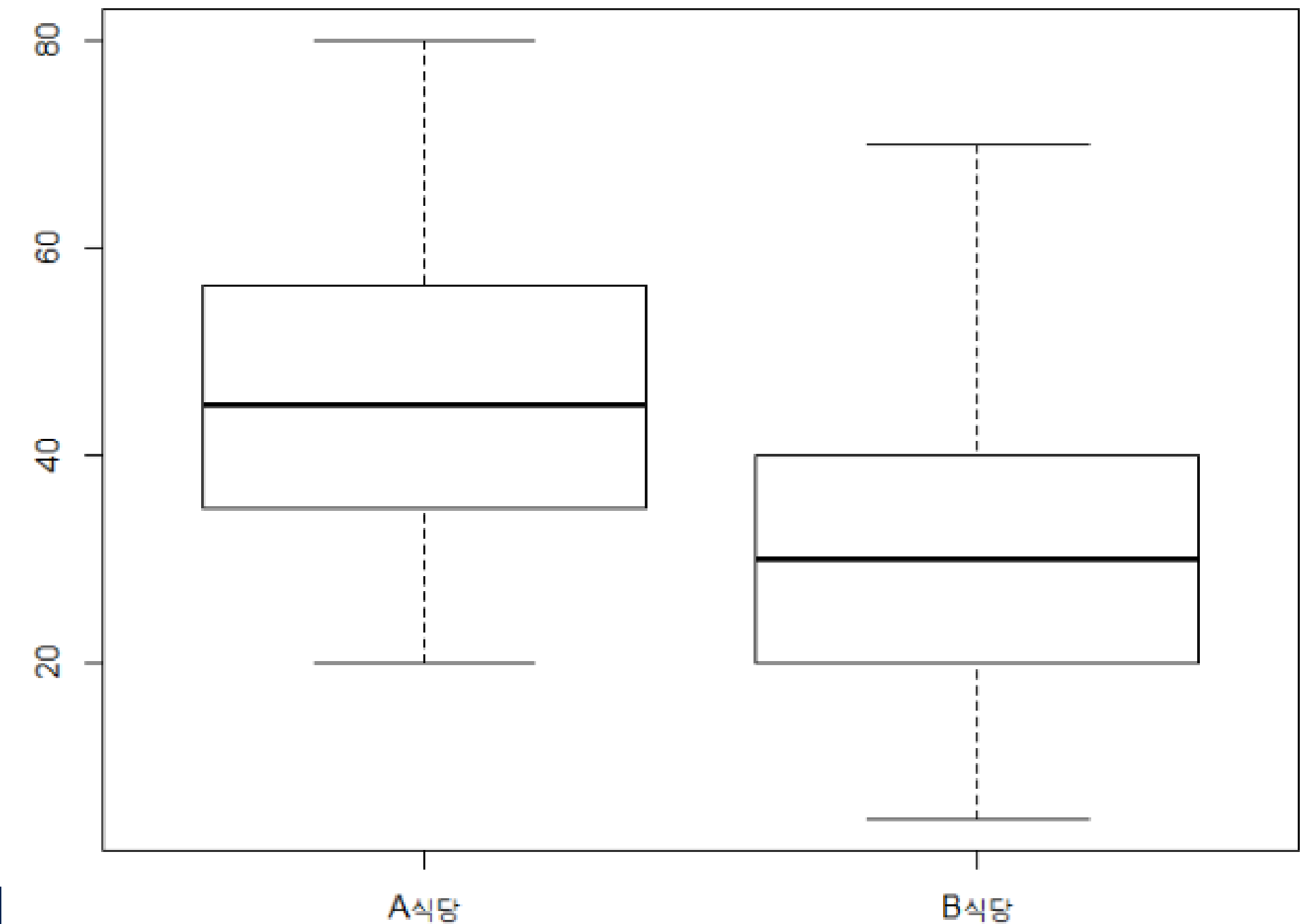
가로축에 이름지정 (두 개 이상의 상자그림 비교할 때 유용)

boxplot(데이터1, 데이터2,...,names=c("데이터1이름","데이터2이름",...))

```
# A식당 배달시간
storeA

# B식당 배달시간
storeB

# 상자그림 그리기
# names 입력 항목을 통해 가로 축의 이름 지정 가능
boxplot(storeA,storeB,names=c("A식당","B식당"))
```



```
# A식당 배달시간
storeA
[1] 20 21 23 22 26 28 35 35 41 42 43 45 44 45 46 47 47 46 47 58 58 59 60 56 57 57 80
# B식당 배달시간
storeB
[1] 5 6 11 16 15 16 20 20 21 23 22 27 27 30 30 32 36 37 37 40 40 43 44 45 51 54 70
# 상자그림 그리기
# names 입력 항목을 통해 가로 축의 이름 지정 가능
boxplot(storeA,storeB,names=c("A식당","B식당"))
```

3 CHAPTER 3.4 points 함수

설명




















points() 함수 : 이미 생성된 그래프에 추가로 점을 그릴 수 있는 함수

points(x, pch, col, cex)

x : 표시하고 싶은 점의 위치를 지정한 벡터

pch : 점의 모양 (1:원, 2:삼각형... ?points 참조)

➔ 책에 잘못 나와있으니까 참고하세요!

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
																		

col : 점의 색깔

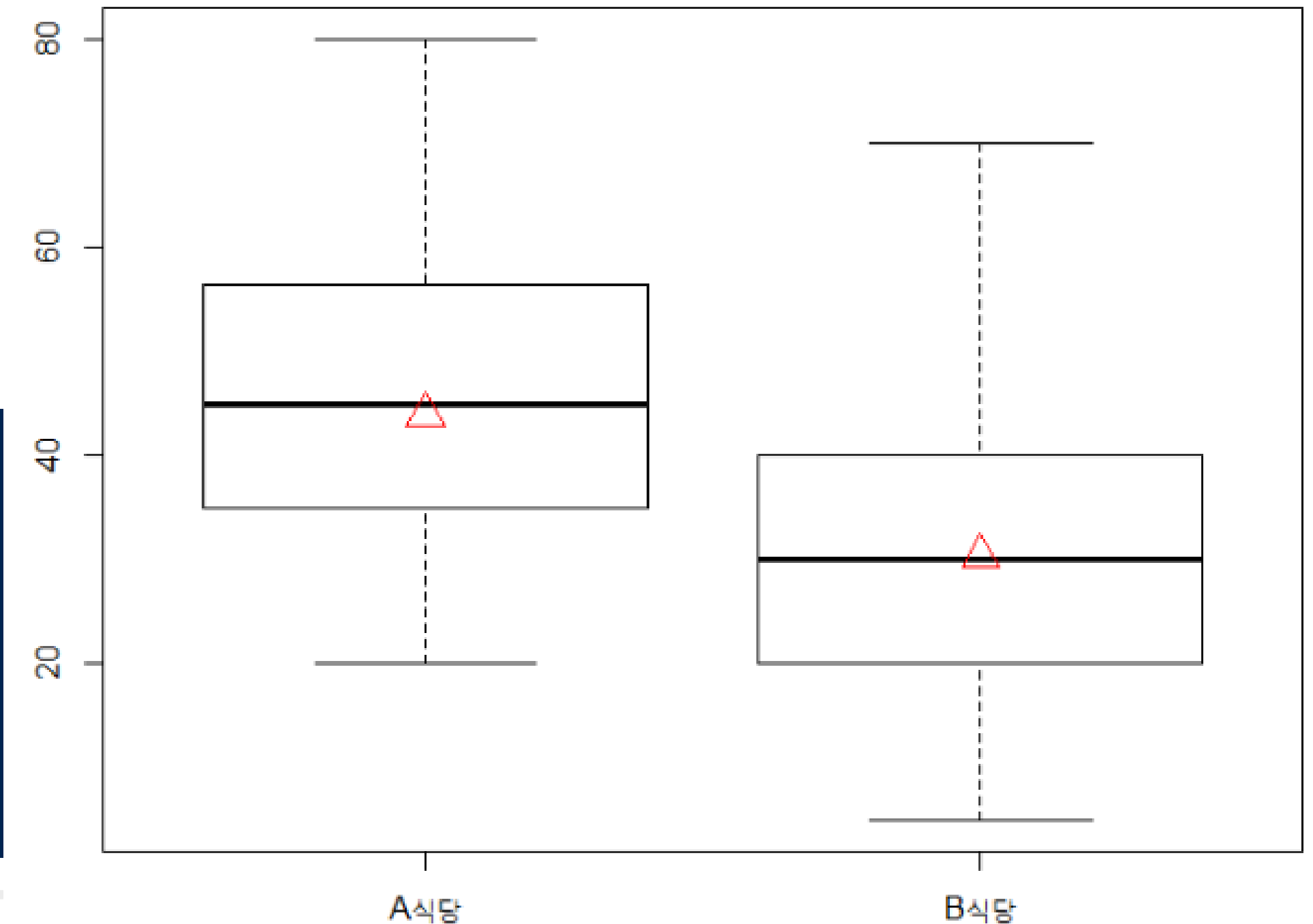
cex : 점의 크기

A식당과 B식당의 상자그림 그리기

```
boxplot(storeA,storeB,names=c("A식당","B식당"))
```

앞서 그려진 상자그림에 평균을 추가로 표기하기

```
points(c(mean(storeA), mean(storeB)),pch=2, col="red", cex=2)
```

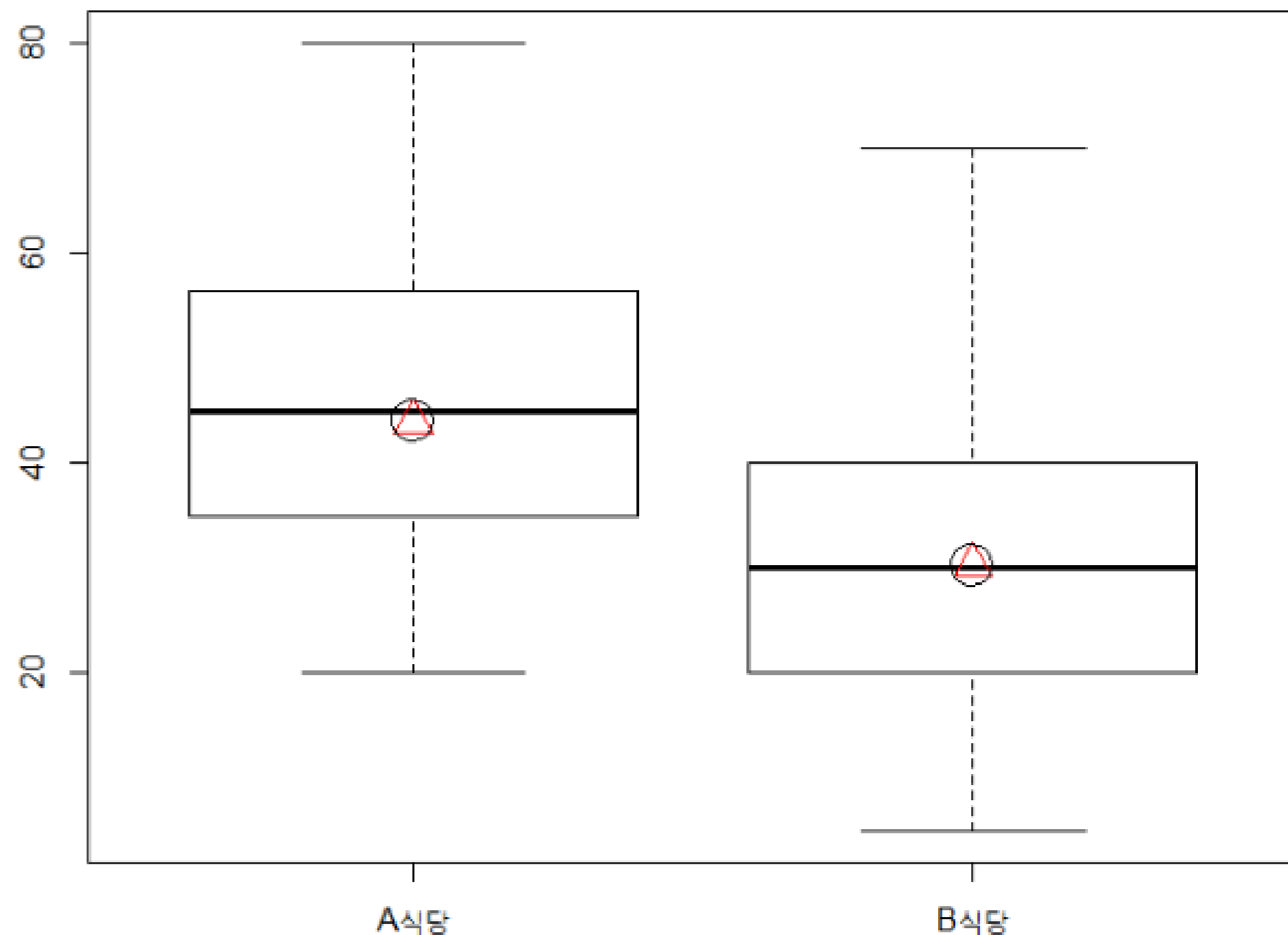


points함수는 앞의 그림에 덧 그리는 함수이기 때문에
points 함수를 쓰기 전에는 반드시 plot 함수와 같이 그림을 그리는 함수가 있어야 한다!

col 입력항목은 color code로도 설정 가능

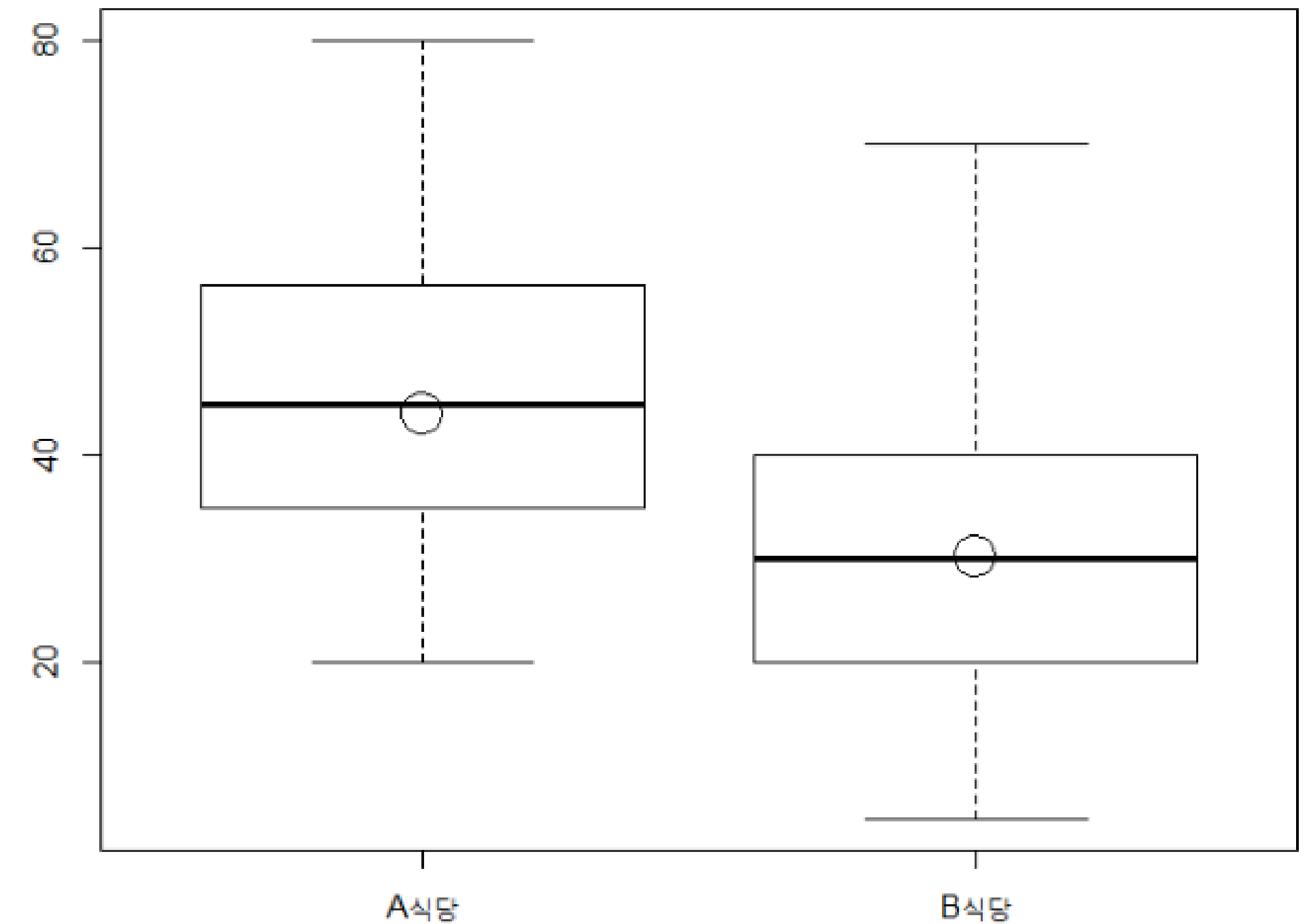
앞의 그래프에 이미 빨간 세모가 그려져 있었으므로
아래 명령문을 실행하면 세모 위에 동그라미가 그려짐

```
# color code 사용
points(c(mean(storeA), mean(storeB)), pch=1, col=1, cex=3)
```



왼쪽의 경우처럼 두 모양이 겹쳐지는게 싫다면
다시 A식당과 B식당의 상자그림을 그리고
실행해야 한다.

```
# A식당과 B식당의 상자그림 그리기
boxplot(storeA, storeB, names=c("A식당", "B식당"))
points(c(mean(storeA), mean(storeB)), pch=1, col=1, cex=3)
```



평균과 사분위수가 동일한 경우 → 그림 상자 비교

```
# 이상치가 제거된 B식당의 배달시간
storeB<-c(5,6,11,13,15,16,20,20,21,23,22,27,27,30,30,32,36,37,37,40,40,43,44,45,51,54,70)
# 평균
mean(storeB)
# 사분위수
quantile(storeB)
```

```
> # 이상치가 제거된 B식당의 배달시간
> storeB<-c(5,6,11,13,15,16,20,20,21,23,22,27,27,30,30,32,36,37,37,40,40,43,44,45,51,54,70)
>
> # 평균
> mean(storeB)
[1] 30.18519
>
> # 사분위수
> quantile(storeB)
0% 25% 50% 75% 100%
5 20 30 40 70
```

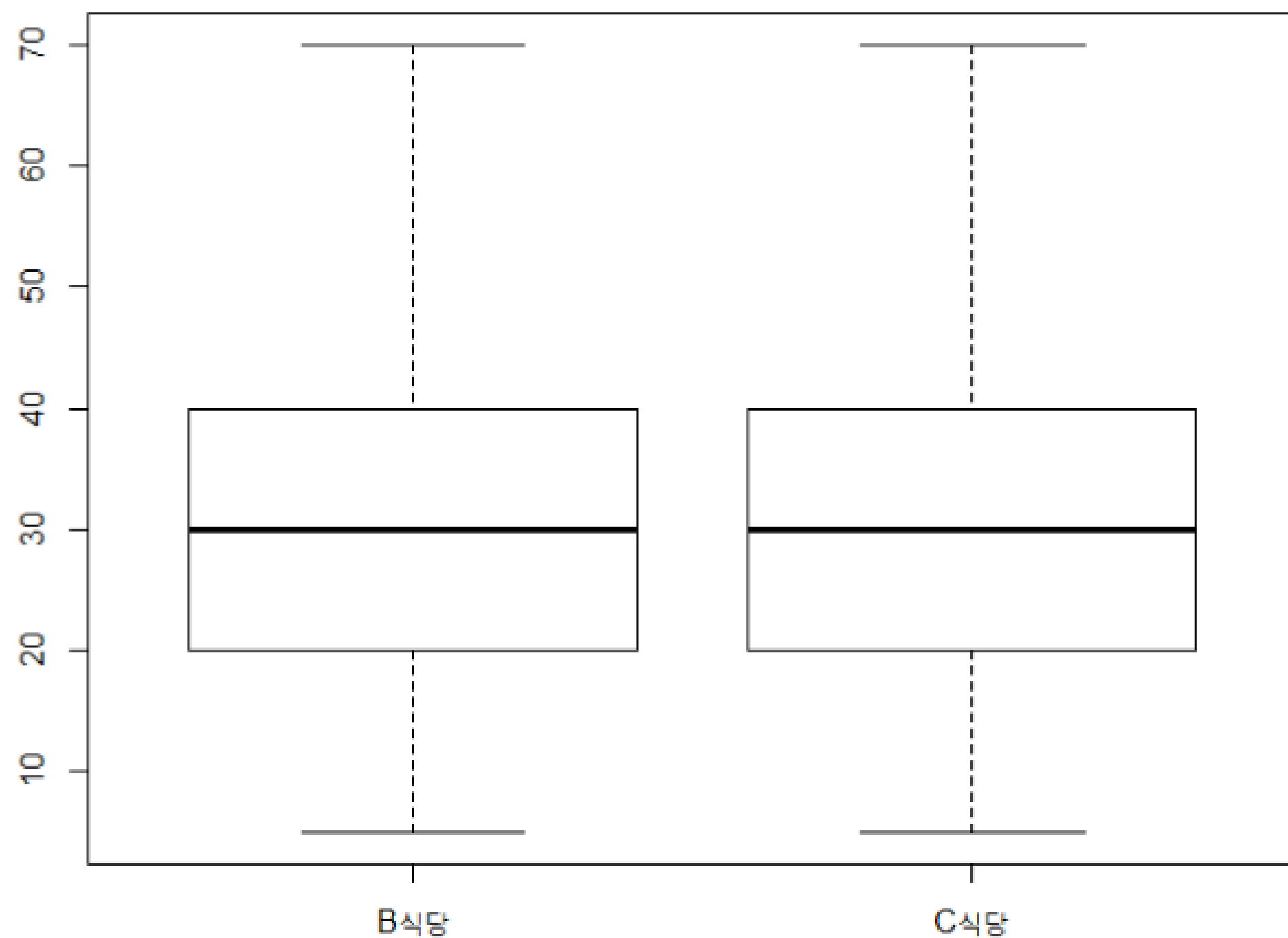
```
# C식당의 배달시간 벡터 생성
storeC<-c(5,5,5,12,10,11,20,20,20,20,20,21,20,30,32,31,31,31,36,40,40,51,61,51,61,61,70)
# 평균
mean(storeC)
# 사분위수
quantile(storeC)
```

```
> # C식당의 배달시간 벡터 생성
> storeC<-c(5,5,5,12,10,11,20,20,20,20,20,21,20,30,32,31,31,31,36,40,40,51,61,51,61,61,70)
>
> # 평균
> mean(storeC)
[1] 30.18519
>
> # 사분위수
> quantile(storeC)
0% 25% 50% 75% 100%
5 20 30 40 70
```

그림 상자도 동일한 경우

```
# B식당과 c식당의 boxplot 그리기  
boxplot(storeB, storeC, names=c("B식당", "c식당"))
```

```
> # B식당과 c식당의 boxplot 그리기  
> boxplot(storeB, storeC, names=c("B식당", "c식당"))
```



주의할 점

평균, 사분위수, 그림 상자가 동일하다고 해서 B식당과 C식당의 배달시간이 같다고 볼 수 없다.

그림 상자의 장단점

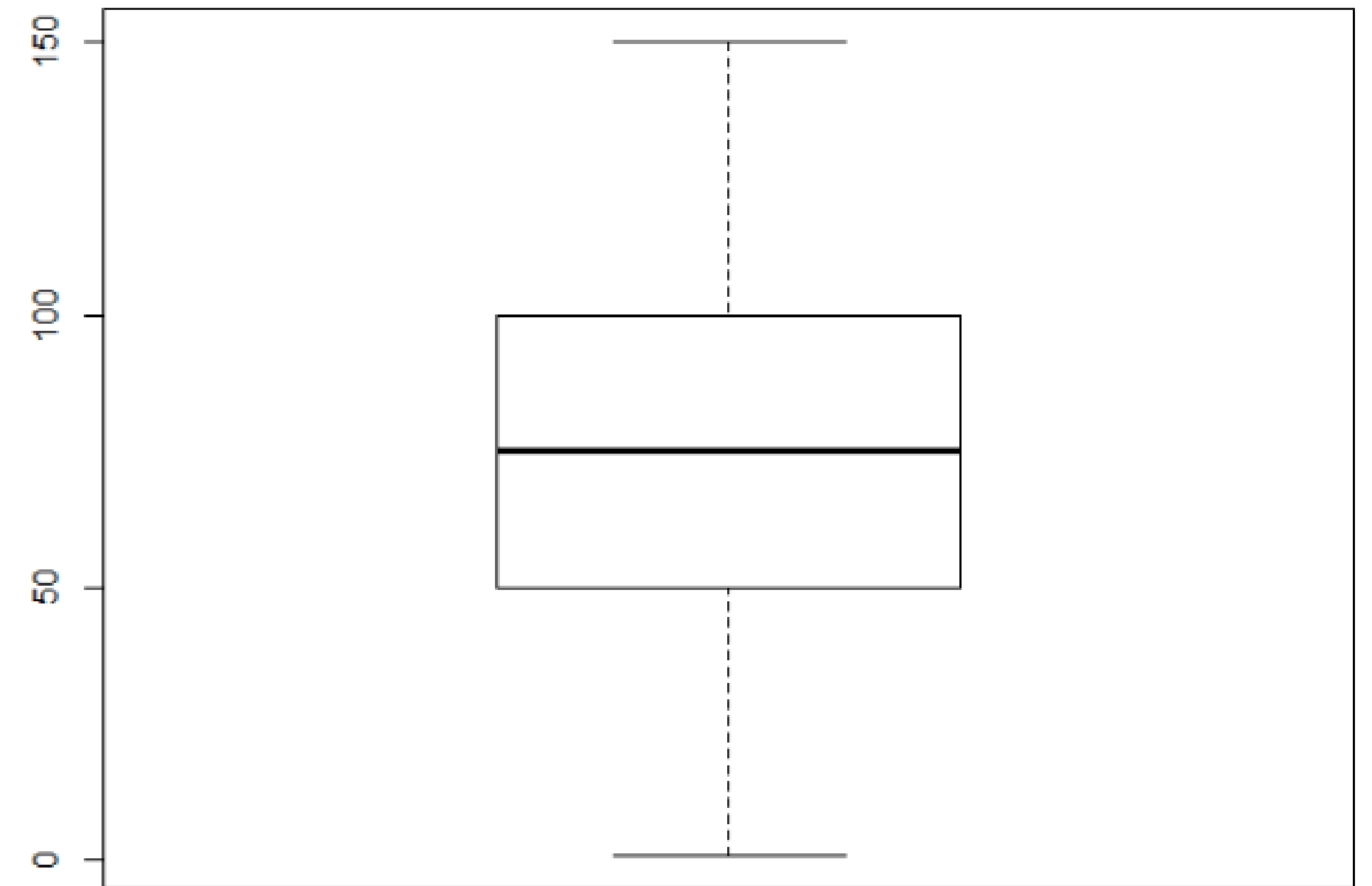
장점 : 분포의 다양한 특성 (중심, 퍼짐 정도)을 하나의 그래프로 표현

단점 : 사분위수 사이의 구간 데이터가 어떻게 구성되어 있는지 확인이 어려움

그림 상자의 단점을 보여주는 예시

```
# 1사분위와 3사분위에 집중된 데이터 만들기  
bpdatt <-c(1,50,50,50,100,100,100,150)  
# 상자그림 생성  
boxplot(bpdatt)
```

```
# 1사분위와 3사분위에 집중된 데이터 만들기  
bpdatt <-c(1,50,50,50,100,100,100,150)  
  
# 상자그림 생성  
boxplot(bpdatt)
```



특정 여러 구간에 데이터가 몰려 있는 경우,

상자그림만으로는 데이터의 분포를 충분히 파악할 수 없기 때문에

구간별 데이터의 분포도 함께 확인하는 것이 좋다.

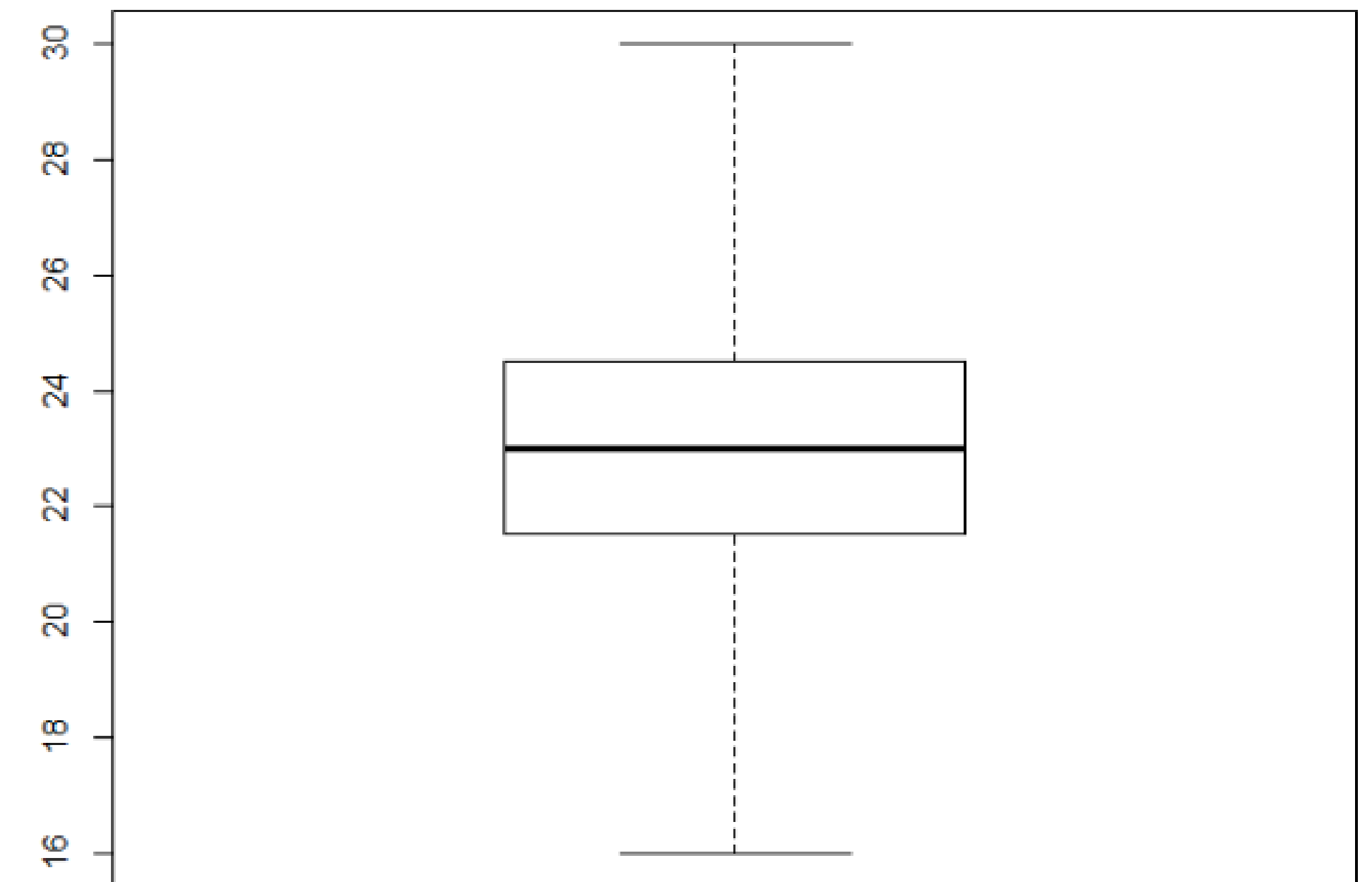
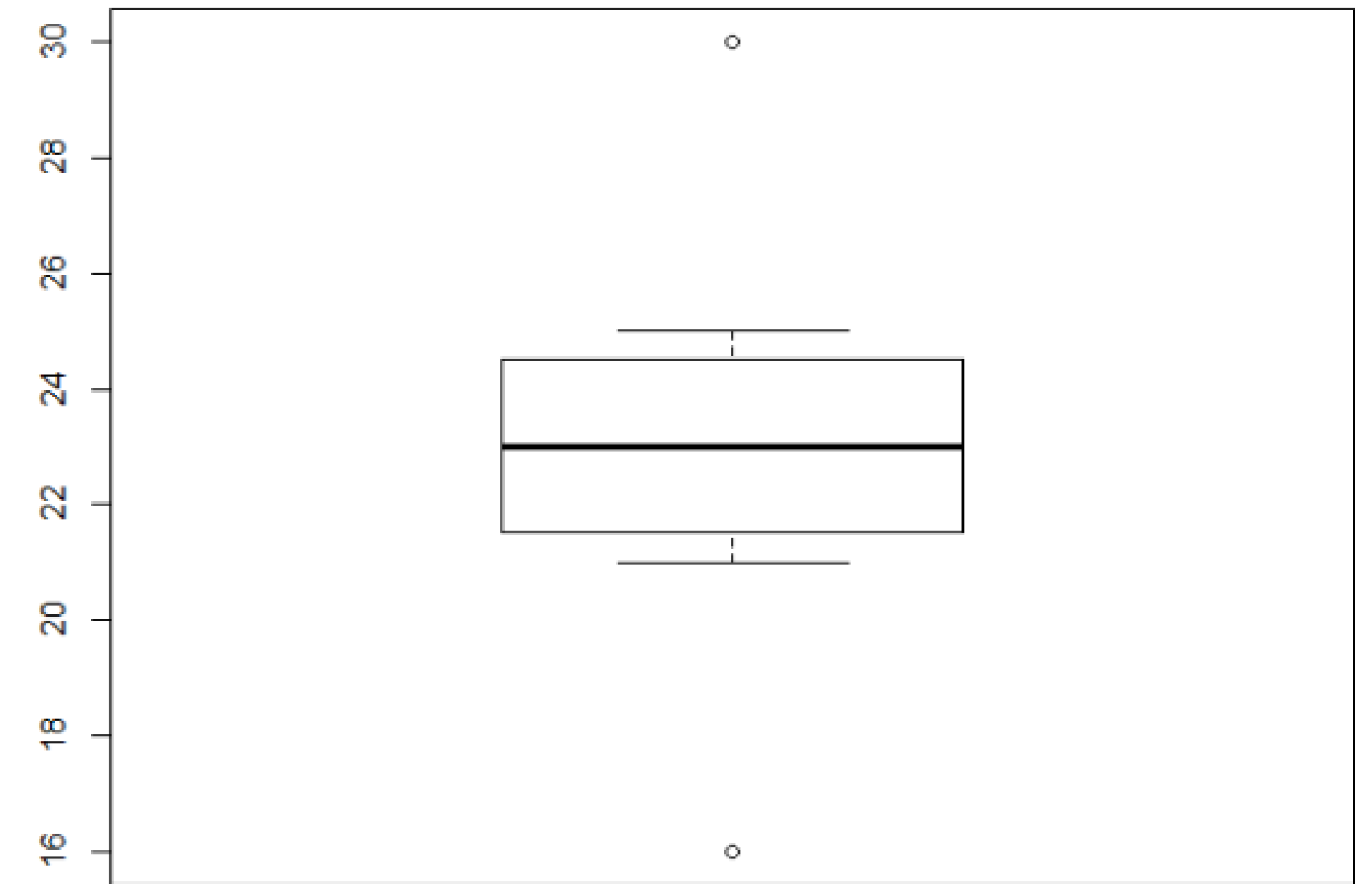
이상치 기준(정상치 기준)

기본 : IQR (3사분위수-1사분위수) * 1.5

특정 상황 : IQR (3사분위수-1사분위수) * IQR 대비 배수

`boxplot(데이터 이름, range=n)`

(n= IQR 대비 배수)



1. IQR*1.5

```
# 벡터 생성
boxNums<-c(16,21,22,23,24,25,30)

# 상자그림 그리기(기본값 : IQR*1.5)
boxplot(boxNums)
```

2. IQR*2

```
# IQR * 2
boxplot(boxNums, range=2)
```

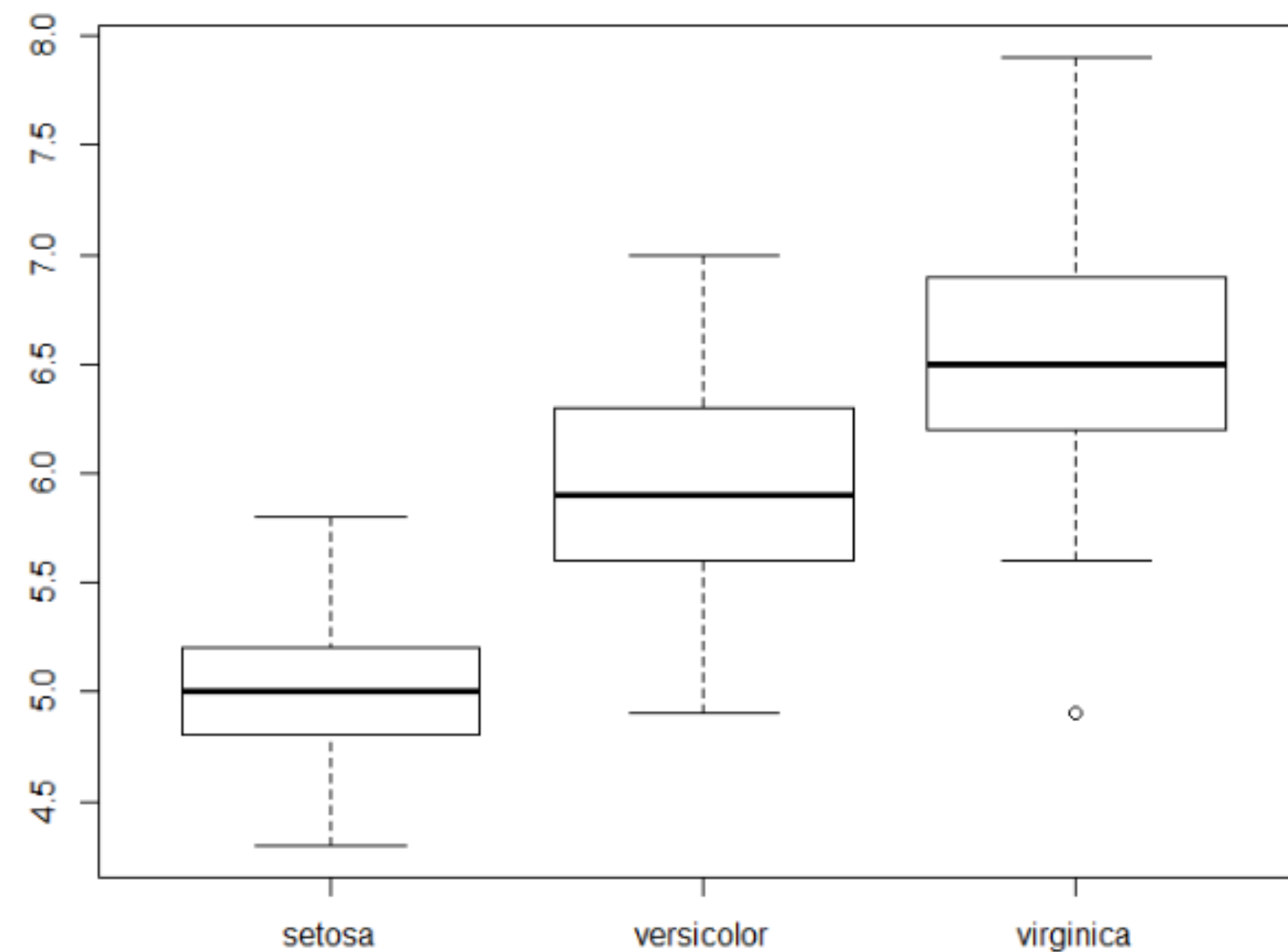
심화 학습 – boxplot(x~y)

boxplot(x~y) : y 그룹별로 x의 상자그림 그리기

예시 : 내장데이터 iris

iris의 Species 별로 Sepal.Length의 상자그림을 그리시오.

```
iris  
boxplot(iris$Sepal.Length~iris$Species)
```



CHAPTER 3.5

히스토그램

설명

히스토그램은 연속된 수를 구간별로 나누고 그 구간에 해당하는 빈도수를 표현한 그래프

hist(x, main, xlab, ylab) : 히스토그램을 그리기 위한 함수

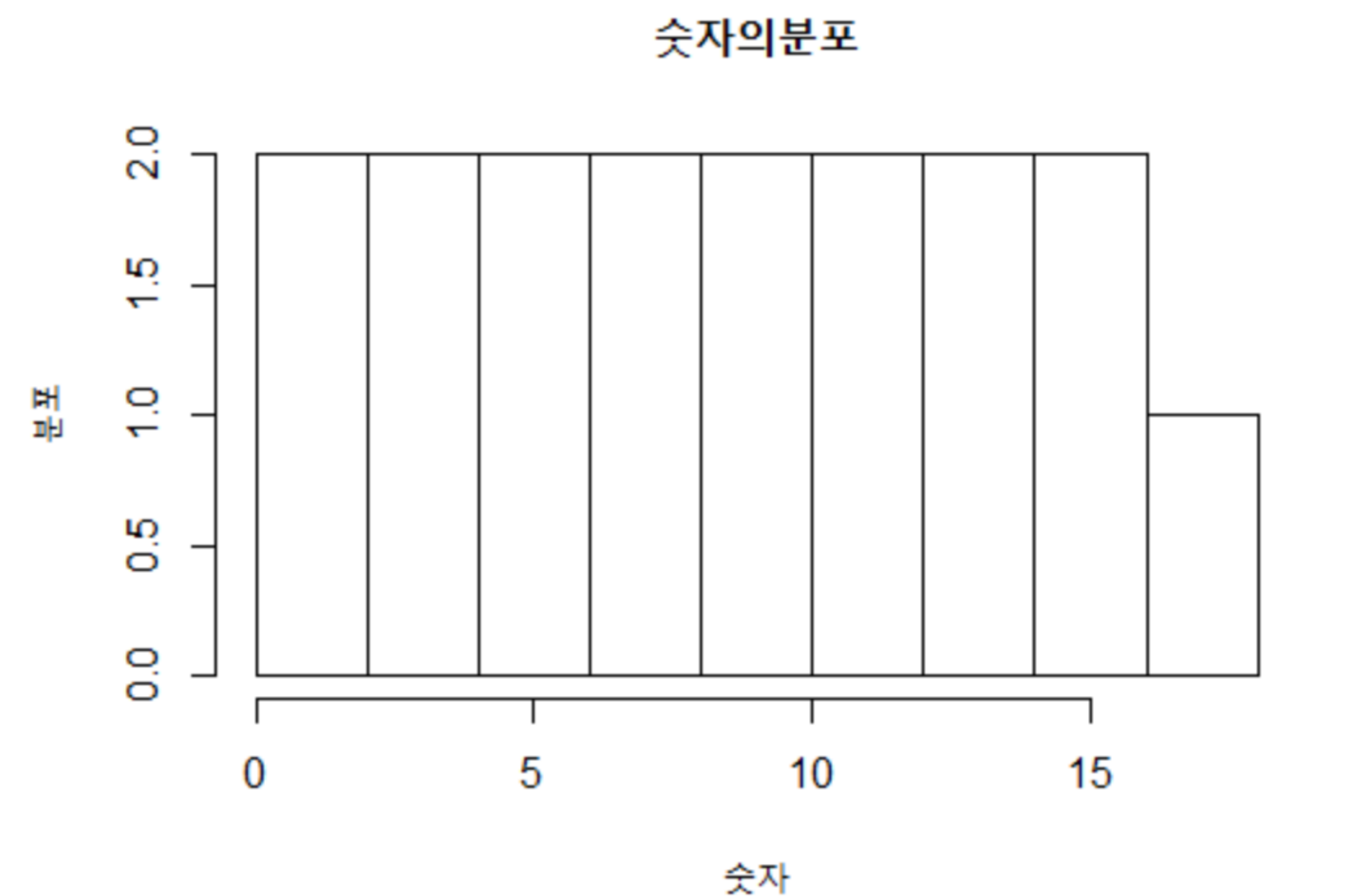
- x : 데이터 벡터의 이름
- main : 그래프 제목
- xlab : x축 이름
- ylab : y축 이름

CHAPTER 3.5

히스토그램

코드

```
1 # 벡터 선언
2 nums <- c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17)
3
4 # 숫자들 히스토그램 생성
5 hist(nums,main='숫자의분포',xlab='숫자',ylab='분포')
```



CHAPTER 3.5

히스토그램

설명

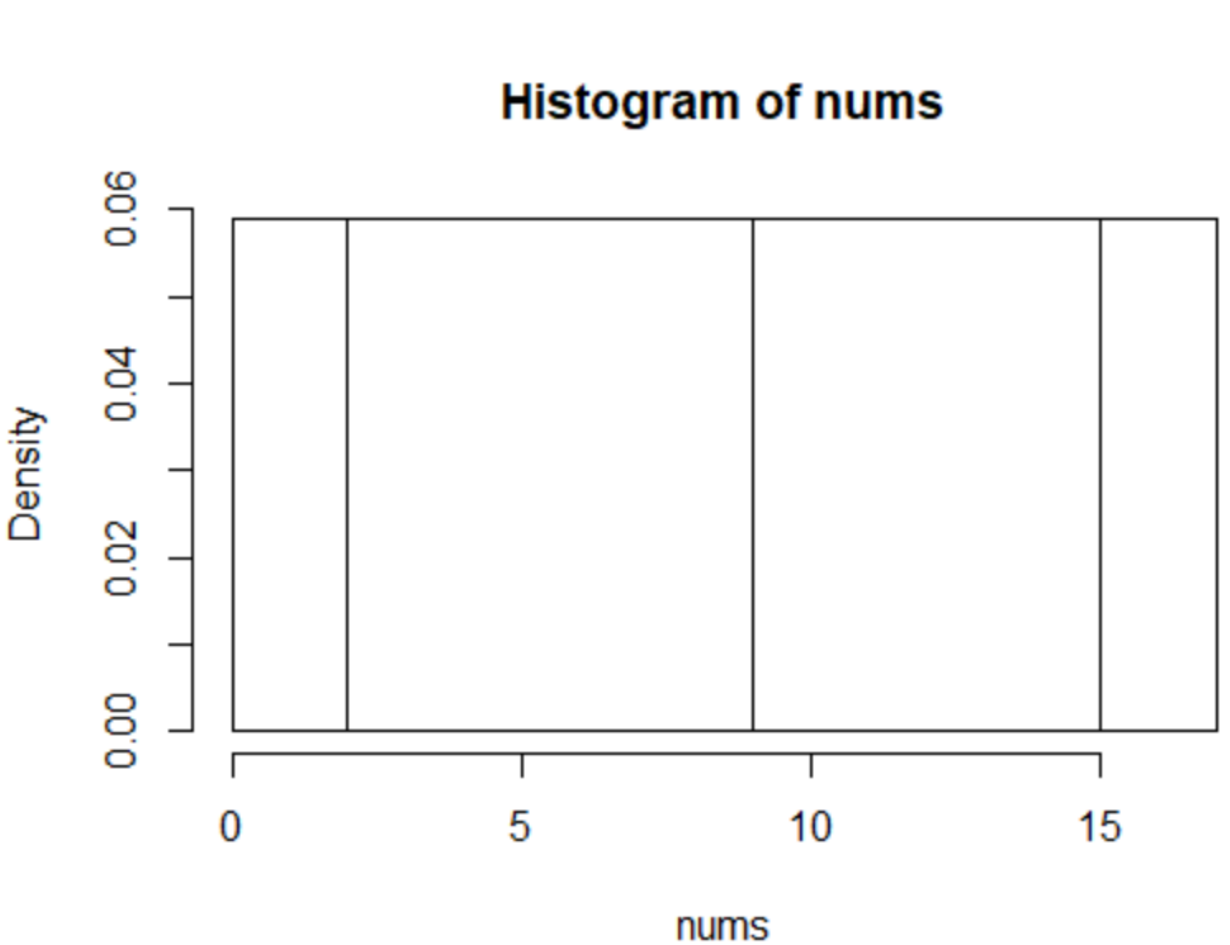
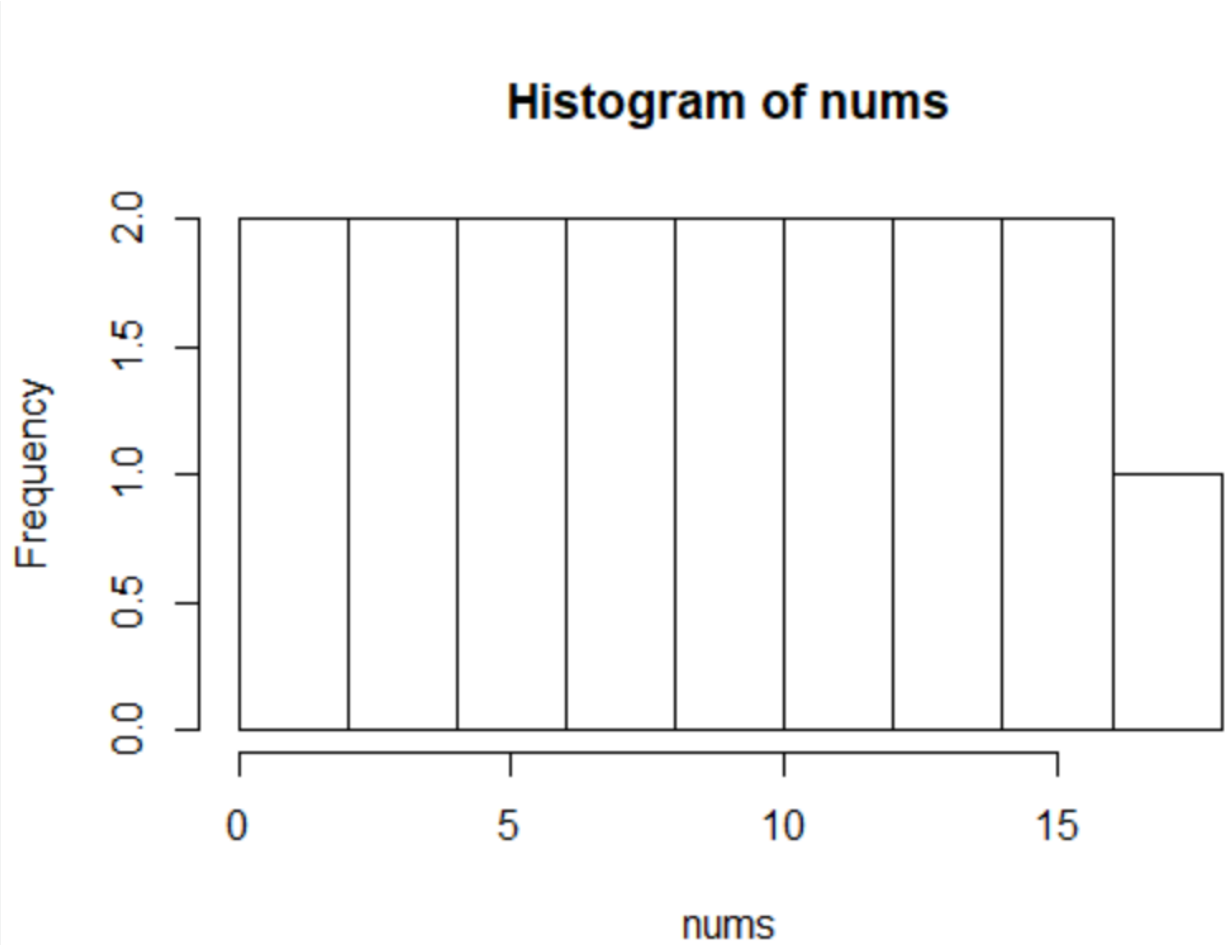
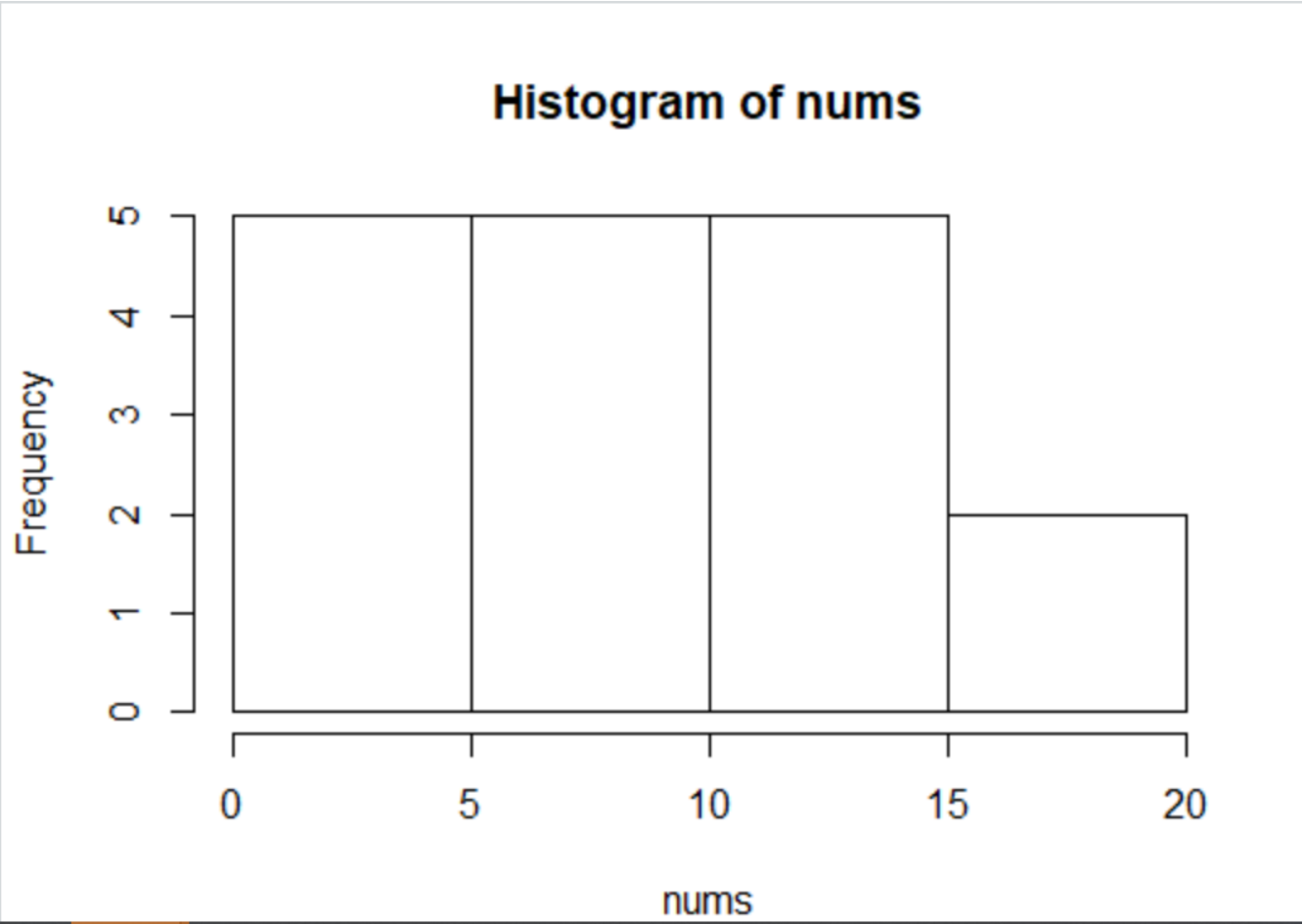
히스토그램의 간격을 조정하고 싶다면?

코드

```
1 # 벡터 선언
2 nums <- c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17)
3
4 # 4개 구간으로 보기
5 hist(nums,4)
6
7 # 6개 구간으로 보기
8 hist(nums,6)
9
10 # 구간 직접 설정해 주기
11 hist(nums,breaks=c(0,2,9,15,17))|
```

CHAPTER 3.5

히스토그램



BREAK TIME

가끔 실패하지 않는다면,
언제나 안이하게만 산다는 증거이다.

-우디 알렌

CHAPTER 3.6

분산과 표준편차

설명

분산 -평균값으로부터 떨어져 있는 정도를 나타내는 지표로서 각 데이터 요소들과 평균 간의 차이에 대한 평균

$$S^2 = \frac{1}{n-1} \sum_{i=1}^N (x_i - \bar{x})^2$$

<일반적인 산술평균 계산과의 차이점>

1. 평균의 차이를 모두 더하는게 아니라 차이를 '제곱'한 값을 더해 평균을 냄 -> **부호를 감안하지 않기 위함**
2. 전체에 대한 분산을 구하는 것이라면 총 데이터 수로 나누지만 현실에서는 전체 중 일부 데이터를 추출해 전체의 분산을 추정하게 됨. -> 오차 감안하기 위해 평균 계산 시, '**총 데이터 수-1**'로 나눔(자유도)

CHAPTER 3.6

분산과 표준편차

설명

분산 -평균값으로부터 떨어져 있는 정도를 나타내는 지표로서 각 데이터 요소들과 평균 간의 차이에 대한 평균

코드

```
var(storeB)
var(storeC)
-
> var(storeB)
[1] 239.5413
> var(storeC)
[1] 355.6952
```

분산은 **var**함수를 통해 산출

C식당이 B식당보다 분산이 큼

->"C식당의 배달시간이 B식당보다 더 들쭉날쭉함 "

CHAPTER 3.6

분산과 표준편차

설명

분산을 구할 때 제곱을 사용하다 보니 어느 정도 차이가 나는 지는 확인하기 쉽지 않음 -> 데이터의 단위로 보정 한 것이 **표준편차**

코드

```
sd(storeB)
sd(storeC)
> sd(storeB)
[1] 15.47712
> sd(storeC)
[1] 18.85988
```

표준편차는 **sd**함수를 통해 산출

$$S = \sqrt{\sum \frac{(X - \bar{X})^2}{N}} = \sqrt{\sum \frac{X^2}{N}} \quad N=\text{자유도}$$

“B식당은 ‘대략’15분~46분
사이에 배달이 온다”

“C식당은 ‘대략’11분~49분
사이에 배달이 온다”

평균배달시간이 30.18519분으로 같아도 표준편차로
흩어진 정도로 확인함으로써 **평균값의 신뢰성** 판단 가능

CHAPTER 3.7

범주형 데이터: 데이터 확인하기

설명

table 함수로 혈액형별 빈도수를 표로 표현하여 손쉽게 범주 별 개수를 산출할 수 있음

코드

#이슬반의 혈액형

```
bloodType<-c('A','B','A','AB','O','A','O','B','A','O',
              'O','B','O','A','AB','B','O','A','A','B')
```

#이슬반은 총 몇 명인가?(벡터 길이 확인)

```
length(bloodType)
```

```
> length(bloodType)
```

```
[1] 20
```

#혈액형별 건수 보기

```
table(bloodType)
```

bloodType			
A	AB	B	O
7	2	5	6

```
> model_table<-table(bloodType)
```

```
> prop.table(model_table)
```

```
bloodType
```

A	AB	B	O
0.35	0.10	0.25	0.30

table 함수로 생성된 표에 대해
prop.table() 명령을 활용해 비율
계산가능

CHAPTER 3.7

범주형 데이터: 데이터 확인하기

설명

table 함수에서 나온 결과값으로 범주명만 조회하거나 별도 계산도 할 수 있음

코드

```
#table 함수의 결과 객체 생성
table_bloodType<-table(bloodType)

#결과 조회
table_bloodType

#범주 확인 하기
names(table_bloodType)

#전체 합계 구하기
sum(table_bloodType)
```

```
> #table 함수의 결과 객체 생성
> table_bloodType<-table(bloodType)
> #결과 조회
> table_bloodType
bloodType
  A AB  B  O
  7  2  5  6
> #범주 확인 하기
> names(table_bloodType)
[1] "A"  "AB" "B"  "O"
> #전체 합계 구하기
> sum(table_bloodType)
[1] 20
```


CHAPTER 3.7

범주형 데이터: 데이터 확인하기

설명

addmargins 함수를 통해 합계를 붙여 함께 볼 수 있음

코드

#총계 붙여보기

```
addmargins(table(bloodType))
```

> #총계 붙여보기

```
> addmargins(table(bloodType))
```

bloodType

A	AB	B	O	Sum
7	2	5	6	20

6

CHAPTER 3.7

범주형 데이터: 파이차트

설명

파이차트 – 동그란 원에 범주 별 비율로 나눠 표현하는 그래프

table 함수를 통해 나온 결과는 **pie**함수를 통해 **파이차트로 표현 가능**

pie(x, labels, col, lty, main)

입력항목	설명
x	데이터 벡터
labels	범주명을 지정한 벡터
col	각 범주 별 색상을 지정한 벡터
lty	파이차트의 선 종류 (0=blank, 1=solid, 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash)
main	차트제목

CHAPTER 3.7

범주형 데이터: 파이차트

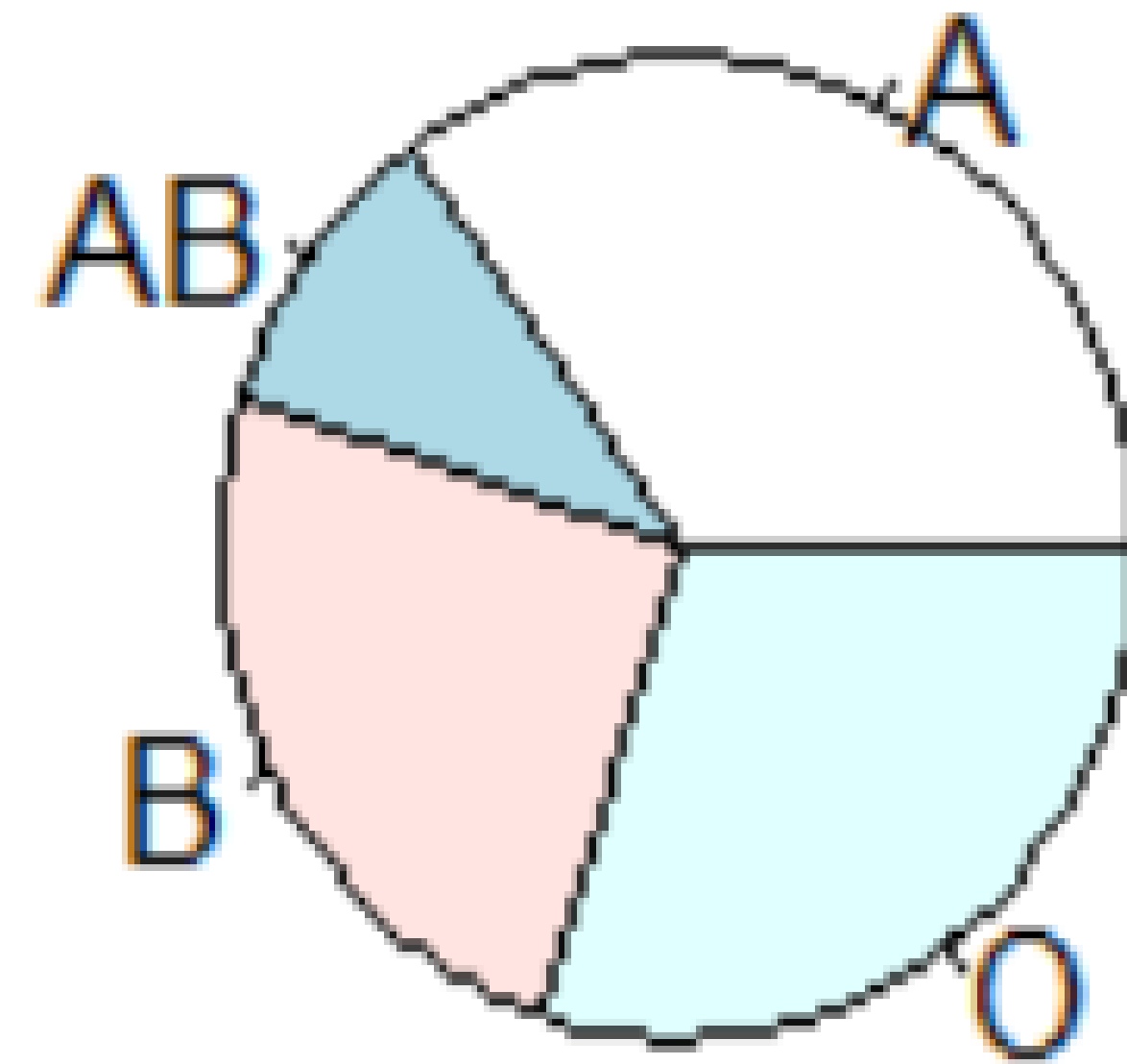
설명

파이차트 – 동그란 원에 범주 별 비율로 나눠 표현하는 그래프

코드

table 함수를 통해 나온 결과는 **pie** 함수를 통해 파이차트로 표현 가능
pie(x, labels, col, lty, main)

```
# 파이차트 생성  
pie(x=table(bloodType))
```



<이슬반의 혈액형 비중을 표현한 파이차트>

CHAPTER 3.7

범주형 데이터: 파이차트

설명

파이차트 – 동그란 원에 범주 별 비율로 나눠 표현하는 그래프

코드

#라벨명, 색상, 선 종류, 차트제목지정

```
pie(x=table(bloodType)
    ,labels=c("A형","AB형","B형","O형")
    ,col=c("chocolate1","chartreuse2","darkgoldenrod1","darkorchid1")
    ,lty=2
    ,main="이슬반 혈액형 분포")
```

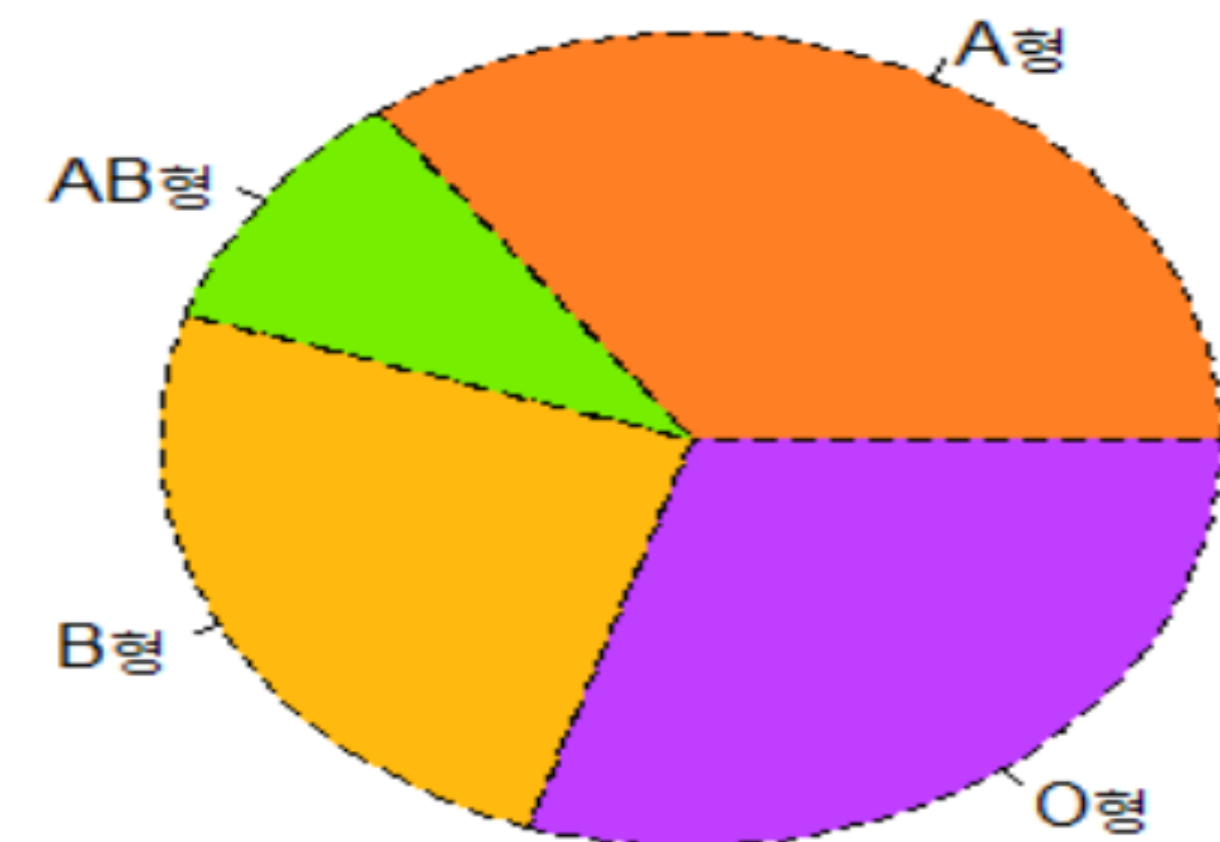
<장점>

단순하고 이해하기 쉬움

<단점>

범주가 많거나 범주 간 비중 차이가 크지
않은 경우 효과적으로 표현하기 쉽지
않음

이슬반 혈액형 분포



<부가 정보가 추가된 파이차트>

CHAPTER 3.7

범주형 데이터: 파이차트

설명

R의 색상지원 1) 색상명으로 지정

코드

#R에서 지원하는 색상명
colors()

```
> colors()
[1] "white"
[3] "antiquewhite"
[5] "antiquewhite2"
[7] "antiquewhite4"
[9] "aquamarine1"
[11] "aquamarine3"
...
중략
...
[653] "yellow1"
[655] "yellow3"
[657] "yellowgreen"
"aliceblue"
"antiquewhite1"
"antiquewhite3"
"aquamarine"
"aquamarine2"
"aquamarine4"
"yellow2"
"yellow4"
```

6 CHAPTER 3.7

범주형 데이터: 파이차트

설명

R의 색상지원 1) 색상명으로 지정

코드

#R에서 지원하는 색상명
colors()

색상 명을 알더라도 어떤
색깔인지 모를 수 있으므로
R 색상표를 확인하면 됨

Named Colours

Brown	Dark Red	Firebrick	Gainsboro
Indian Red	Light Coral	Red	Rosy Brown
Snow	White Smoke	Misty Rose	Salmon
Tomato	Dark Salmon	Coral	Orange Red
Light Salmon	Sienna	Sea Shell	Chocolate
Saddle Brown	Sandy Brown	Peach Puff	Peru
Linen	Bisque	Dark Orange	Burlywood
Antique White	Tan	Navajo White	Blanched Almond
Papaya Whip	Moccasin	Orange	Wheat
Old Lace	Floral White	Dark Goldenrod	Goldenrod
Corn Silk	Light Goldenrod	Gold	Lemon Chiffon
Khaki	Pale Goldenrod	Dark Khaki	Beige
Ivory	Light Goldenrod Yellow	Light Yellow	Yellow
Olive Drab	Yellow Green	Dark Olive Green	Green Yellow
Chartreuse	Lawn Green	Dark Green	Dark Sea Green
Forest Green	Green	Honeydew	Light Green
Lime Green	Pale Green	Sea Green	Medium Sea Green
Spring Green	Mint Cream	Medium Spring Green	Medium Aquamarine
Aquamarine	Turquoise	Light Sea Green	Medium Turquoise
Azure	Cyan	Dark Cyan	Dark Slate Gray
Light Cyan	Pale Turquoise	Dark Turquoise	Cadet Blue
Powder Blue	Light Blue	Deep Sky Blue	Sky Blue
Light Sky Blue	Steel Blue	Alice Blue	Dodger Blue
Light Slate Gray	Slate Gray	Light Steel Blue	Comflower Blue
Royal Blue	Blue	Dark Blue	Ghost White
Lavender	Medium Blue	Midnight Blue	Navy

Named Colours

Navy Blue	Slate Blue	Light Slate Blue	Dark Slate Blue
Medium Slate Blue	Medium Purple	Blue Violet	Purple
Dark Orchid	Dark Violet	Medium Orchid	Dark Magenta
Magenta	Plum	Thistle	Violet
Orchid	Violet Red	Medium Violet Red	Deep Pink
Hot Pink	Maroon	Lavender Blush	Pale Violet Red
Pink	Light Pink		

Named Colour Shades

Brown 1	Brown 2	Brown 3	Brown 4
Fire Brick 1	Fire Brick 2	Fire Brick 3	Fire Brick 4
Indian Red 1	Indian Red 2	Indian Red 3	Indian Red 4
Red 1	Red 2	Red 3	Red 4
Rosy Brown 1	Rosy Brown 2	Rosy Brown 3	Rosy Brown 4
Snow 1	Snow 2	Snow 3	Snow 4
Misty Rose 1	Misty Rose 2	Misty Rose 3	Misty Rose 4
Salmon 1	Salmon 2	Salmon 3	Salmon 4
Tomato 1	Tomato 2	Tomato 3	Tomato 4
Coral 1	Coral 2	Coral 3	Coral 4
Orange Red 1	Orange Red 2	Orange Red 3	Orange Red 4
Light Salmon 1	Light Salmon 2	Light Salmon 3	Light Salmon 4
Sienna 1	Sienna 2	Sienna 3	Sienna 4
Sea Shell 1	Sea Shell 2	Sea Shell 3	Sea Shell 4
Chocolate 1	Chocolate 2	Chocolate 3	Chocolate 4
Peach Puff 1	Peach Puff 2	Peach Puff 3	Peach Puff 4
Bisque 1	Bisque 2	Bisque 3	Bisque 4
Dark Orange 1	Dark Orange 2	Dark Orange 3	Dark Orange 4
Burlywood 1	Burlywood 2	Burlywood 3	Burlywood 4
Antique White 1	Antique White 2	Antique White 3	Antique White 4
Tan 1	Tan 2	Tan 3	Tan 4
Navajo White 1	Navajo White 2	Navajo White 3	Navajo White 4
Orange 1	Orange 2	Orange 3	Orange 4
Wheat 1	Wheat 2	Wheat 3	Wheat 4
Dark Goldenrod 1	Dark Goldenrod 2	Dark Goldenrod 3	Dark Goldenrod 4
Goldenrod 1	Goldenrod 2	Goldenrod 3	Goldenrod 4

<http://r-diary.tistory.com/attachment/4921618c98febAZ.pdf>

CHAPTER 3.7

범주형 데이터: 파이차트

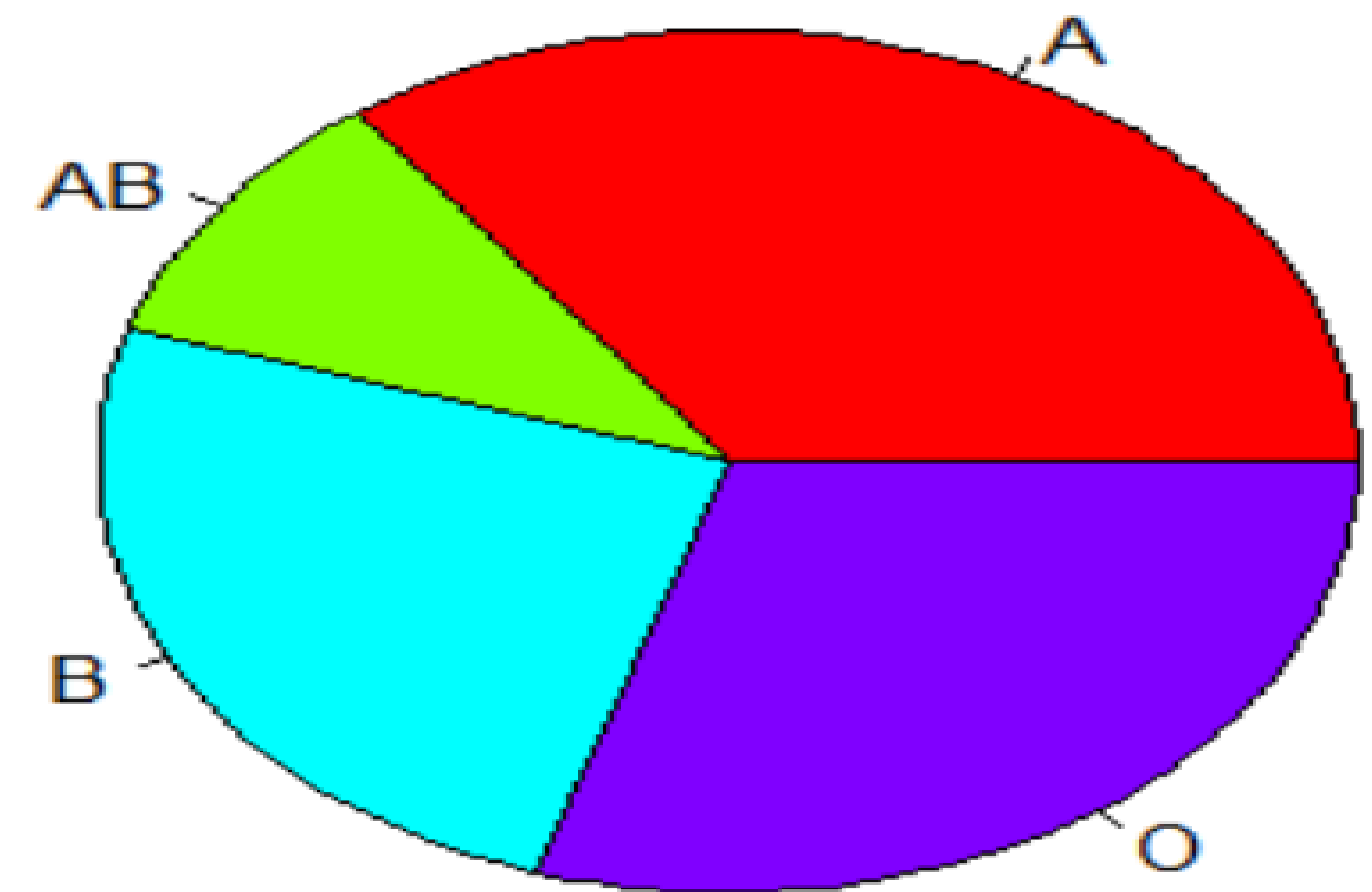
설명

R의 색상지원 2) 색상 팔레트로 지정

코드

heat.colors, rainbow, terrain.colors, topo.colors 등이 있음

```
# rainbow를 이용해 자동선정  
pie(x=table(bloodType), col=rainbow(4))  
|
```



CHAPTER 3.7

범주형 데이터: 막대차트

설명

막대 형식으로 데이터의 분포를 보여주는 시각화 방식

코드

barplot(데이터,부가 정보들)

입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정

*자주 쓰는 코드

CHAPTER 3.7

범주형 데이터: 막대차트

기본

table을 통해 빈도를 확인 후, 해당 table에 대한 막대차트를 표현.

코드

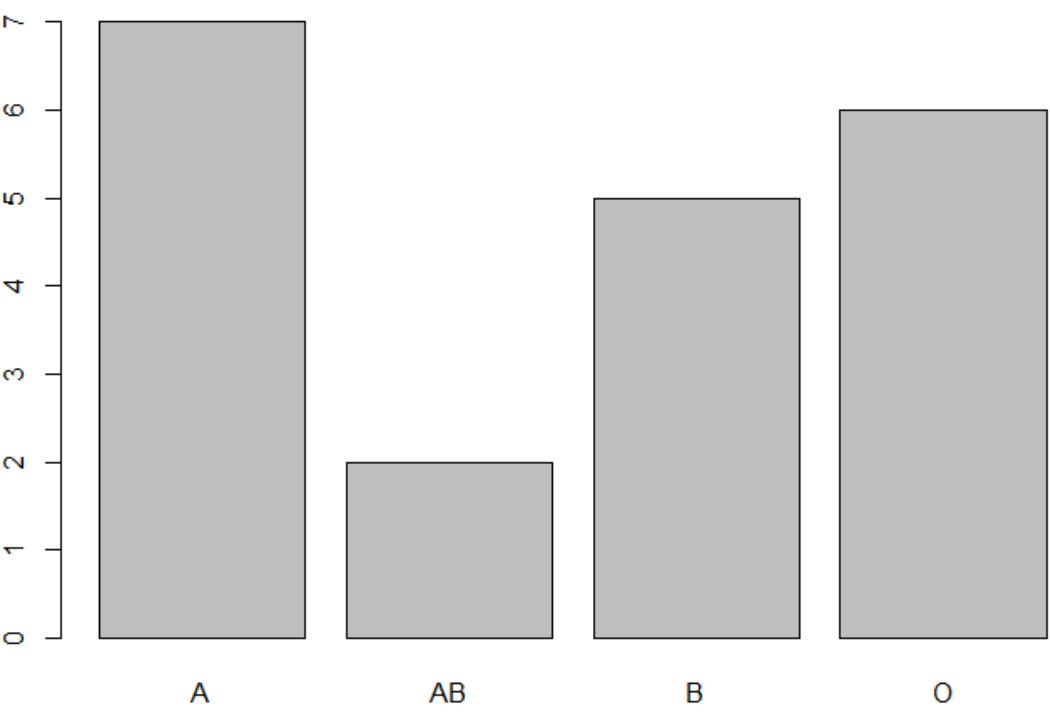
사용할 table(빈도표)

```
table(bloodType)
```

```
bloodType
  A AB  B  O
7  2  5  6
```

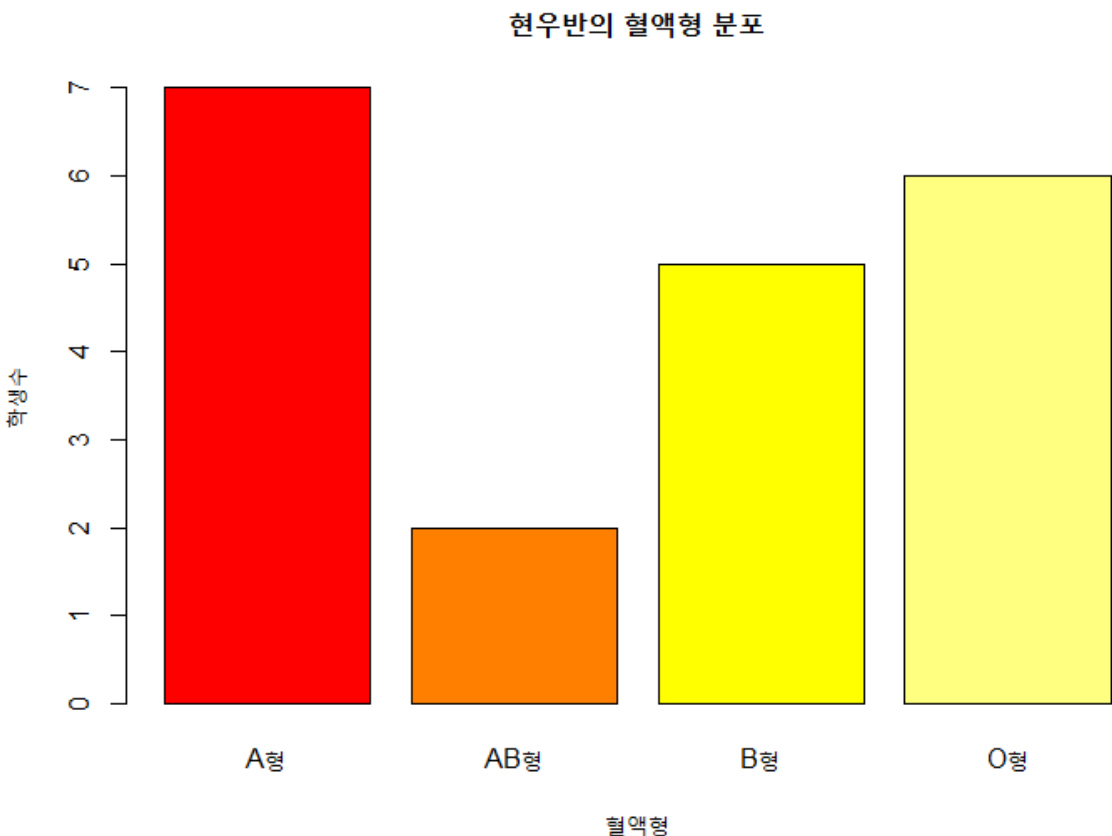
table에 대한 막대차트(기본)

```
barplot(table(bloodType))
```



table에 대한 막대차트(옵션조건 추가)

```
barplot(table(bloodType),
  names.arg=c("A형","AB형","B형","O형"),
  main="현우반의 혈액형 분포",
  xlab="혈액형",
  ylab="학생수",
  col=heat.colors(4))
```



입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정

CHAPTER 3.7

범주형 데이터: 막대차트

심화1

막대차트에 앞서 dataframe과 table 설정

코드

```
name <- c("1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20")
gender <- c('남','남','여','남','남','여','여','남','남','여','남','남','여','여','남','남','여','여','남','남','여')
bloodType <- c('A','B','A','AB','O','A','O','B','A','O','O','B','O','A','AB','B','O','A','A','B')

classDf <- data.frame(name,gender,bloodType)
str(classDf)
```

```
> str(classDf)
'data.frame':   20 obs. of  3 variables:
 $ name      : Factor w/ 20 levels "1","10","11",...: 1 12 14 15 16 17 18 19 20 2 ...
 $ gender    : Factor w/ 2 levels "남","여": 1 1 2 1 1 2 2 1 1 2 ...
 $ bloodType: Factor w/ 4 levels "A","AB","B","O": 1 3 1 2 4 1 4 3 1 4 ...
```

```
#classDf의 2열과 3열의 첫 6행 보기#
head(classDf[,c(2,3)])
```



	gender	bloodType
1	남	A
2	남	B
3	여	A
4	남	AB
5	남	O
6	여	A

입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정

CHAPTER 3.7

범주형 데이터: 막대차트

심화1

막대차트에 앞서 dataframe과 table 설정

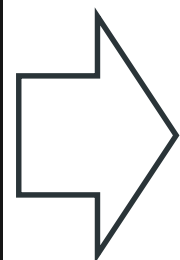
코드

```
name <- c("1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20")
gender <- c('남','남','여','남','남','여','여','남','남','여','남','남','여','여','남','남','여','여','남','남','여')
bloodType <- c('A','B','A','AB','O','A','O','B','A','O','O','B','O','A','AB','B','O','A','A','B')

classDf <- data.frame(name,gender,bloodType)
str(classDf)
```

```
> str(classDf)
'data.frame':   20 obs. of  3 variables:
 $ name      : Factor w/ 20 levels "1","10","11",...: 1 12 14 15 16 17 18 19 20 2 ...
 $ gender    : Factor w/ 2 levels "남","여": 1 1 2 1 1 2 2 1 1 2 ...
 $ bloodType: Factor w/ 4 levels "A","AB","B","O": 1 3 1 2 4 1 4 3 1 4 ...
```

```
#classDf 중 gender와 bloodType열만 선택해서 table객체 생성#
table(classDf[,c(2,3)])
```



	bloodType			
gender	A	AB	B	O
남	5	1	2	4
여	2	1	3	2

입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정

CHAPTER 3.7

범주형 데이터: 막대차트

심화1

막대차트에 앞서 dataframe과 table 설정

코드

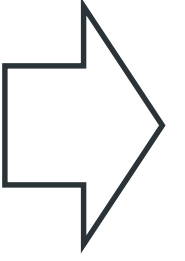
```
name <- c("1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20")
gender <- c('남','남','여','남','남','여','여','남','남','여','남','남','여','여','남','남','여','여','남','남','여')
bloodType <- c('A','B','A','AB','O','A','O','B','A','O','O','B','O','A','AB','B','O','A','A','B')

classDf <- data.frame(name,gender,bloodType)
str(classDf)
```

```
> str(classDf)
'data.frame':   20 obs. of  3 variables:
 $ name      : Factor w/ 20 levels "1","10","11",...: 1 12 14 15 16 17 18 19 20 2 ...
 $ gender    : Factor w/ 2 levels "남","여": 1 1 2 1 1 2 2 1 1 2 ...
 $ bloodType: Factor w/ 4 levels "A","AB","B","O": 1 3 1 2 4 1 4 3 1 4 ...
```

입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정

```
#addmargins 함수로 합계 행과 열 추가#
addmargins(table(classDf[,c(2,3)]))
```



	bloodType				
gender	A	AB	B	O	Sum
남	5	1	2	4	12
여	2	1	3	2	8
Sum	7	2	5	6	20

CHAPTER 3.7

범주형 데이터: 막대차트

심화2

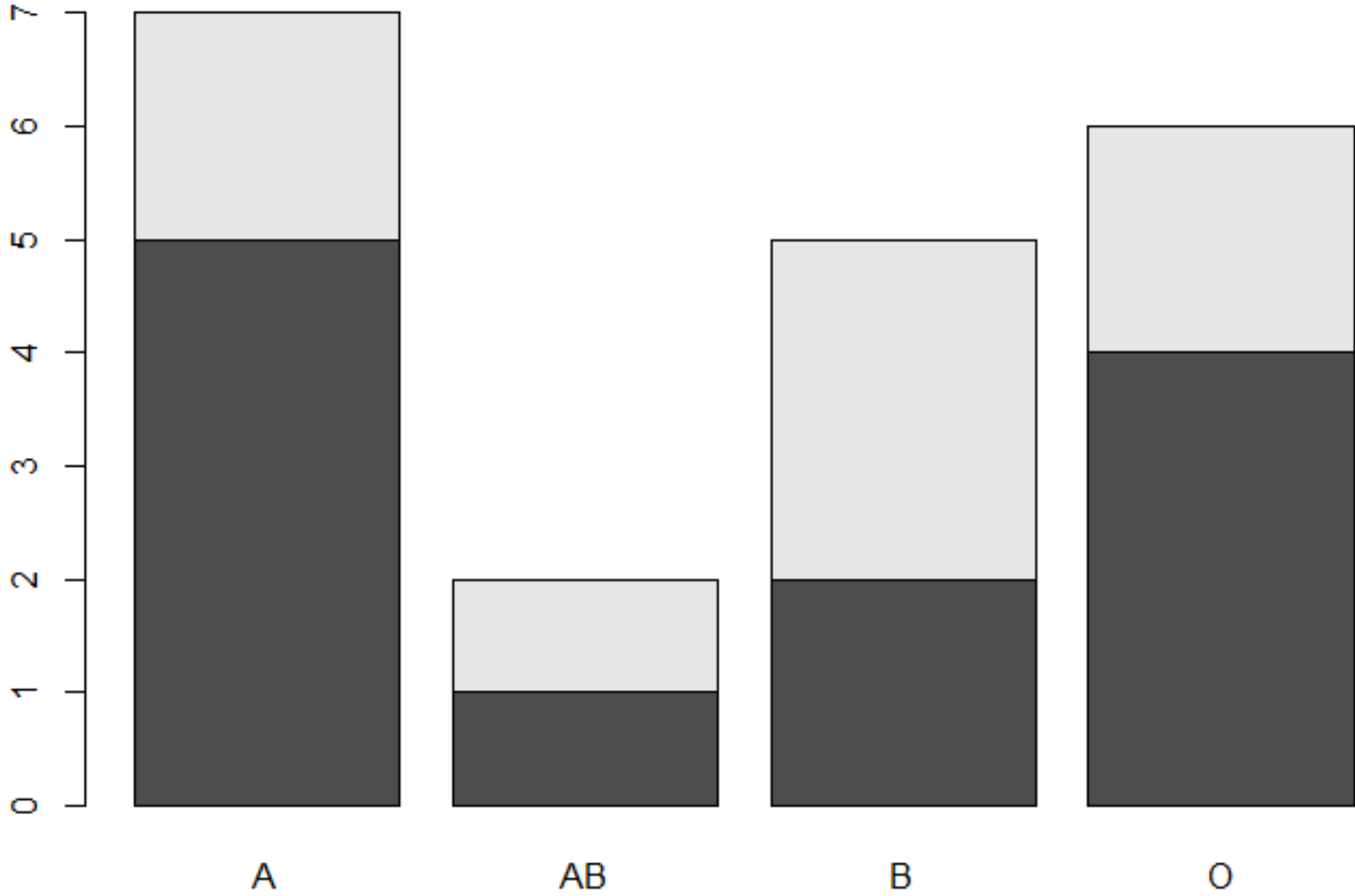
Table을 통해 다양한 방식의 막대차트 생성

코드

```
#table 함수의 결과와 객체 생성#
classTable <- table(classDf[,c(2,3)])

#막대차트를 활용한 성별과 혈액형별 분포 확인#
barplot(classTable)
```

입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정



CHAPTER 3.7

범주형 데이터: 막대차트

심화2

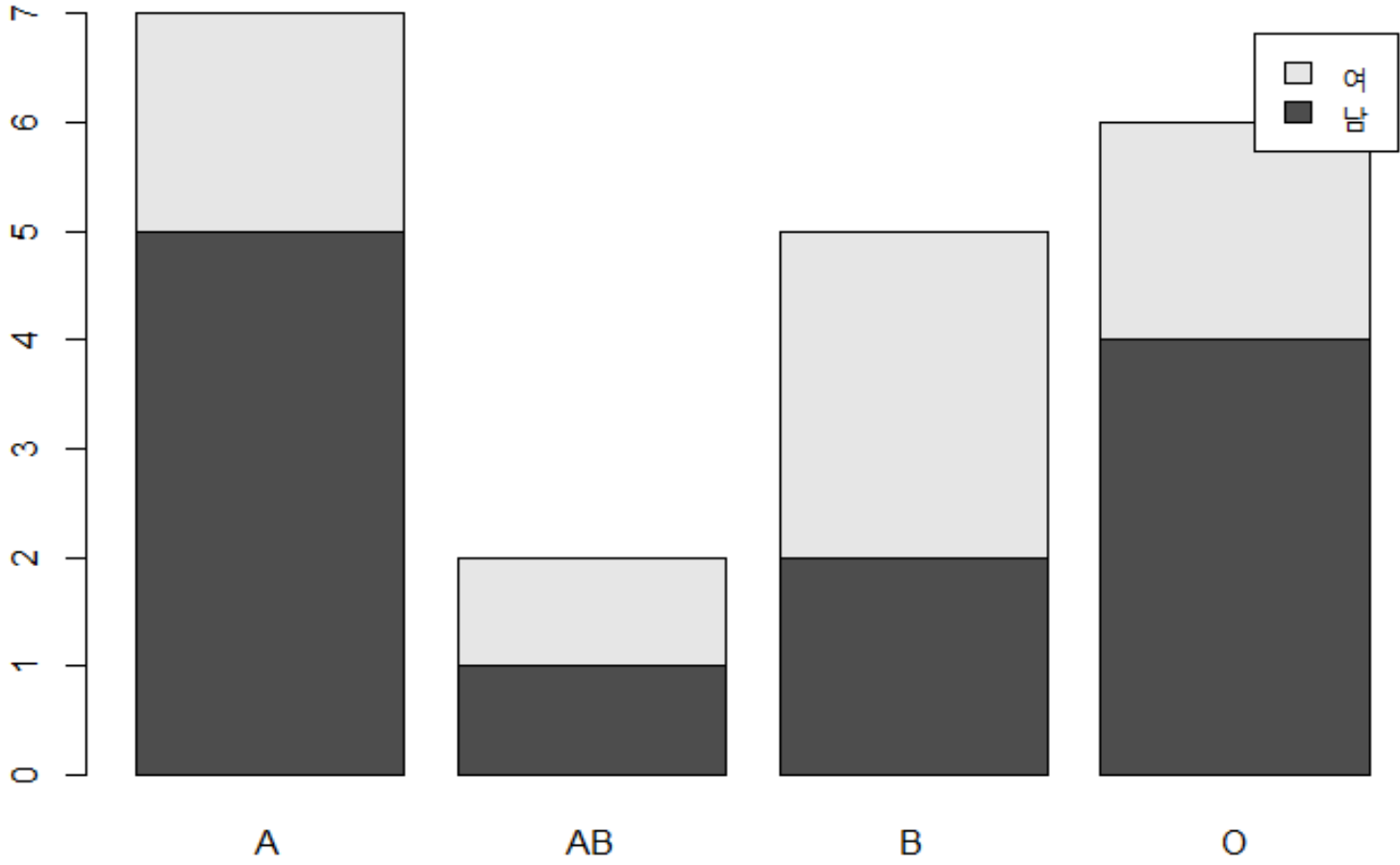
Table을 통해 다양한 방식의 막대차트 생성

입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정

코드

#범주형 추가#

```
barplot(classTable, legend=TRUE)
```



CHAPTER 3.7

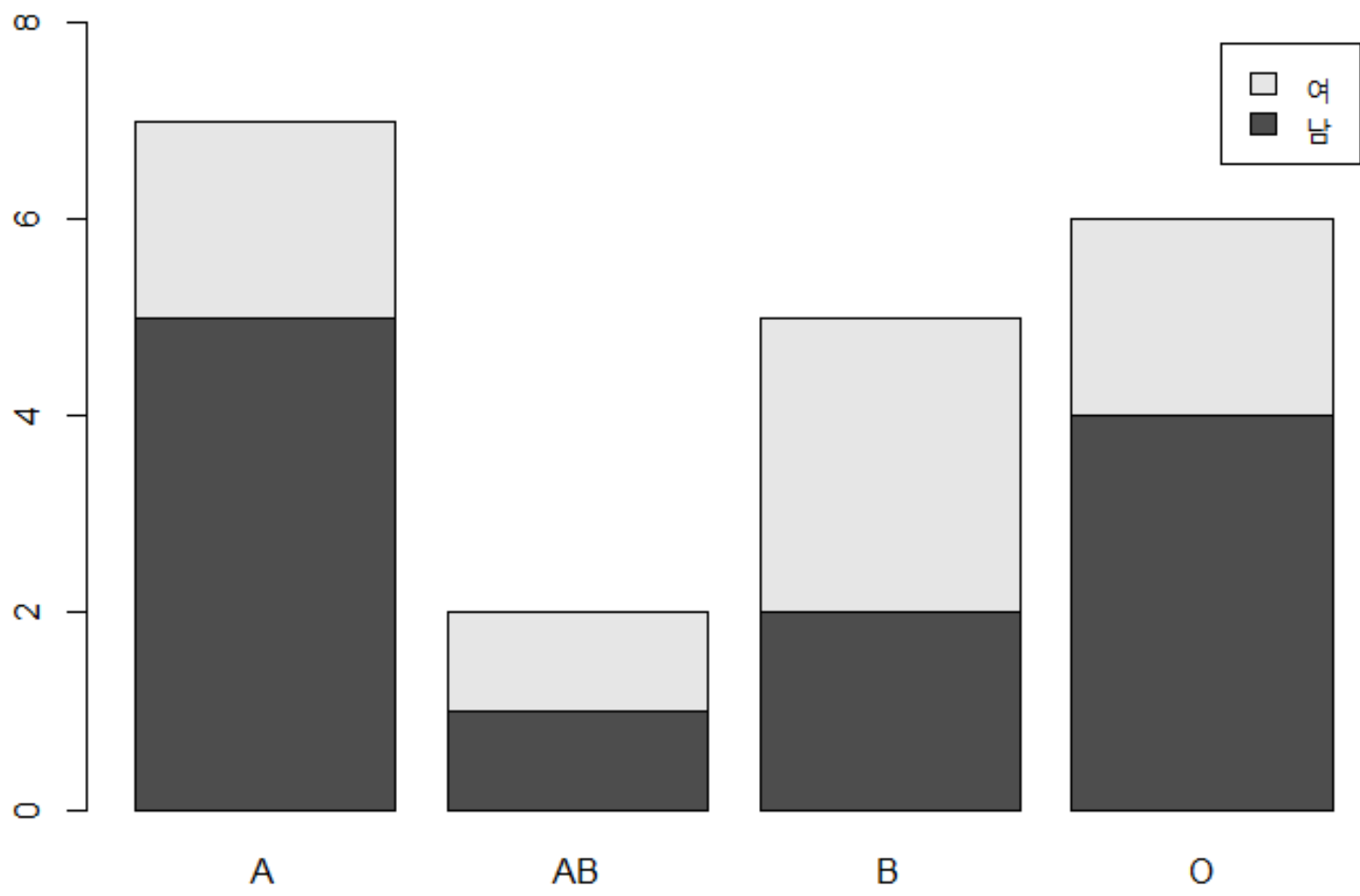
범주형 데이터: 막대차트

심화2

Table을 통해 다양한 방식의 막대차트 생성

코드

```
#y축을 0~8까지 출력하도록 설정#
barplot(classTable, legend=TRUE, ylim=c(0,8))
```



입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정

CHAPTER 3.7

범주형 데이터: 막대차트

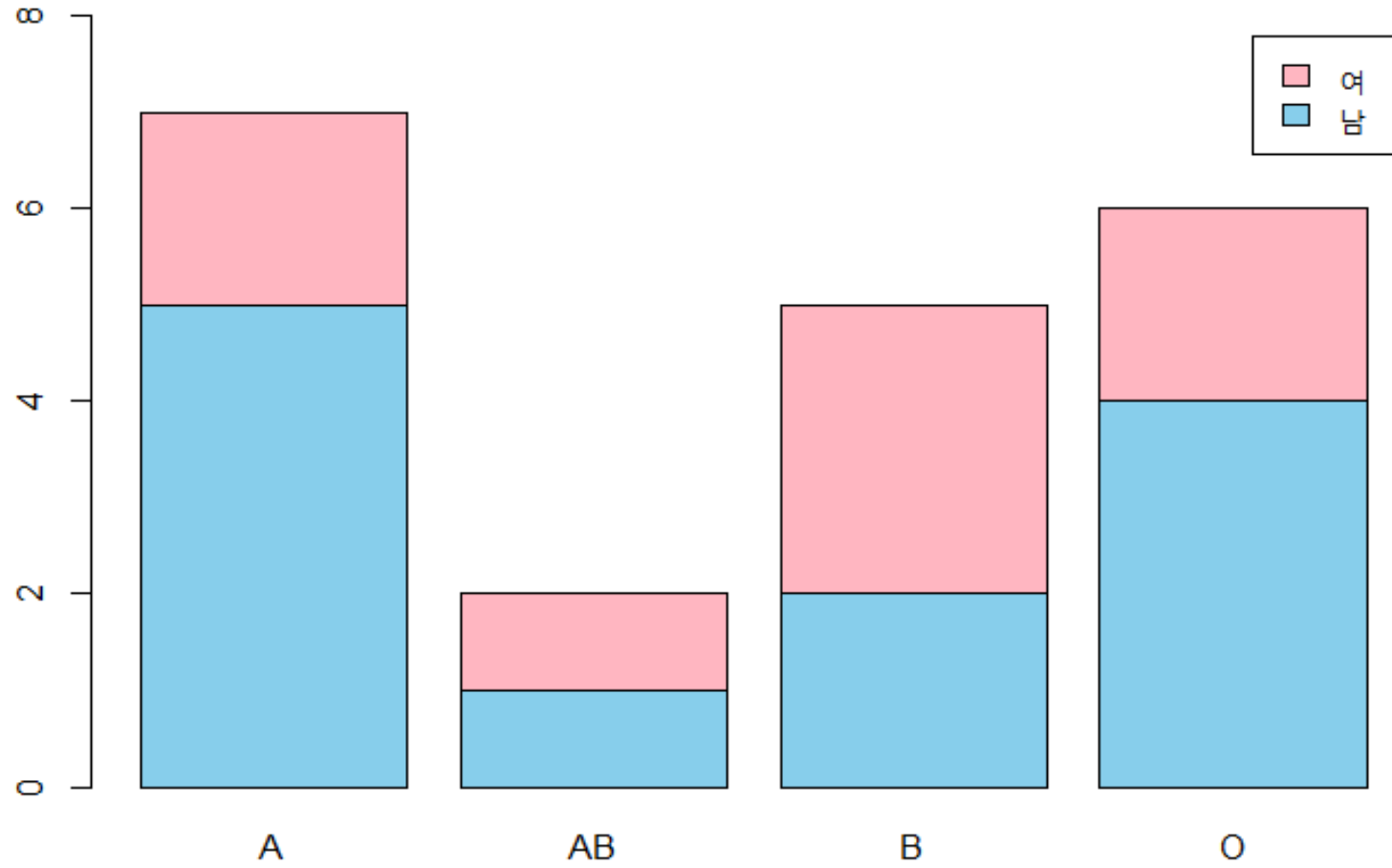
심화2

Table을 통해 다양한 방식의 막대차트 생성

코드

입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정

```
# 색상 지정하기#
barplot(classTable, legend=TRUE, ylim=c(0,8),
        col=c("skyblue", "lightpink"))
```



CHAPTER 3.7

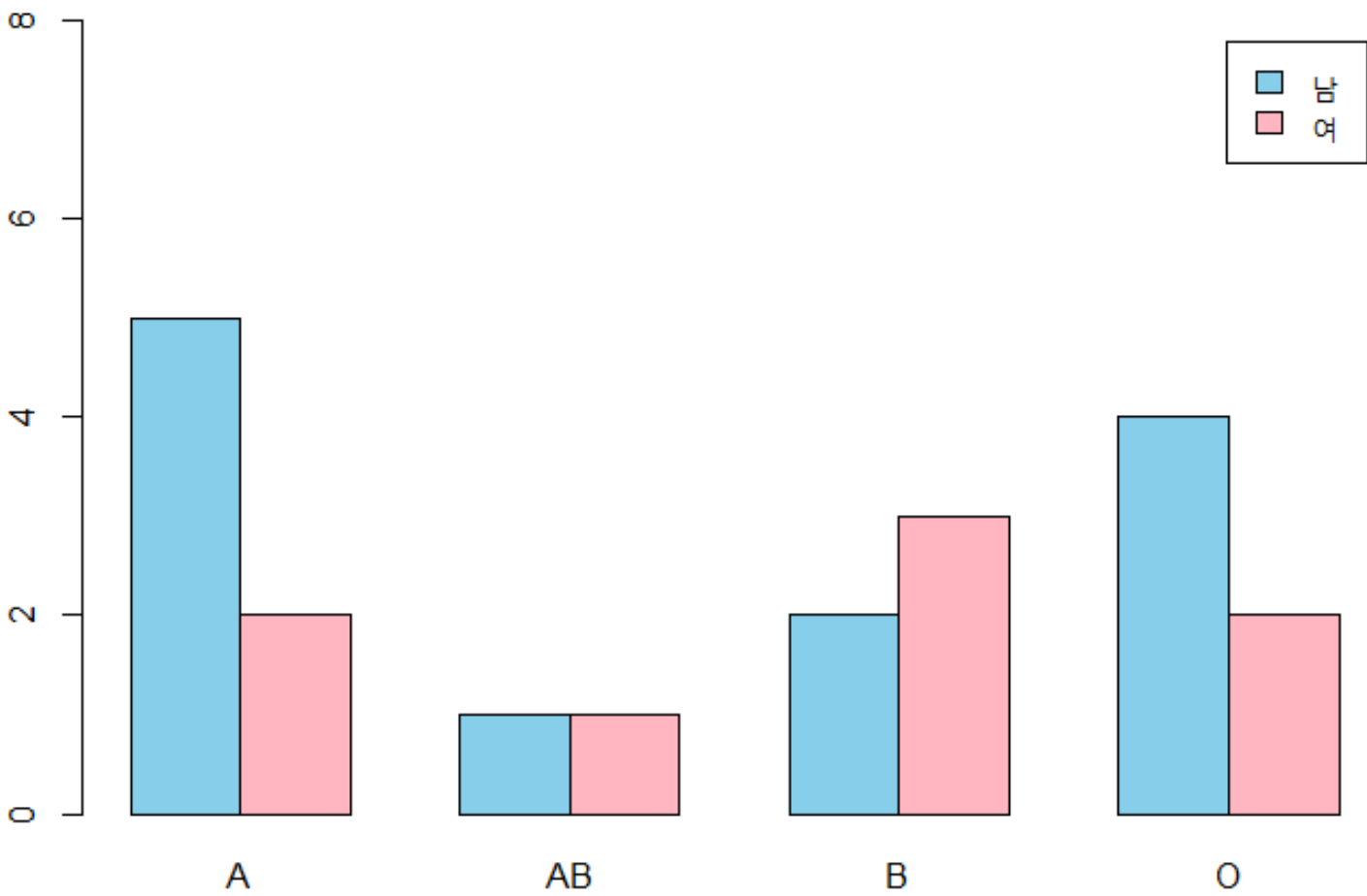
범주형 데이터: 막대차트

심화2

Table을 통해 다양한 방식의 막대차트 생성

코드

```
#beside 입력 항목들 활용해 성별을 별도 막대로 표기, 범례 영역을 위해 y축 길이를 설정#
barplot(classtable, legend=TRUE, ylim=c(0,8),
        col=c("skyblue","lightpink"),
        beside=T)
```



입력 항목	
height=	막대 높이를 지정한 벡터
names.arg=	막대 이름을 지정한 벡터
space=	막대 간의 간격
horiz=	막대를 세로로 표현(TRUE), 가로로 표현(FALSE), 기본값=세로(TRUE)
main=	차트 제목
xlab=	X축 이름
ylab=	Y축 이름
col=	각 범주별 색상을 지정한 벡터
beside=	*범주(종류)가 여러 개인 경우에 사용* 누적된 막대차트로 표현(FALSE), 병렬로 연결된 막대차트로 표현(TRUE), 기본값=누적(FALSE)
xlim=	X축 구간 설정
ylim=	Y축 구간 설정



안 하고 죽어도 좋은 일만
내일로 미루어라.

-피카소

END OF PRESENTATION

통계량 분석과 시각화 기초

비타민 2조

김재승 노신희 우현우 이지선

참고 문헌:

박훈, '시작하세요! 데이터분석 with R', p.161-189