

# pie 함수

📖 pie 함수의 사용법

📖 연속인 변수

Lorem ipsum dolor sit amet, ius an molestie facilisi erroribus, mutat nalerum delectus ei vis. Has ornatus conclusionemque id, an videri molestatis sit. In etqui praesent sit. An vel agan porro comprehensan, ad ludus constituto nea, et ius utroque scaevola assuaverit.

Vis cu nodus nulla feugait, oratio facilisi ex usu, eilit vitae sea te. Ea fabulas accusamus dissonantia sea, facete tacinates definitiones et per. Nihil dicant mediocrem pro eu, no mei nostro sensibus platonem. Qui id sunno perpetua neglegantur. Vel ipsum novum copiosae ut. Quo et liber detracto probatus. Nam augue scribentur an. Sea oporteat percipitur incidere et. Qui viris nemore an.





# pie 함수의 사용법



## 원그래프

- + 범주형 자료의 각 범주별 빈도수(또는 각 범주의 비율)를 원 내부의 각도에 비례하게 그린 그림
- + 범주형 변수 중 주로 명목형에 적합함





# pie 함수의 사용법



R-그래픽의 내장함수인 pie 함수



사용법

```
pie(x, labels = names(x), radius = 0.8, density = NULL, angle = 45,  
    col = NULL, border = NULL, lty = NULL, main = NULL, ...)
```



매개변수

- x : 원그래프를 그릴 값(주로 빈도)
- labels : 원그래프에서 각 슬라이스에 사용할 이름, x에 이름이 없으면 인덱스를 사용함
- radius : 원의 반지름을 설정하며 원 그래프에서 x, y축 모두 -1에서 1사이의 정사각형에 그리는 것을 기본으로 하므로 적절한 반지름 설정
- col : 원의 내부의 색깔
- border : 선(테두리)의 색깔
- density, angle : 각 슬라이스의 내부를 빗금으로 그릴 때 인치(2.54cm)당 빗금의 개수 및 각도 설정



# pie 함수의 사용법



## 예시



### 자료

(옵션에 따른 파이 그림의 형태) 다음은 빈도수 aa를 사용하여 네 개의 서로 다른 형태의 원그래프를 그린 보기이다.

첫 번째 그림은 기본 원그래프이며, 두 번째 그림은 radius를 1로 원을 키운 것이며, 세 번째는 시작하는 위치가 180도인 경우이며, 마지막 그림은 원의 내부 및 테두리의 색을 다르게 설정한 것이다.

```
aa <- c(10,20,30,40)
name.aa <- c("그룹1", "그룹2", "그룹3", "그룹4")
par(mfrow=c(2,2))

pie(aa, main="기본 원크기")
pie(aa, labels=name.aa, radius=1, main="radius=1")
pie(aa, clockwise=T, init.angle=180)
pie(aa, col=2:5, border=6)
```



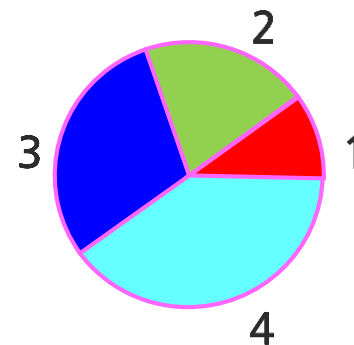
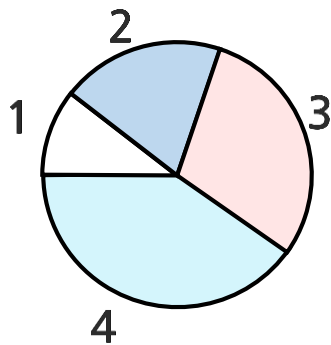
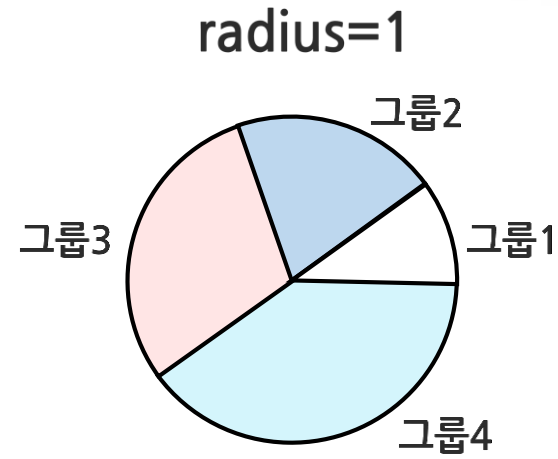
# pie 함수의 사용법



예시



함수 사용 결과



Pie(aa, col=2:5, border=6)





# 연속인 변수



## 원그래프



범주형자료, 그 중에서도 명목형 자료에 적합하며 연속인 자료에는 적합하지 않으나 연속인 변수에도 합에 대한 상대적 개념을 사용할 수 있을 때 원그래프가 가능함



## 예시



### 자료

다음의 지역별 GRDP 자료를 생각해보자.

| 전국        | 서울      | 부산     | 대구     | 인천     | 광주     | 대전     | 울산     | 경기      | 강원     | 충북     | 충남      | 전북     | 전남     | 경북     | 경남      | 제주     |
|-----------|---------|--------|--------|--------|--------|--------|--------|---------|--------|--------|---------|--------|--------|--------|---------|--------|
| 1,484,542 | 327,602 | 73,744 | 46,592 | 68,374 | 30,998 | 32,723 | 69,548 | 329,449 | 36,886 | 49,137 | 103,740 | 44,623 | 63,095 | 91,653 | 102,484 | 13,894 |





# 연속인 변수



## 예시

```
area <- c("서울", "부산", "대구", "인천", "광주", "대전", "울산", "경기",  
"강원", "충북", "충남", "전북", "전남", "경북", "경남", "제주")  
grdp2014 <- c(327602, 73744, 46592, 68374, 30998, 32723, 69548, 329449, 36886,  
49137, 103740, 44623, 63095, 91563, 102484, 13894)  
labs <- paste(area, prettyNum(grdp2014, big.mark=",", preserve.width="none"), sep=": ")  
pie(grdp2014, labels= labs, radius=0.9, col=topo.colors(length(grdp2014)))
```

- Labs : 각 슬라이스에 이름을 주기 위해 만든 문자열로 지역명과 grdp의 문자열을 합한(paste 함수) 것
  - GRDP는 숫자에 콤마(,)를 넣기 위해 prettyNum 함수를 사용함
- prettyNum 함수의 preserve.width 옵션 : 콤마나 소숫점을 사용할 때 원래 길이를 유지할 것인지 설정함





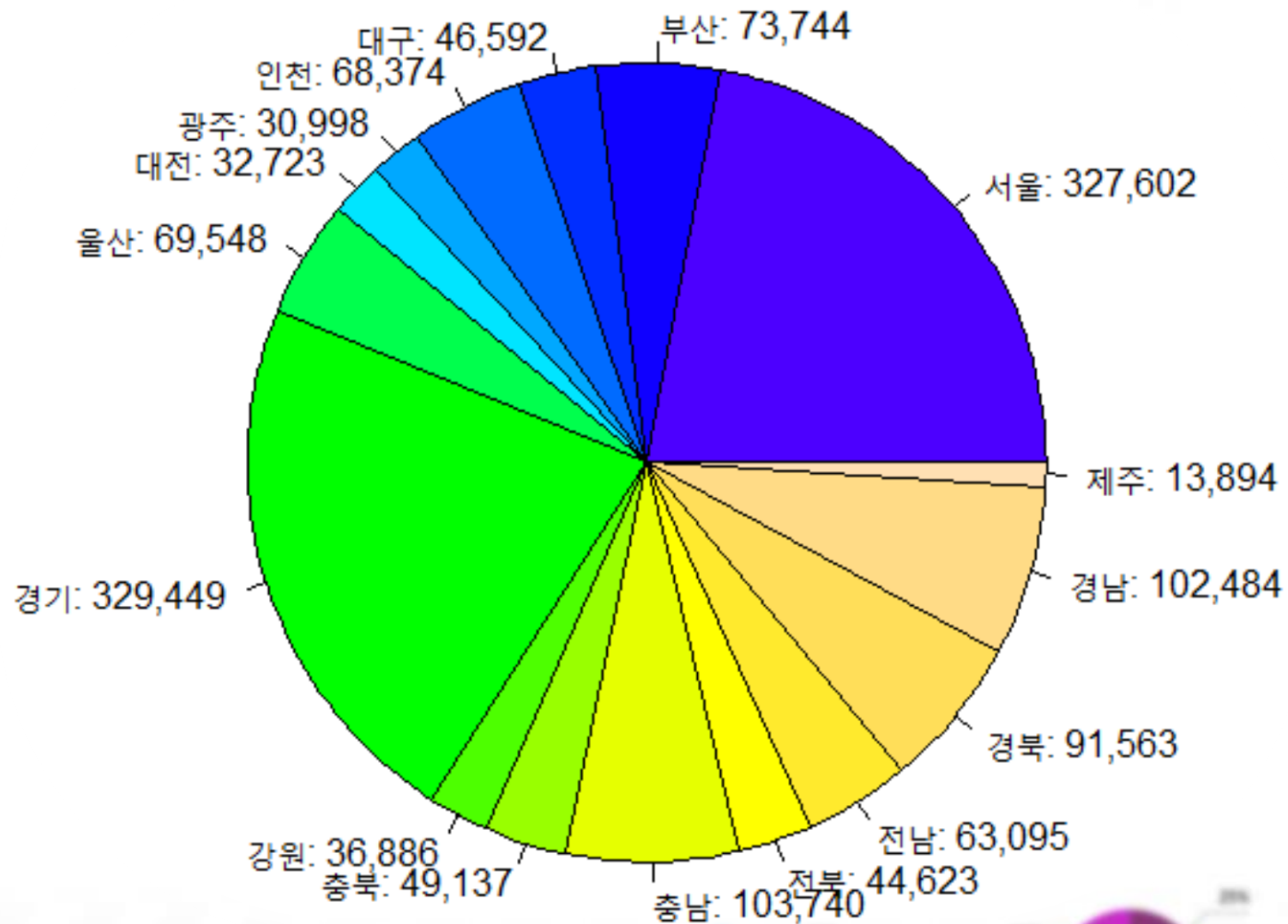
# 연속인 변수



예시



함수 사용 결과





# plot3D 함수

- 📖 plotrix 패키지를 사용한 3차원 원그림
- 📖 마이크로 데이터에서 원그래프

Loren ipsum dolor sit amet, ius an molestie facilisi erroribus, mutat nalerum delectus ei vis. Has ornatus conclusionemque id, an videri molestatis sit. In etqui praesent sit. An vel agan porro comprehensan, ad ludus constituto nea, et ius utroque scaevola assumaverit.

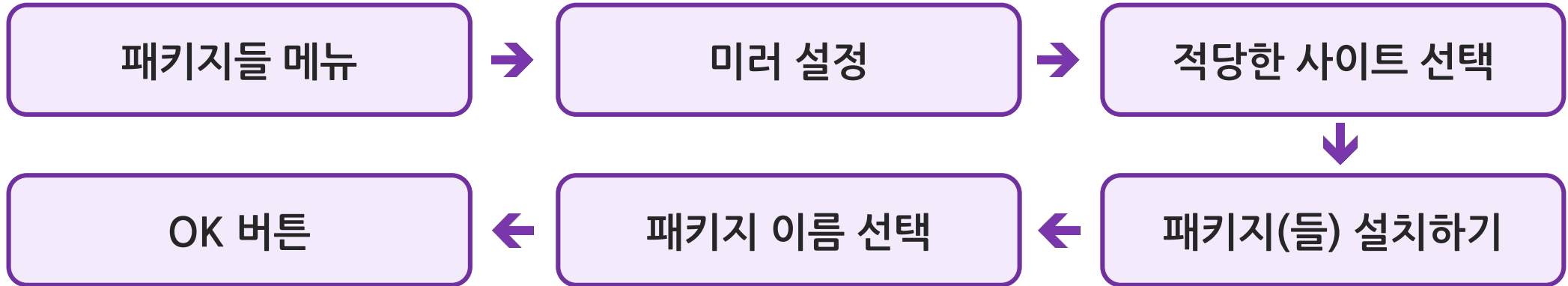
Vis cu nodus nulla feugait, oratio facilisi in usu, eilit vitae sea te. Ea fabulas accusamus dissentias sea, facete tacinates definitiones et per. Nihil dicant mediocrem pro eu, no mei nostro sensibus platonem. Qui id sunno perpetua neglegentur. Vel ipsum novum copiosae ut. Quo et liber detracto probatus. Nam augue scribentur an. Sea oporteat percipitur incidereat ab. Qui viris nemore an.

# plotrix 패키지를 사용한 3차원 원그림



## plotrix 패키지의 pie3D 함수 사용

### 패키지의 설치



### 설치된 패키지를 사용하려면 library 함수로 해당 패키지를 불러옴



# plotrix 패키지를 사용한 3차원 원그림



## plotrix 패키지의 pie3D 함수 사용

### + 사용법

```
pie3D(x, radius=1, height=0.1, border=par("fg"), col=NULL, labels=NULL,  
labelcol=par("fg"), labelcex=1.5, explode=0, border,...)
```

### + 매개변수

- x : 빈도를 포함한 벡터
- labels : 각 슬라이스의 이름
- labelcex : 각 슬라이스 이름의 글자 크기 설정. 1이 기본 크기이며 1보다 작으면 작게 크면 크게 만듦
- labelcol : 슬라이스 이름에 사용할 색깔 설정
- radius : 원의 크기 설정
- height : 높이 설정
- explode : 원그림의 각 슬라이스를 얼마나 분리할지 설정
- border : 각 슬라이스의 테두리 색 설정



# plotrix 패키지를 사용한 3차원 원그림



## 예제




### 자료

다음 보기는 각각의 옵션에 따라 그려지는 plot3D 함수의 결과를 비교해보자.

### 사용할 자료

```
> BMI <- read.table(url("http://jupiter.hallym.ac.kr/ftpdata/data/bmi.txt"),  
  col.names=c("height", "weight", "year", "religion", "gender", "marriage"))
```

-  내용 : 2000년, 177명에 대한 조사 결과
  - 키, 몸무게, 출생년도
  - 종교(Bu=불교, C1=개신교, C2=가톨릭, No=없음)
  - 성별(F=여자, M=남자)
  - 결혼여부(N=미혼, Y=기혼)

# plotrix 패키지를 사용한 3차원 원그림



## 예제

```
library(plotrix)
par(mfrow=c(2,2))
pie3D(table(BMI$religion), labels=c("개신교", "가톨릭", "불교", "없음"))
pie3D(table(BMI$religion), labels=c("개신교", "가톨릭", "불교", "없음"), height=0.3)
pie3D(table(BMI$religion), labels=c("개신교", "가톨릭", "불교", "없음"), explode=0.2)
pie3D(table(BMI$religion), labels=c("개신교", "가톨릭", "불교", "없음"), explode=0.2,
      labelcex=0.7, labelcol="blue", border=1)
```

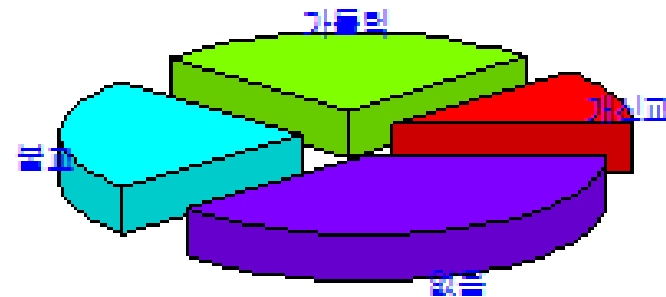
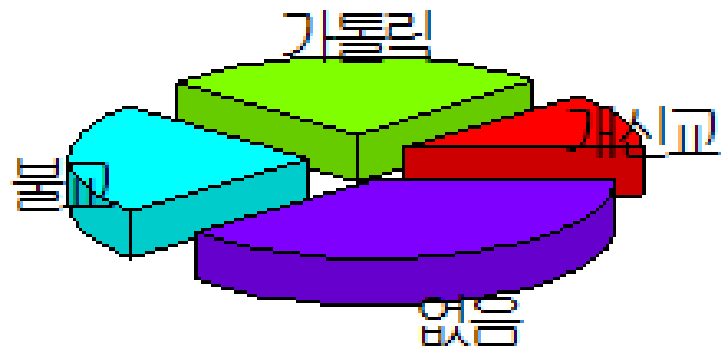
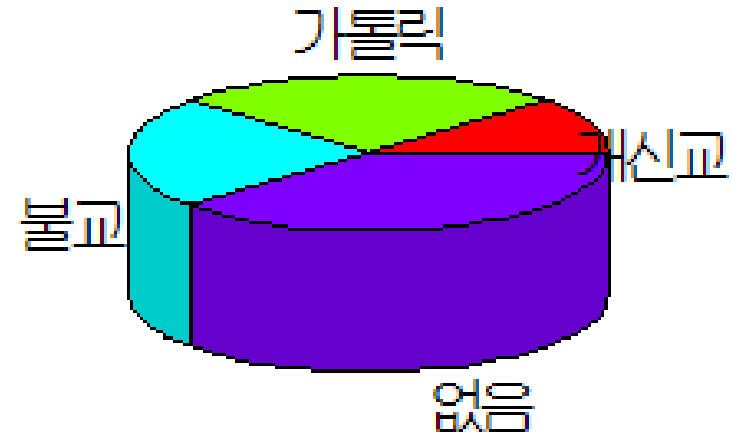
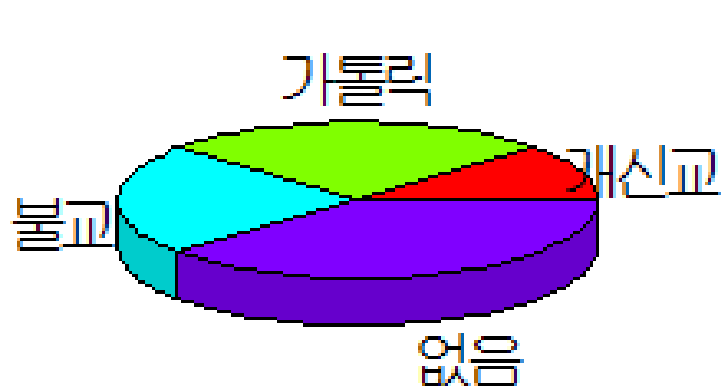
- 첫 번째 : 기본 3D 원그래프를 그림
- 두 번째 : height 값을 바꾸어 높이를 올림
- 세 번째와 네 번째 : explode를 사용하여 각 슬라이스를 분리하였음
- 네 번째 : 레이블의 글자크기를 기본 크기보다 작게(0.7) 하고 슬라이스의 경계의 색을 1(검정)으로 설정함



# plotrix 패키지를 사용한 3차원 원그림

## 예제

### + 함수 사용 결과





# 마이크로 데이터에서 원그래프



마이크로 데이터인 경우 table 함수 사용

- + 원그래프의 매개변수는 barplot 함수와 마찬가지로 마이크로 데이터를 직접 사용하지 못함
- + 마이크로 데이터인 경우 table 함수 등을 사용하여 도수분포표를 얻은 후 그려야 함





# 마이크로 데이터에서 원그래프



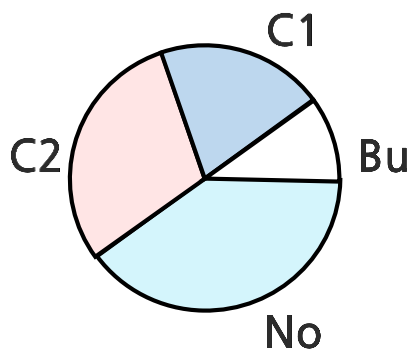
## 예제



### 자료

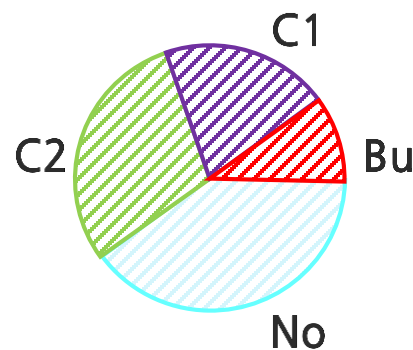
BMI 자료의 종교에 대한 원그래프를 원자료 및 빈도표를 이용하여 그려보자.  
density 옵션에 따른 결과도 함께 알아본다. 다음의 세 명령을 시도해보자.

```
pie(BMI$religion) # 오류 * 마이크로 데이터 직접 사용 불가
pie(table(BMI$religion))
pie(table(BMI$religion), density=10, col=c("red", "blue", "green", "cyan"))
```



### 두 번째 pie 함수

\* table 함수를  
사용하여 얻은  
도수분포표



### 세 번째 pie 함수

\* density를 설정하여  
각 슬라이스에 빗금  
만들었음

# hist 함수

- 📖 hist 함수의 사용법
- 📖 두 개의 히스토그램 비교
- 📖 히스토그램 함수의 출력

Loren ipsum dolor sit amet, ius an molestie facilisi erroribus, mutat nalerum delectus ei vis. Has ornatus conclusionemque id, an vide molestatis sit. In etqui praesent sit. An vel agan porro comprehensan, ad ludus constituto nea, et ius utroque scaevola assuaverit.

Vis cu nodus nulla feugait, oratio facilisi ex usu, eili vitae sea te. Ea fabulas accusamus dissentias sea, facete tacinates definitiones et per. Nihil dicant mediocram pro eu, no mei nostro sensibus platonem. Qui id sunno perpetua neglegantur. Vel ipsum novum copiosae ut. Quo et liber detracto probatus. Nam augue scriben- tur an. Sea oporteat percipitur incidereat ab. Qui viris nemore an.



# hist 함수의 사용법



## 히스토그램(histogram)

- + 연속인 자료에 대해서 구간을 나누어 해당 구간에 포함된 자료의 빈도(또는 비율)을 기둥의 높이로 하여 막대를 연속으로 그린 그림
- + 막대그래프와 히스토그램의 차이

### 막대그래프

범주형 자료에 적용되고 따라서  
각 기둥이 분리되어 그려짐

VS

### 히스토그램

연속인 자료에 적용되고 따라서 기둥을  
분리하지 않음





# hist 함수의 사용법



R-그래픽의 내장함수인 hist 함수



사용법

```
hist(x, freq=, probability = !freq, breaks="Sturges", density = NULL,  
     angle = 45, col = NULL, border = NULL,  
     main = paste("Histogram of" , xname),  
     labels = FALSE, nclass = NULL, ...)
```



매개변수

- x : 히스토그램을 그릴 자료를 포함한 벡터
- freq, probability : 히스토그램에서 기둥 높이를 절대빈도로 할지 상대빈도로 할지 설정(둘 중 하나만 설정함)
- labels : 기둥에 대한 설명을 넣을지 설정하거나 기둥의 값을 설정(TRUE이면 해당 기둥의 빈도수가 표시됨)
- nclass : 기둥의 개수 설정
- breaks : 각 구간의 끝점 설정
- col, border : 각 기둥의 내부 색깔 및 테두리 색깔 설정





# hist 함수의 사용법



## 예제



### 자료

정규분포 난수를 사용하여 히스토그램을 그리되 각 기둥의 이름을 설정하는 방법을 함께 고려하여 그려보자.

```
x <- rnorm(100)
par(mfrow=c(1,2))
hist(x, labels=T)
hist(x, labels=c("A", "B"))
```



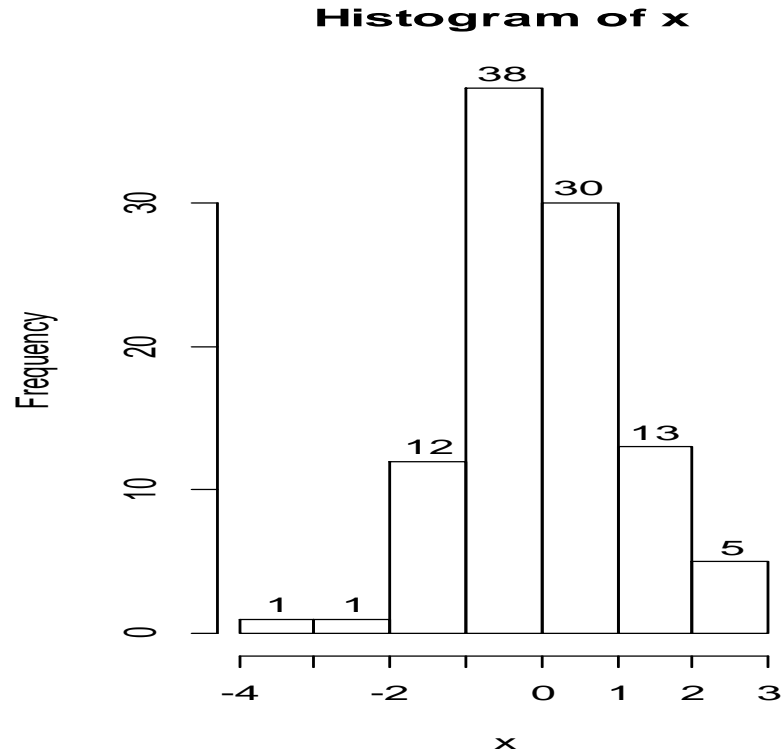
# hist 함수의 사용법



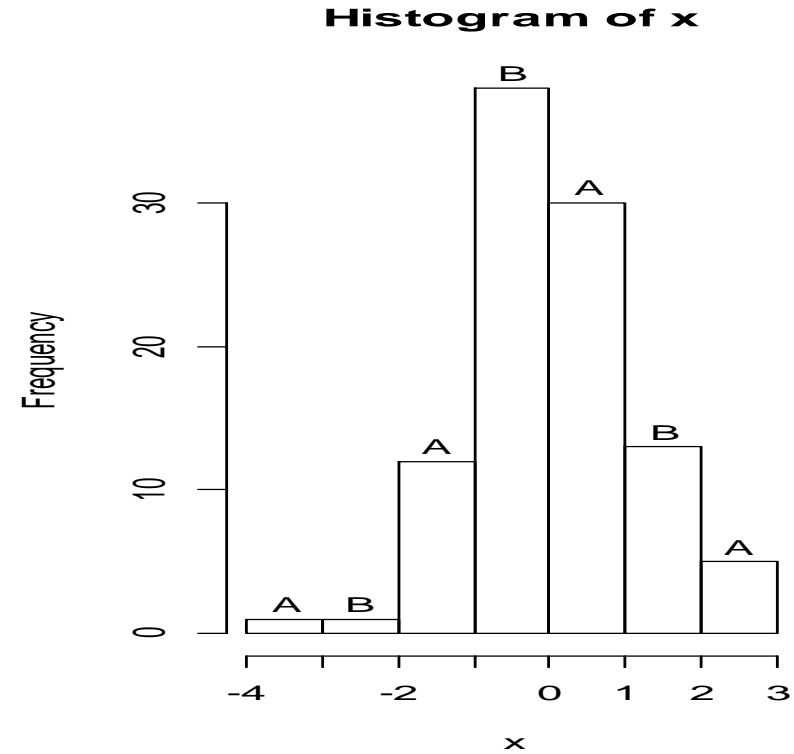
## 예제



### 함수 사용 결과



\* 각 기둥에 빈도수가 인쇄됨



\* 설정된 문자 A와 B가 인쇄됨  
\* 설정된 문자의 수가 기둥 수보다 작으므로 반복하여 사용됨



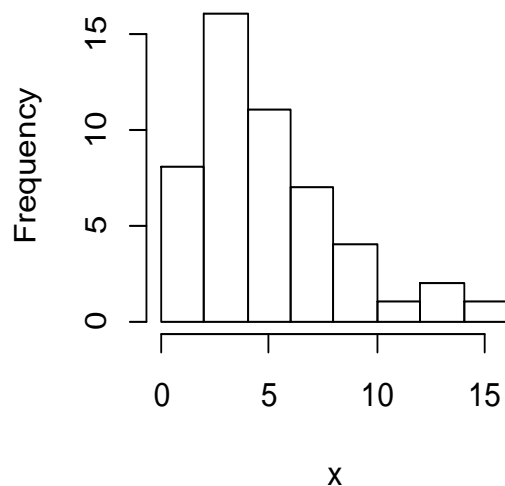
# 두 개의 히스토그램 비교



기둥의 수 또는 기둥의 폭이 다른 히스토그램

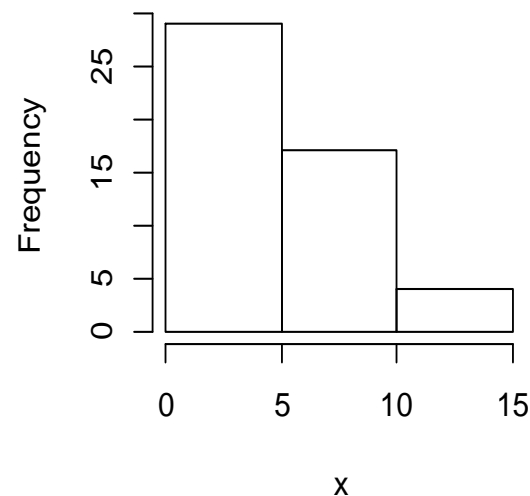
```
x <- c(8, 2, 10, 1, 1, 6, 6, 3, 4, 4, 3, 4, 5, 3, 14, 5, 3, 4, 7, 9, 5, 1, 10, 6, 2, 4, 6, 8, 3, 4, 13,  
       3, 5, 7, 7, 6, 6, 7, 3, 8, 4, 4, 5, 1, 1, 12, 2, 9, 3, 15)  
par(mfrow=c(1,2))  
hist(x)  
hist(x, nclass=3)
```

Histogram of x



\* 기본값 적용

Histogram of x



\* 기둥의 수를  
3으로 설정





# 색깔과 구간 설정



## 예시



### 색깔과 구간설정

히스토그램을 작성할 때 기둥의 수를 설정할 수도 있지만 각 기둥이 만들어질 구간을 설정할 수도 있음 → breaks 설정

```
par(mfrow=c(1,2))  
x <- rnorm(100)  
mybreaks <- seq(-3, 3, by=1)  
hist(x)  
hist(x, breaks=mybreaks, col=rainbow(8), border="blue")
```



# 색깔과 구간 설정

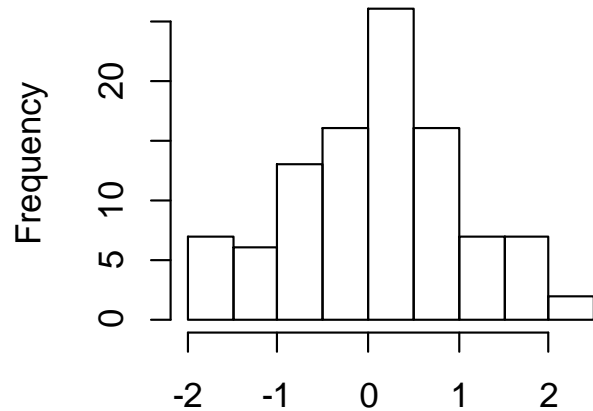


예시



함수 사용 결과

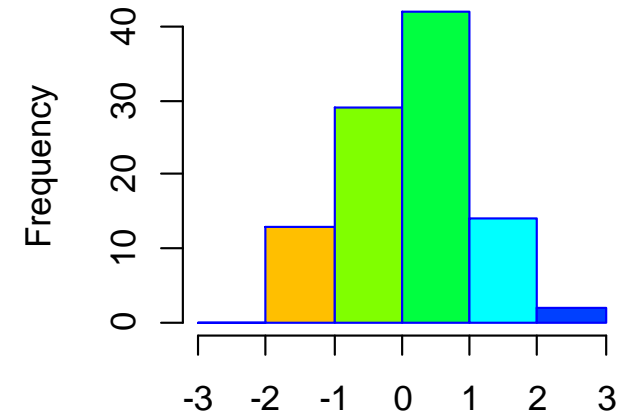
Histogram of x



x

\* 기본값 적용

Histogram of x



x

- \* 기둥의 경계는 breaks 옵션에 설정
- \* 내부색 : 무지개색(rainbow 함수)
- \* 기둥의 테두리는 blue
- \* 양쪽 끝점이 -3,3





# 색깔과 구간 설정



## 참고

- + breaks를 설정할 때 자료의 값 중 일부가 breaks의 범위를 벗어나 있는 경우 에러 메시지가 나올 수 있음

[예] 자료 중 -3과 3 바깥에 3.5, -3.1 등-이 있으면 이 명령은 '일부 x가 세어지지 않았다'는 에러가 발생함





# 히스토그램 함수의 출력



히스토그램 함수 hist의 호출로 출력되는 값들

- + breaks : 각 기둥의 경계점
- + count : 각 기둥의 절대빈도
- + density : 각 기둥의 상대빈도
- + mids : 각 기둥의 중간점





# 히스토그램 함수의 출력



## 사용 함수

```
x <- rnorm(100)
hist.res <- hist(x)
hist.res
$breaks
[1] -4 -3 -2 -1 0 1 2 3
$counts
[1] 1 2 13 36 30 13 5
$density
[1] 0.01 0.02 0.13 0.36 0.30 0.13 0.05
$mids
[1] -3.5 -2.5 -1.5 -0.5 0.5 1.5 2.5
:
```



# pyramid 함수

- 📖 pyramid 패키지를 사용한 인구 피라미드
- 📖 pyramid 함수의 사용

Lorem ipsum dolor sit amet, ius an molestie facilisi erroribus, mutat nalerum delectus ei vis. Has ornatus conclusionemque id, an vide molestatis sit. In etqui praesent sit. An vel agan porro comprehensan, ad ludus constituto nea, et ius utroque scaevola assuaverit.

Vis cu nodus nulla feugait, oratio facilisi ex usu, eili vitae sea te. Ea fabulas accusamus dissentias sea, facete facinorae definitiones et per. Nihil dicant mediocrem pro eu, no mei nostro sensibus platonem. Qui id sunno perpetus neglegentur. Vel ipsum novum copiosae ut. Quo et liber detracto probatus. Nam augue scribentur an. Sea oporteat percipitur incidereit at. Qui viris nemore an.



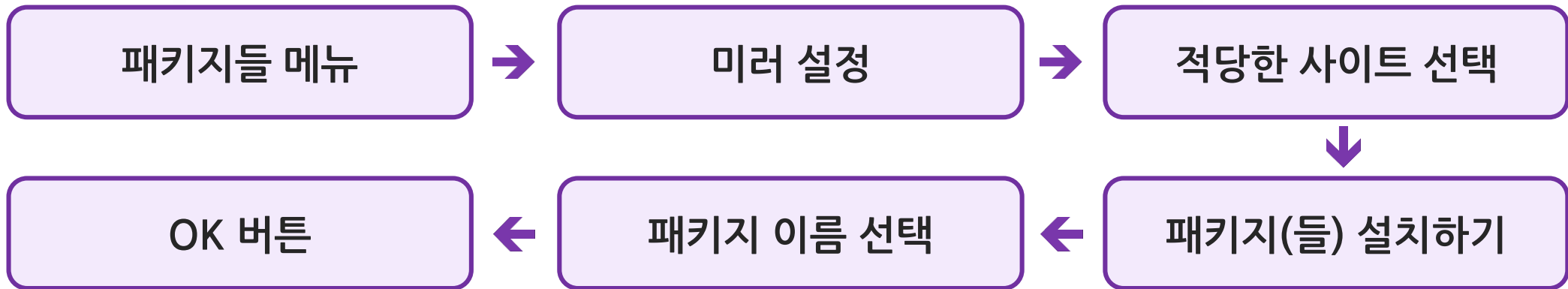
# pyramid 패키지를 사용한 인구 피라미드



## pyramid 패키지의 pyramid 함수 사용



### 패키지의 설치



### 설치된 패키지를 사용하려면 library 함수로 해당 패키지를 불러옴





# pyramid 패키지를 사용한 인구 피라미드



## pyramid 패키지의 pyramid 함수 사용



### 사용 함수

```
pyramid(data, Llab="Males", Rlab="Females", Clab="Ages", ...)
```



### data에는 세 개의 열을 포함한 데이터 프레임을 설정함

- data의 첫 번째 열은 왼쪽, 두 번째 열은 오른쪽에 해당하는 빈도수를 가지며, 세 번째 열은 이름이 포함됨
- 이름이 포함된 세 번째 열은 생략할 수 있으며 생략된 경우 data의 행 이름이 사용됨



### 매개변수

- Llab, Rlab, Clab : 인구 피라미드의 왼쪽, 중간 및 오른쪽에 사용할 이름을 설정함

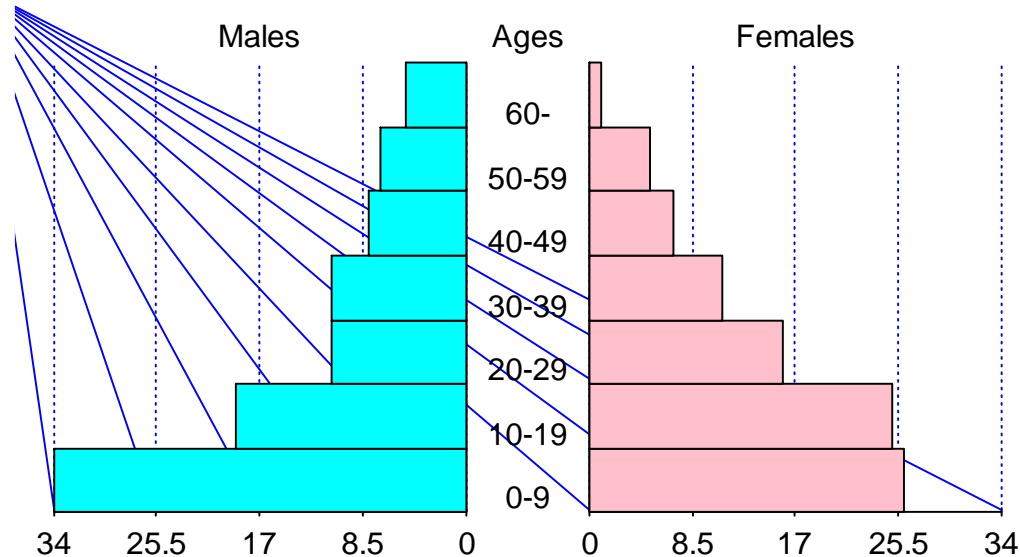


# pyramid 함수의 사용



## 연령별 인구를 인구 피라미드로 그리기

```
library(pyramid)
ages <- c('0-9','10-19','20-29','30-39','40-49','50-59','60-')
males <- c(34,19,11,11,8,7,5)
females <- c(26,25,16,11,7,5,1)
data <- data.frame(males,females,ages)
pyramid(data)
```





# pyramid 함수의 사용



## 예시



### 자료

2010년 통계청 인구주택 총조사에 의한 우리나라 인구에 대한 인구 피라미드는 다음과 같이 얻을 수 있다. 이 자료는 국가통계포털 KOSIS(<http://kosis.kr/>)에서 얻을 수 있으며 아래의 xlsx 패키지는 R-기초 과목에서 다룬 패키지로 엑셀 파일을 다룰 수 있는 기능을 제공한다. 이 패키지의 read.xlsx 함수는 엑셀의 스프레드 시트를 R의 데이터 프레임으로 읽어 들이는 함수이다.





# pyramid 함수의 사용



예시



사용 함수

```
library(xlsx)
df <- read.xlsx("d:/hwp/STI/population2010.xlsx", sheetIndex=1, endRow=20,
               startRow=2, encoding="UTF-8")
head(df)
```

|   | NA.      | 전체      | 남자      | 여자      |
|---|----------|---------|---------|---------|
| 1 | 0 - 4세   | 2219084 | 1142220 | 1076864 |
| 2 | 5 - 9세   | 2394663 | 1243294 | 1151369 |
| 3 | 10 - 14세 | 3173226 | 1654964 | 1518262 |
| 4 | 15 - 19세 | 3438414 | 1826179 | 1612235 |
| 5 | 20 - 24세 | 3055420 | 1625371 | 1430049 |
| 6 | 25 - 29세 | 3538949 | 1802805 | 1736144 |

```
xx <- data.frame(df$남자/10000, df$여자/10000, df$연령)
pyramid(xx, Llab="남자", Rlab="여자", Clab="연령", main="2010년 인구(만명)")
```



# pyramid 함수의 사용



예시



함수 사용 결과

2010년 인구(만명)

