

# R 기초 프로그래밍

한국환경정책·평가연구원  
부연구위원 진대용

강의자료 : <https://github.com/dyjin1217>

# 프로그래밍 언어

- 사람과 컴퓨터 사이에 존재하는 일종의 의사소통 수단
  - 컴퓨터한테 명령을 내리기 위한 수단



# R 프로그램

- R은 통계 계산과 그래픽을 위한 **프로그래밍 언어**이자 소프트웨어 환경 (위키피디아)
- R의 활용
  - 통계분석
  - 기계학습
  - 네트워크 분석
  - 웹 프로그램
  - 유전체 분석
  - ...



# R 프로그램 장/단점

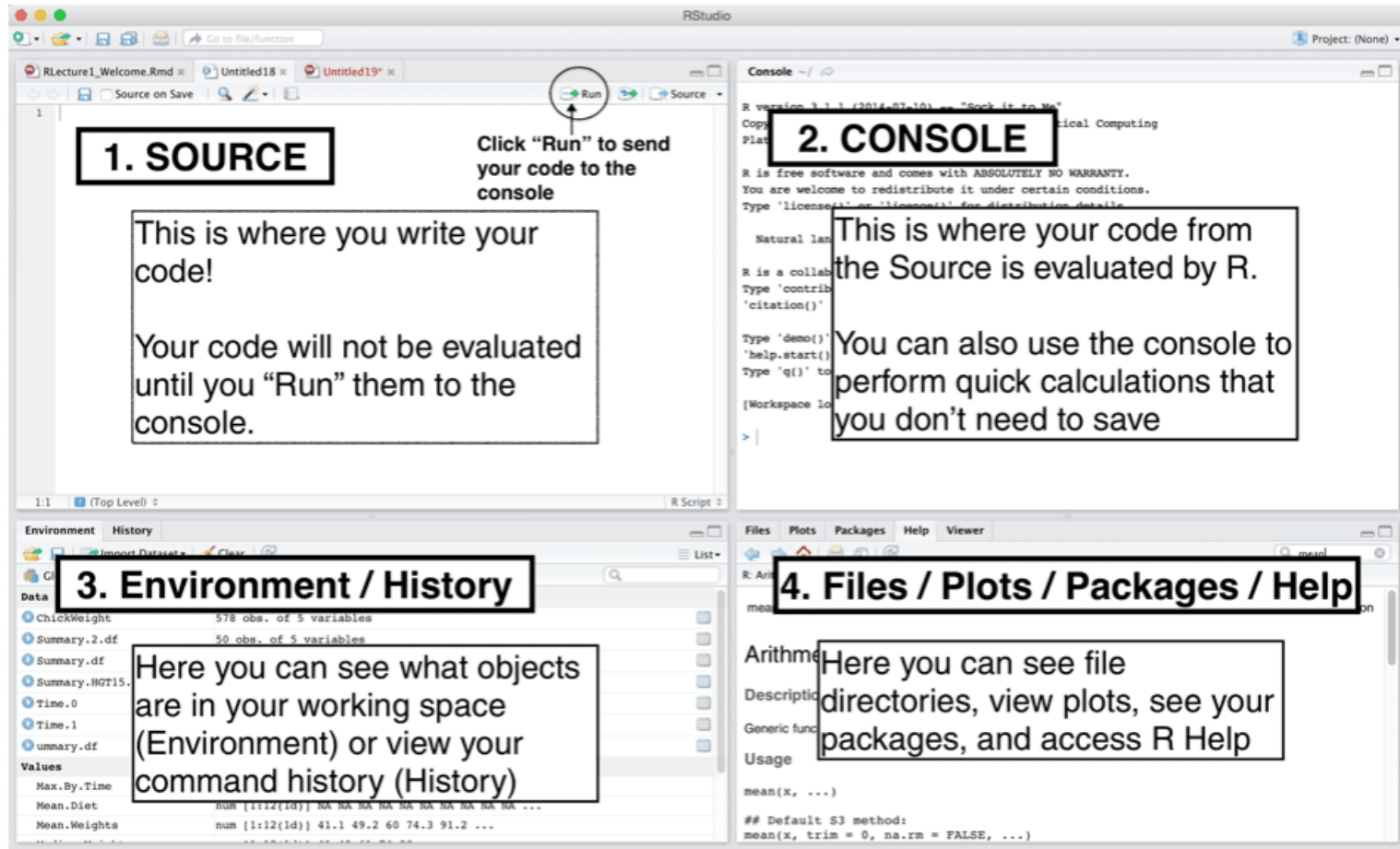
- 장점
  - 다양한 패키지
  - 시각화
  - 무료 (오픈 소프트웨어)
- 단점
  - 느림
  - 높은 메모리 요구
- 그 외
  - 유연성 (예 : 데이터 타입)



# R 프로그램 설치

- R 설치
  - <https://www.r-project.org>
- R-studio 설치
  - <https://www.rstudio.com/>
  - 통합개발환경 : 개발에 관련된 모든 작업을 하나의 프로그램 안에서 처리하는 환경
- 참조 사이트
  - <https://backgomc.tistory.com/34>

# R-Studio 시작화면



# 첫 프로그래밍

- $2 + 3$
- "Hello World"
- $X \leftarrow 3$
- $X * X$

# R의 연산

- 산술 연산 (+, -, /, %%, ^ ...)
  - $3 + 2$ ,  $X + 2$
- 논리 연산 (AND(&), OR(|), NOT(!)...)
  - $X \% 2 == 0 \ \& \ X \% 3 == 0$
- 비교(관계) 연산 (>, <, ==, ...)
  - $X > 2$
  - $X == 3$
- 대입/할당 (연산)
  - $X <- 3$  (기본)
  - $X = 3$  (함수에서 주로 사용)



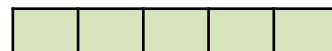
# R 데이터 타입

기본 데이터 타입 (원자 벡터)

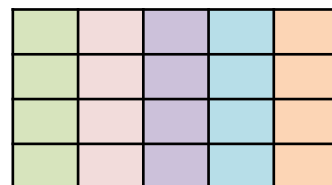
객체(변수)	예
integer	100
double	0.05
character	"Hello"
logical	TRUE
factor	"Black"

데이터 타입(자료의 형태)

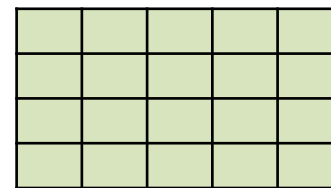
Vector



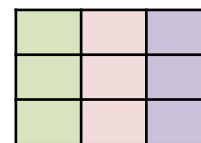
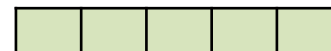
Data frame



Matrix



List



"R에서 모든 객체(변수)는 데이터 타입(자료의구조)를 가지고, 기본이 되는 데이터 타입은 벡터(Vector)"

# R 데이터 타입

- 벡터(vector)
  - 1차원 데이터, 동일 데이터 타입
- 행렬(matrix)
  - 2차원 행\*열 데이터, 동일 데이터 타입
- 데이터 프레임(dataframe)
  - 2차원 행\*열 데이터, 열 단위 다른 데이터 가능
- 리스트(list)
  - 다수 객체를 인덱스를 활용하여 사용

# R 데이터 타입 연습

- 연습
  - `A <- 3`
  - `A <- 4.5`
  - `A <- "안녕"`
  - `A <- c(1,2,3)`
  - `A <- c("사과","바나나","배")`
  - `A <- matrix(c(1,2,3,4), nrow=2)`
  - `A <- data.frame(숫자=c(1,2,3),과일= c("사과","바나나","배"))`

# 데이터 타입 알아보기

- `typeof(객체)`
  - 기본 데이터 타입을 출력
- `class(객체)`
  - 객체의 종류 출력, 벡터는 기본 데이터 타입을 출력
- `str(객체)`
  - 객체구조 출력

# 데이터 타입 확인 및 변경

- `is.character(객체)`
  - `is.character(c(1,2))` # FALSE
- `as.character(객체)`
  - `as.character(c(1,2))`
- `as.matrix(객체)`
  - `A <- as.matrix(c(1,2,3,4))`
- `as.vector(객체)`
  - `A <- as.vector(A)`

# 벡터 생성

- 1~5 벡터 생성
  - `A <- c(1,2,3,4,5)`
  - `A <- 1:5`
  - `A <- seq(1,5,1)` # seq(시작값, 끝값, 등차)
- 5가 5번 벡터 생성
  - `A <- rep(5,5)` # rep(숫자, 반복횟수)

# 벡터 원소 추출

- R의 인덱스는 1부터 시작
  - 예 : 1번째 원소는 인덱스 1
- 인덱스 기반 벡터 원소 추출
  1. `A <- 1:10`
  2. `A[3]` : 3번째 원소 추출 #A[인덱스]
  3. `A[1:3]` : 1~3번째 원소 추출 # A[시작인덱스:끝인덱스]
  4. `A[5:10]` : 5번째 원소부터 끝까지 출력 # A[시작인덱스:]
  5. `A[1:2]` : 2번째 원소까지 출력 # A[:끝인덱스]
  6. `A[-1]` : 1번째 인덱스 제외
  7. `A[-(1:3)]` : 1번째 인덱스 제외

# 벡터 원소 추출

- 값 기반 원소 추출
  - `A <- c(1,3,5,3,2,5,3)`
  - `A == 3`
  - `A[A==3]`
  - `A[A>3]`
  - `A %in% c(3,5)`
  - `A[A %in% c(3,5)]`



# 행렬 생성 및 추출

- 1~4 2 × 2 행렬 생성
  1. `V <- c(1,2,3,4)`
  2. `A <- matrix(V, nrow=2, byrow=2)`
- 1~25사이 숫자 생성 및 5\*5 행렬 생성
  1. `V <- 1:25`
  2. `A <- matrix(V, nrow=5) # matrix(데이터(벡터), 행, 열)`
  3. `A[3,3] # 3,3`
  4. `A[1:3,4:5] #1~3행, 4~5열 추출`
  5. `A[,4:5] #모든행 4~5열 추출`
  6. `A[-1,] #행 제외 추출`
  7. `A[,-(2:3)] # 2~3열 제외 추출`

# 데이터 프레임 생성 및 추출

- 2개 다른 기본타입의 벡터를 데이터프레임으로
  1. `A <- data.frame(숫자=c(1,2,3),과일= c("사과","바나나","배"))`
  2. `A[1:2,1:2]` # 1~2행, 1~2열
  3. `A[1:3,]` # 1~3행 추출
  4. `A[-1,]` # 1행 제외 추출
  5. `A[,-c(2,3)]` #2~3열 제외 추출
  6. `A$과일` or `A[, '과일']` # 과일열 추출
  7. `A$과일[2]`
- 참고 : `data.frame(벡터1,벡터2, ...)`, 동일길이 벡터를 주로 활용

# 리스트 생성 및 추출

- 리스트 생성 및 추출
  1. `L <- list(x = 1:3, y = c("A","B","C"))`
  2. `L$x`
  3. `L[1]`
  4. `L[[1]]`

# 대입연산자

- 대입연산
  - `A <- 1:10`
  - `A[1] <- 4`
  - `A[1:3] <- 4 # Broadcasting`
  - `A[1:3] <- 3:5`
  - `A <- matrix(1:8,nrow=2)`
  - `A[,2] <- c(4,1)`
  - `A[1:2,1:2] <- 2`
  - `A[1:2,1:2] <- matrix(c(1,2,3,4),nrow=2)`

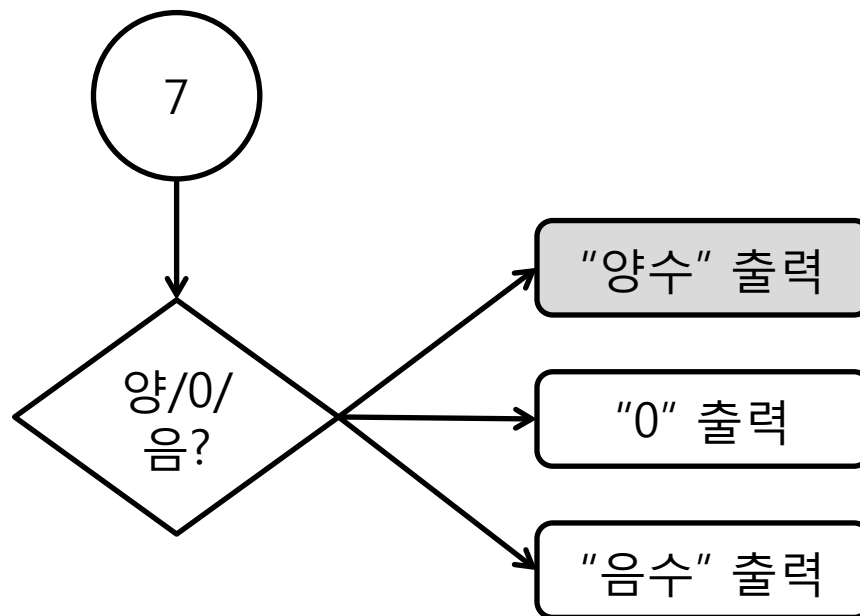
# 팩터(factor)

- 특정 수준 값만을 가질 수 있는 데이터 타입
  1. `A <- factor(c("바나나","사과","오렌지"))`
  2. `A[3] <- "Hello" # Error`
  3. `A[3] <- "오렌지"`
  4. `A <- data.frame(숫자=c(1,2,3),과일= c("사과","바나나","배"))`
  5. `A <- data.frame(숫자=c(1,2,3),과일= c("사과","바나나","배"), stringsAsFactors=FALSE)`

# 조건문

- 조건문 if

1. if(조건1){
2. 코드1
3. } else if(조건2)
4. 코드2
5. } else{ # 그외
6. 코드3
7. }



# 조건문

- X가 7일때 양수/음수/0 판단 조건문 프로그램

```
1. X <- 7
2. if(X>0){
3.   cat("양수")
4. } else if(X==0){
5.   cat("0")
6. } else{
7.   cat("음수")
8. }
```

# 연습문제

- 점수를 입력받아 성적을 판단 프로그램
  - 90 이상 수
  - 80~90 이하 우
  - 70~80 이하 미
  - ...
- 훌쩍 판단하는 프로그램



# 반복문

- 반복문
  - for : 일반적으로 가장 많이 사용
  - while : 무한루프, while(1){ }, while(TRUE){} 형태
  - repeat

# for문

- for 문
  1. for(반복변수 in 객체(보통 벡터)){
  2.       코드
  3. }

# for문

- 벡터값 1개씩 출력

1. `fruits <- c('사과','배','귤','참외','자두')`
2. `for(fruit in fruits){`
3.     `print(fruit)`
4. `}`

```
fruits <- c('사과','배','귤','참외','자두')
```

인덱스	1	2	3	4	5
값	사과	배	귤	참외	자두

# for문

- 1에서 100까지의 합

1:100 벡터

1. `sum <- 0`

2. `for(i in 1:100)`

3. `{`

4.     `sum <- sum + i`

5. `}`

6. `print(sum)`

인덱스	1	2	3	...	100
값	1	2	3	...	100

# for문

- 구구단 출력

1. `for(i in 1:9){`
2.     `for(j in 1:9){`
3.         `print(paste(i, "*", j, "=", i*j, sep=""))`
4.     `}`
5. `}`

# while문

- 1에서 100까지의 합

1. `i <- 1`

2. `sum <- 0`

3. `while (i <= 100){`

4.     `sum <- sum + i`

5.     `i <- i + 1`

6.     `}`

7. `print(sum)`

# while문

- 구구단 출력

```
1. i <- 1
2. while (i <= 9){
3.   j <- 1
4.   while (j <= 9){
5.     cat(i, "*", j, "=", i*j, "\n", sep="")
6.     j <- j + 1
7.   }
8.   i <- i + 1
9. }
```

# 반복문 제어

- break : 루프 탈출

```
1. for(i in 1:10){  
2.     if(i %% 2 == 0){  
3.         break  
4.     }  
5.     print(i)  
6. }
```



# 반복문 제어

- next : 반복문 블록의 수행을 중단하고 다음 반복을 시작

```
1. for(i in 1:10){  
2.     if(i %% 2 == 0)  
3.         next  
4.     print(i)  
5. }
```

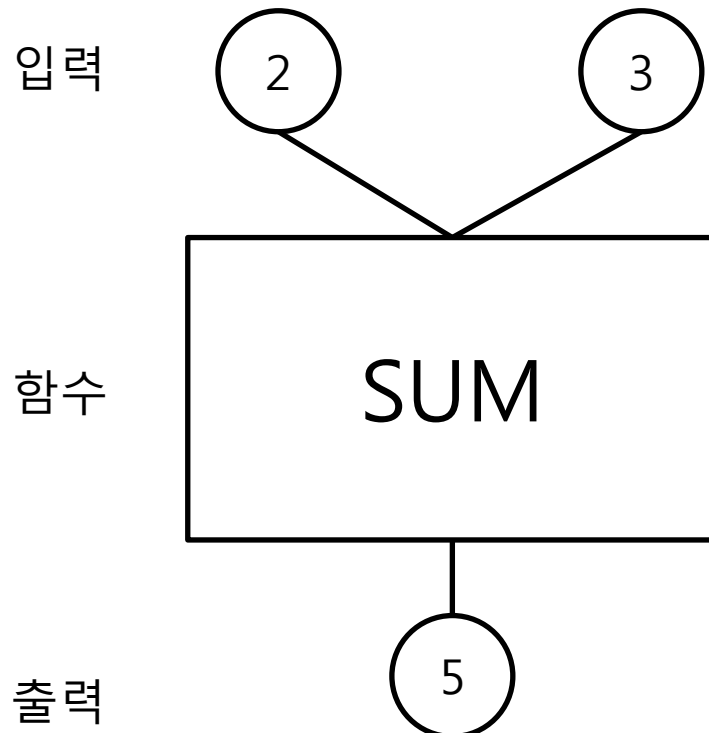
# 연습문제

- “exit”가 입력될 때 까지 무한 루프를 수행하는 코드를 작성
- $1+2+3+\dots+N$ 이 150이상인 되는 첫  $N$ 을 구하는 코드를 작성
- 1~9까지 제곱합 출력 ( $1+4+9\dots$ )

# 함수

- 함수 : Function

- 함수정의 : 함수 <- function(인자1,인자2,...) { 내용 }
- 함수호출 : 함수(인자1,인자2,...)



```
> sum(2,3)
[1] 5
```

# 함수

- 2개의 값을 입력 받아 평균을 내는 함수

- 함수정의 : 함수 <- function(인자1,인자2,...) { 내용 }

- 1. ave\_two <- function(n1,n2)

- 2. {

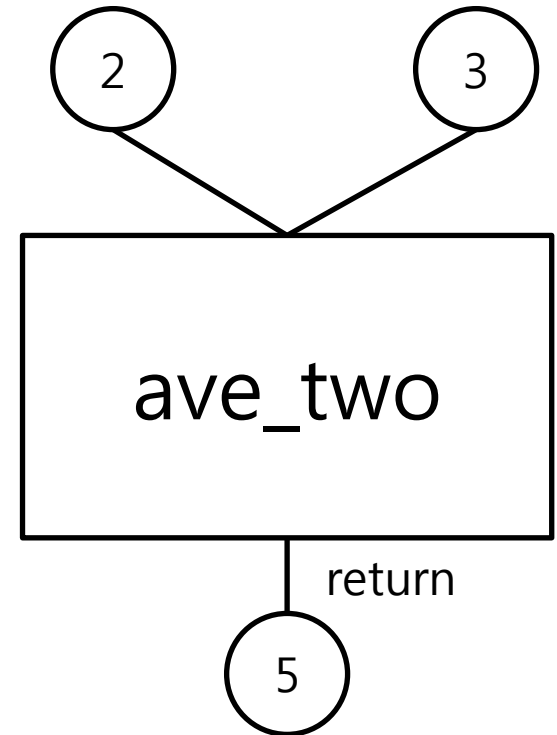
- return (n1 + n2)

- 3. }

- 함수호출 : 함수(인자1,인자2,...)

- 1. ave\_two(2,3)

입력



출력

# 연습문제

- 2개의 값을 입력 받아 큰 수 출력하는 함수 구현
- 2개의 값을 입력 받아 그 사이에 해당하는 값을 더해주는 함수 구현 (두수도 포함)
- 3개의 값을 입력 받아 1번째 수 + 2번째 수 - 3번째 수를 수행하는 함수 구현

# 그외 주요내용

- names : 벡터에 이름 부여
- colnames : 열 이름
- rownames : 행 이름
- NULL/NA 처리 : is.na(객체), is.null(객체)
- which : 인덱스 위치
- paste : 문자열을 붙일때
- unlist
- sort : 정렬
- order : 정렬 색인
- ....

# R 통계량 구하기

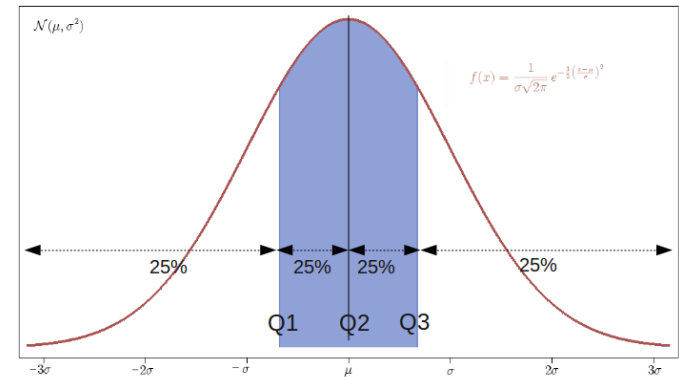
## Maths Functions

<code>log(x)</code>	Natural log.	<code>sum(x)</code>	Sum.
<code>exp(x)</code>	Exponential.	<code>mean(x)</code>	Mean.
<code>max(x)</code>	Largest element.	<code>median(x)</code>	Median.
<code>min(x)</code>	Smallest element.	<code>quantile(x)</code>	Percentage quantiles.
<code>round(x, n)</code>	Round to n decimal places.	<code>rank(x)</code>	Rank of elements.
<code>signif(x, n)</code>	Round to n significant figures.	<code>var(x)</code>	The variance.
<code>cor(x, y)</code>	Correlation.	<code>sd(x)</code>	The standard deviation.

# R 통계량 구하기

- 통계량 구하기

1. `score = c(50,60,100,95,85)`
2. `mean(score)` # 평균
3. `median(score)` # 중앙값
4. `max(score)` # 최고값
5. `quantile(score)` # 4분위수,  $IQR = Q3 - Q1$
6. `which(score == 95)` # 인덱스



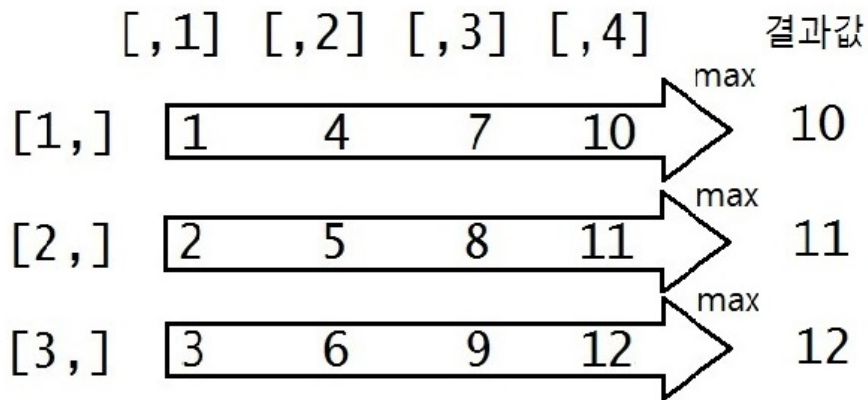


# 연습문제

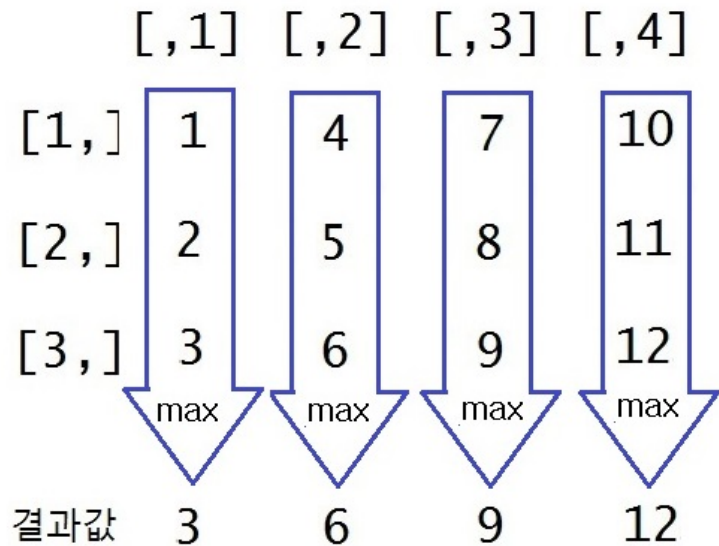
- 성적표
  - kor : 90, math : 95, hist : 80, ps : 70, music : 70
- 문제
  - 점수를 score 벡터에 저장
  - names를 이용해 제목 부여
  - 평균, 중앙값, 표준편차, 최고점 과목

# apply 함수

```
> apply(a, 1, max)
```



```
> apply(a, 2, max)
```



- apply(객체, 1 or 2, 함수)
- 1 : 행 , 2 : 열

# 연습문제

- 1~25로 이루어진  $5 * 5$  행렬 생성
- 문제
  - 행 합, 열 합
  - 행 평균, 열 평균
  - 행 최대값, 열 최대값

# R 시각화

## SET GRAPHICAL PARAMETERS

the following can only be set with `par()`  
`par(...)`

<i>multiple plots</i>	<code>mfcol = c(nrow,ncol)</code>	<i>plot margins (outer)</i>	<code>oma = c(bottom, left, top, right) default: c(0,0,0,0) lines</code>
	<code>mfrow = c(nrow,ncol)</code>		
<i>plot margins</i>	<code>mar = c(bottom, left, top, right) default: c(5.1,4.1,4.1,2.1) lines</code>	<i>query x &amp; y limits</i>	<code>par("usr")</code>

## CREATE A NEW PLOT

<b>Bar charts</b>	<code>barplot(height,...)</code>	<b>Histograms</b>	<code>hist(x,...)</code>
<i>bar labels</i>	<code>names.arg =</code>	<i>breakpts</i>	<code>breaks =</code>
<i>border</i>	<code>border =</code>		
<i>fill color</i>	<code>col =</code>		
<i>horizontal</i>	<code>horiz = TRUE</code>		
<b>Box plots</b>	<code>boxplot(x,...)</code>	<b>Line charts</b>	<code>plot(x, type = "l")</code>
<i>horizontal</i>	<code>horizontal = TRUE</code>	<i>line type</i>	<code>lty = "blank" 0 "solid" 1 "dashed" 2 "dotted" 3</code>
<i>box labels</i>	<code>names =</code>	<i>line width</i>	<code>lwd =</code>
<b>Dot plots</b>	<code>dotchart(x,...)</code>	<b>Scatterplots</b>	<code>plot(x,...)</code>
<i>dot labels</i>	<code>labels =</code>	<i>symbol</i>	<code>pch =</code>

## REMOVE

<i>axis labels</i>	<code>ann = FALSE</code>
<i>axis, tickmarks, and labels</i>	<code>xaxt = "n" yaxt = "n"</code>
<i>plot box</i>	<code>bty = "n"</code>

NOTE: Many of the parameters here can be also be set in `par()`. See R help for more options.

## ADJUST

<i>allow plotting out of plot region</i>	<code>xpd = TRUE</code>
<i>aspect ratio</i>	<code>asp =</code>
<i>axis limits</i>	<code>xlim =, ylim =</code>
<i>axis lines to match axis limits</i>	<code>xaxs = "i", yaxs = "i" (internal axis calculation)</code>

## ADD TEXT

### location

<i>axis labels</i>	<code>xlab =, ylab =</code>
<i>subtitle</i>	<code>sub =</code>
<i>title</i>	<code>main =</code>

### style

<i>font face</i>	<code>font = 1 (plain) 2 (bold) 3 (italic) 4 (bold italic)</code>
<i>font family</i>	<code>family = "serif" "sans" "mono"</code>

### size

(magnification factor)

<i>all elements</i>	<code>cex =</code>
<i>axis labels</i>	<code>cex.lab =</code>
<i>subtitle</i>	<code>cex.sub =</code>
<i>tick mark labels</i>	<code>cex.axis =</code>
<i>title</i>	<code>cex.main =</code>

### position

<i>text direction</i>	<code>las = 1 (horizontal)</code>
<i>justification</i>	<code>adj = 0 .5 1 (left, center, right)</code>

## ADD TO AN EXISTING PLOT

**Add new plot** [any plot function]  
(...,add = TRUE)  
ex. `barplot(x, add = TRUE)`

**Axes** `axis(side,...)`  
*location* `side = 1 2 3 4  
(bottom, left, top, right)`

*tick mark:*  
*labels* `labels =`  
*location* `at =`  
*remove* `tick = FALSE`  
*rotate text* `las = 1 (horizontal)`

**Axis labels** `mtext(text,...)`  
*location* `side = 1 2 3 4  
(bottom, left, top, right)`  
*lines to skip* `line = (from plot  
region, default=0)`


*position* `at = x or y-coord  
(depending on side)`

*justification* `adj = 0 .5 1  
(left, center, right)`

**Lines** `lines(x,...)`

*line style* `lty =`  
*line width* `lwd =`  
*color* `col =`

**Points** `points(x,...)`

*symbol* `pch =`  
  
*color* `col =`  
*fill color* `bg = (pch: 21-25 only)`

**Text** `text(x,y,text,...)`  
*position (rel. to x,y)* `pos = 1 2 3 4  
(below, left, above, right)  
(default=center)`

**Title** `title(main,...)`  
*axis labels* `xlab =, ylab =`  
*subtitle* `sub =`  
*title* `main =`

Joyce Robbins, joycerobbins1@gmail.com

# R 시각화

- Scatter Plot

1. `x <- c(1,2,3,4,5)`
2. `y <- c(1,4,3,5,9)`
3. `plot(x,y)`
4. `plot(x,y,main="title")`
5. `plot(x,y,font=3, main="title")`
6. `plot(x,y,font.main=3, main="title")`
7. `plot(x,y,xlim=c(1,5),main="title")`
8. `plot(x,y,font.main=3,xlim=c(1,5),main="title",ann=FALSE)`

# R 시각화

- Scatter Plot

1. `x <- c(1,2,3,4,5)`
2. `y <- c(1,4,3,5,9)`
3. `plot(x,y)`
4. `title(main="title")`
5. `title(xlab="num")`
6. `text(3,3, "Hello")`

# R 시각화

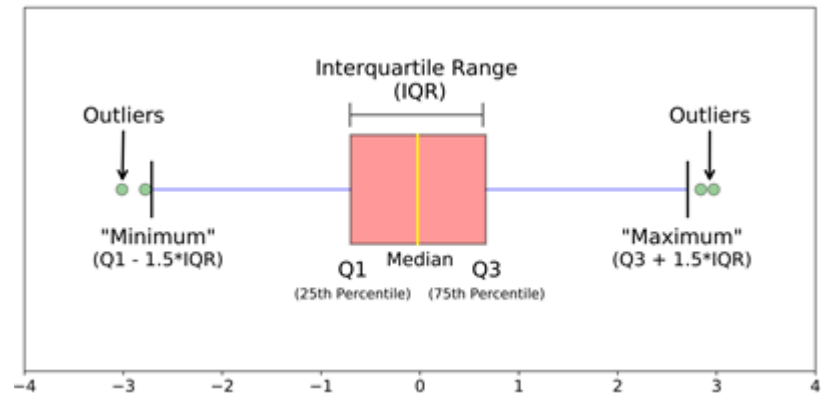
- 히스토그램

1. `A <- round(runif(100, min = 0,max = 100),0)`
2. `hist(A, main="")`
3. `title("histogram")`

# R 시각화

- Boxplot

1. `A <- round(runif(100, min = 0, max = 100), 0)`
2. `boxplot(A)`
3. `title(main="boxplot")`
4. `title(xlab="Hello")`
5. `axis(4)`
6. `lines(c(1,1), c(30,50))`





# R 활용사례 : iris 데이터 분석

변수명	변수설명
Species	붓꽃의 종. setosa, versicolor, virginica 세 가지 값 중 하나
Sepal.Width	꽃받침의 너비
Sepal.Length	꽃받침의 길이
Petal.Width	꽃잎의 너비
Petal.Length	꽃잎의 길이

- iris\_analysis.R 참조

# R 활용사례 : mtcars 데이터 분석

변수명	변수설명
mpg	연비 (Miles per Gallon)
cyl	실린더 개수
disp	배기량
hp	마력
drat	후방차축 비율
wt	무게
qsec	1/4 마일에 도달하는데 걸린 시간
vs	엔진 (0 : V engine 1 : Straight engine)
am	변속기 (0 : 자동, 1 : 수동)
gear	기어 개수?
carb	기화기(카뷰레터) 개수

- mtcars\_analysis.R 참조