

# R 프로그래밍 기초다지기

---

## 1강 - R 프로그래밍 첫 걸음마 떼기

슬기로운통계생활

Issac Lee



# 코스 훑어보기.

Big picture가 중요합니다.

# 프로그래밍이란 무엇일까?



## 결국에는 계산기

- 옛날 주판에서 시작해서
- 오늘날에 컴퓨터가 나옴.

## 무엇이 변했을까?

- 고급 연산 가능
- 반복되는 연산 저장 사용 - 함수
- 개인 코드를 공유 가능 - 패키지



# R 프로그래밍 역사



## 알아두면 재미있는 이야기

- AT&T의 Bell 연구소에 근무하던 통계학자 John M. Chambers가 1976년 **S**라는 언어를 만듦.
- **S**tatistical computing language
- 초기 S는 2가지 버전이 있었음.  
S (무료), S-Plus (유료)



# 왜 R 인가



## R의 갓파더스

- 1991년 뉴질랜드의 University of Auckland에 있는 Ross Ihaka과 Robert Gentleman 교수 둘이서 S 언어를 복사하기 시작함.



- 두 사람의 이름의 첫글자와 S보다 "앞선" R로 이름을 붙임.
- S-Plus에서 돌아가는 코드 R에서 다 돌아감.

# R의 장점



## 프로그래머가 아닌 사람들을 위한 언어

- 테마: 사용자가 왕이다.
- 비프로그래머를 위한 프로그램 언어. (말 다했쥬?)
- 번거로운 것은 내가 다 해줄게, 사용자 너는 그냥 🌸꽃길🌸만 걸자.





# R 코드 실행 방법

R 코드를 돌리는 방법에는 2가지가 있습니다.

## Interactive mode

- R 콘솔 (Console) 창에서 돌리는 방법
- R script에서 코드를 콘솔로 보내는 방법. (Ctrl + Enter)

## Batch mode

- 미리 저장된 `.R` 파일을 window 커맨드 혹은 파워셸(PowerShell) 에서 돌리는 방법

```
R CMD BATCH test.R
```



# R 세션 실행하기

## 변수 만들기

- 변수는 object가 들어갈 수 있는 상자라고 생각하자.
- **R**에서는 상자의 크기가 자유자재로 늘어날 수 있다.

```
x <- 1  
x
```

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

```
x <- c(1, 2, 4)  
x
```

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

- 변수에 어떤 것을 할당 (assignment) 할 때, **R**에서는 `<-` 오퍼레이터를 사용함.



# 벡터 (Vector)



앞선 예제에서 숫자를 여러개 묶어서 하나의 물건 (객체, object) 처럼 만들 수 있는 방법.

```
c(1, 2, 4, 5)
```

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



`c`는 영어로 `concatenate`을 의미.



# 객체 접근방법 - `[]` 연산자

- 벡터는 여러 숫자를 색인을 통하여 접근함.
- 첫번째 자리가 1에서부터 시작.
- 대괄호 `[]` 를 사용해서 부분에 접근 할 수 있음.
- 4개 숫자가 들어있는 벡터 `x`의 3번째 숫자에 접근하는 방법

```
x <- c(1, 2, 4, 5)  
x[3]
```

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



# 객체 접근방법 - 콜론(:) 연산자

- 처음 시작 숫자와 마지막 숫자까지 1씩 증가 혹은 감소하는 수열을 만들어줌.

1:5

5:1

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

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

- 이것을 사용해서 Subset을 할 수 있음.

```
x[2:4]
```

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

# 기본 제공함수 사용하기



## base 함수들

- base 함수 - R에서 기본으로 제공하는 함수들의 모임
- R이 시작될 때부터 자동으로 R의 시스템에 올라와있음. 그냥 쓰면됨.

```
mean(x)
```

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

## 함수 사용법 확인법

- 콘솔 창에 `?mean()` 쳐보자.
- 함수이름 {소속된 패키지}

# R 뜯어보기



작은 차이가 고수를 만든다.

# R 함수 시스템의 구조



## 패키지 불러오기

- 유용한 함수들이 모여있는 패키지들을 시스템 상에 붙일(attach) 때 `library()` 함수를 사용

```
library(dplyr) # already installed
```

# 함수의 namespace 개념



관찰을 잘 하자.

- `library(dplyr)`를 하고나면 경고 메시지가 뜬.

```
The following objects are masked from 'package:stats':
```

```
filter, lag
```

- 무슨 뜻일까? 해석 연습하기!

# 벡터화 (Vectorized) 코드



## R을 R로 만들어주는 특징

- 수학에서 배우는 벡터 연산을 기본적으로 지원함.

```
c(1, 2, 4) + c(2, 3, 5)
```

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

```
x * 2
```

```
## [1] 2 4 8 10
```



# 재활용(Recycling) 개념



- 길이가 맞지 않는 벡터들을 자동으로 길이를 맞춰줌.

```
c(1:4) + c(1, 2)
```

```
## [1] 2 4 4 6
```

```
c(1:4) + c(1:3)
```

```
## Warning in c(1:4) + c(1:3): 두 객체의 길이가 서로 배수관계에 있지 않습니
```

```
## [1] 2 4 6 5
```



# R 프로젝트 만들기

## 기본 작업단위

- 오른쪽 위 상자버튼 클릭 > New Project
- 프로젝트 이름 및 경로 설정
- 프로젝트를 열면 우리가 설정한 폴더가 기본 작업경로(working directory)로 설정이 됨.

```
getwd()
```

- 장점: 특별한 일이 없을 시 경로 자동 설정
  - 예: `write_csv(mtcars, "mtcars.csv")`

# R 시작 설정



The dotfiles: `.Renviron`과 `.Rprofile` 파일

## `.Renviron`

- `.Renviron`: 프로젝트 시작 시 설정하고 싶은 환경변수 설정
- 개인정보, 비밀번호 설정하기 좋음.

```
MYPASSWORD = hi1234  
my_pw <- Sys.getenv("MYPASSWORD")
```

## `.Rprofile`

- `.Rprofile`: 프로젝트 시작 시 돌리고 싶은 R 코드

```
print("Welcome again, Issac!")
```

# 다음시간

본격 R 코드 수업

$$\begin{Bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{Bmatrix}$$

벡터와 행렬



## 사용교재

### [1] [The art of R programming](#)

- R 공부하시는 분이면 꼭 한번 보셔야하는 책입니다.
- 위 교재의 한글 번역본 [빅데이터 분석 도구 R 프로그래밍](#)도 있습니다. 도서 제목 클릭하셔서 구매하시면 저의 [사리사욕](#)을 충당하는데 도움이 됩니다.

## 참고자료

[1] [https://cran.r-project.org/doc/FAQ/R-FAQ.html#Why-is-R-named-R\\_003f](https://cran.r-project.org/doc/FAQ/R-FAQ.html#Why-is-R-named-R_003f)

[2] [https://en.wikipedia.org/wiki/John\\_Chambers\\_\(statistician\)](https://en.wikipedia.org/wiki/John_Chambers_(statistician))

[3] Rstudio 블로그: [Rprofile 파일 설명](#)