

전산통계학 실습

06. R 설치 및 실행

목차

- R 설치와 실행 (Rstudio)
- R 패키지와 데이터 다루기
- R 의 자료형

R

- R
 - 통계학자들의 요구에 따라 설계한 언어 및 소프트웨어
 - 통계와 관련된 전반적인 업무가 가능한 프로그래밍 언어
- R의 활용
 - 의학, 수학 등 통계가 필요한 다양한 학문에서 사용되는 중
 - 특히, 다양한 자료들을 가져와 모두 저장할 수 있어 유리함



R

- 장점

- 다양한 패키지를 통한 case study 가 가능
 - 수많은 통계 관련 패키지 및 기능이 이미 개발되어 저장되어 있음
- 진입장벽이 낮으며, 시각화를 위한 각종 편의 도구가 존재함
- 오픈소스이기 때문에 무료로 사용 가능

- 단점

- In-Memory 기술을 이용함
 - 프로그램 데이터가 하드 디스크가 아닌 메인 메모리에 모두 올라감
 - 대용량 데이터를 다루는 경우 처리에 불편이 있을 수 있음
- 다른 프로그램 대비 GUI 기능이 부족

R 설치

- Windows

- 아래 링크 중 하나로 접속하여 운영체제에 맞도록 다운로드
 - Official: <http://www.r-project.org>
 - CRAN: <http://cran.nexr.com>
 - 위 사이트에서 튜토리얼, 함수의 설명, 추가 패키지 등을 볼 수 있음

- Mac

- Mac에는 R 프로그래밍이 기본으로 설치되어 있음
 - 터미널에 'R' 을 입력하여 설치되어 있는지 확인
- 없거나 최신 버전이 아닌 경우, 위 CRAN 사이트에서 다운로드

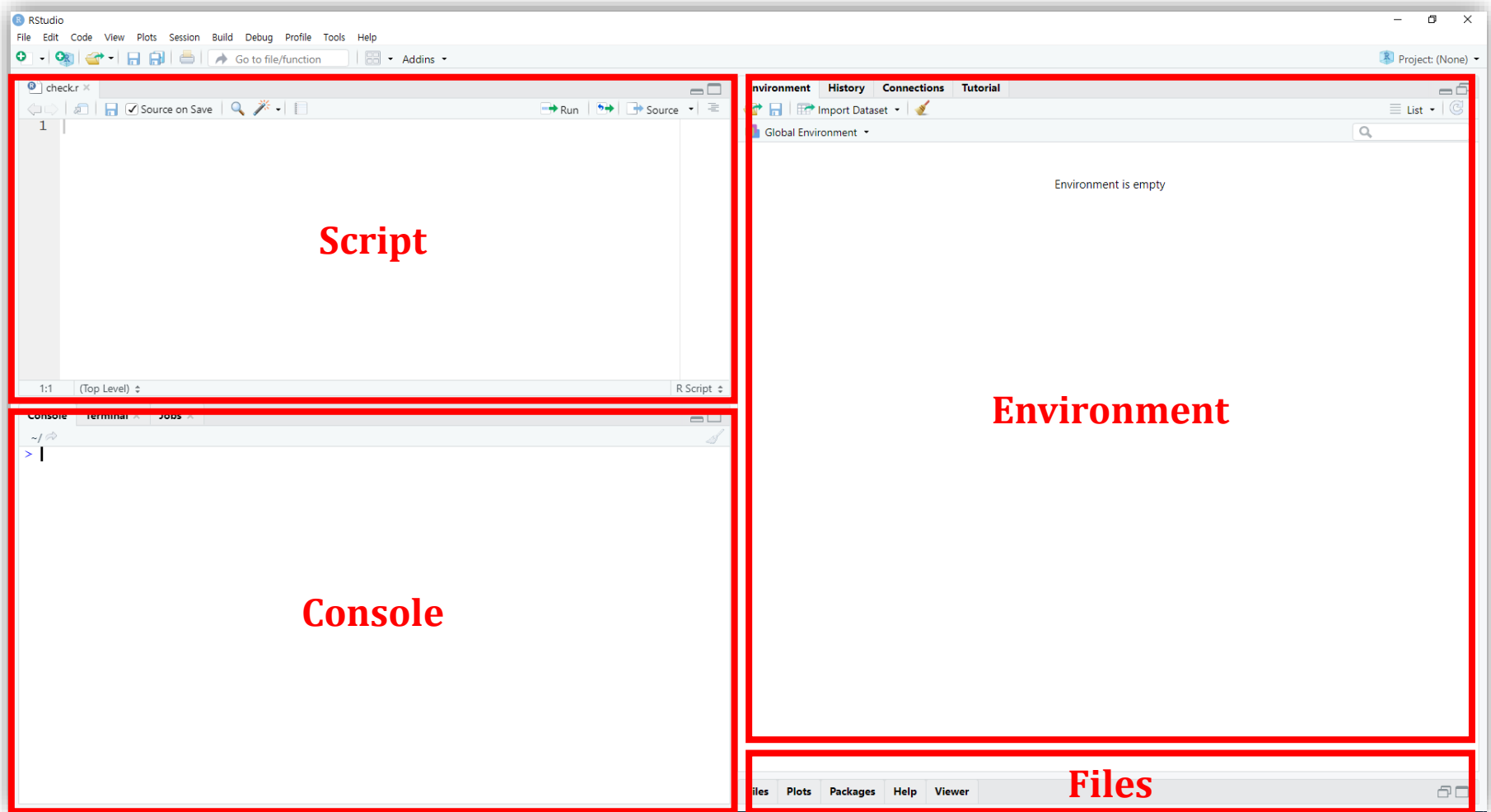
- 통합 개발 환경 IDE

- Rstudio: <http://www.rstudio.com>

R 설치

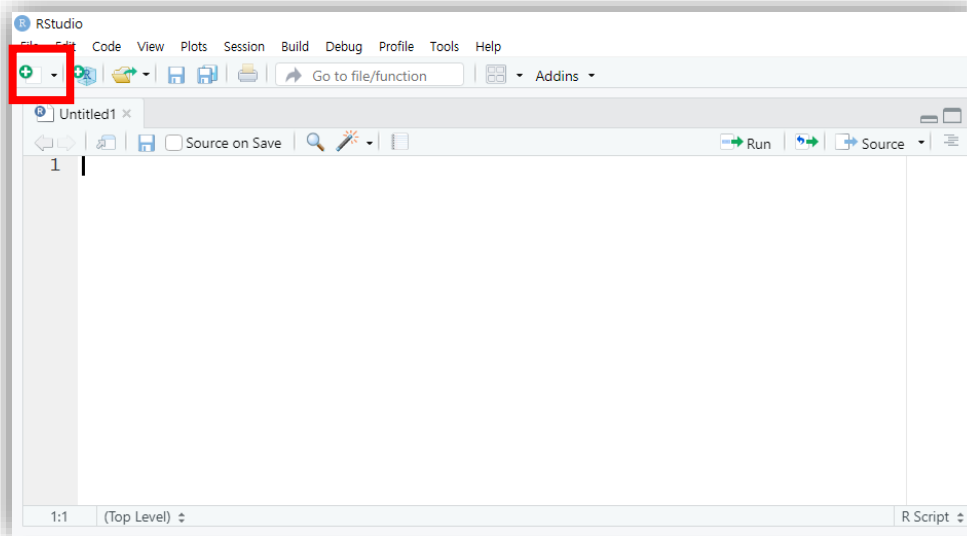
- Rstudio
 - R은 실행을 위한 메인 interpreter
 - 실제 코드는 Rstudio에서 작성 및 실행하여 이용
 - 운영체제에 맞는 Rstudio를 free 버전으로 다운로드
- 이외의 스크립트(R 코드) 작성 방법
 - Windows
 - 위 메뉴에서 파일 > 새 스크립트
 - R 편집기에서 코드 작성 (작성한 코드는 확장자 (.R)로 저장됨)
 - 작성된 코드를 모두 선택(Ctrl+A)하여 F5로 실행
 - R 터미널 내부에서 코드 실행됨
 - Mac
 - Vi 에디터 / 다른 에디터 툴 등을 이용하여 코드 작성
 - 작성된 코드를 확장자 (.R)로 저장
 - R 터미널 내부에서 > source("파일경로") 로 실행

Rstudio



Rstudio

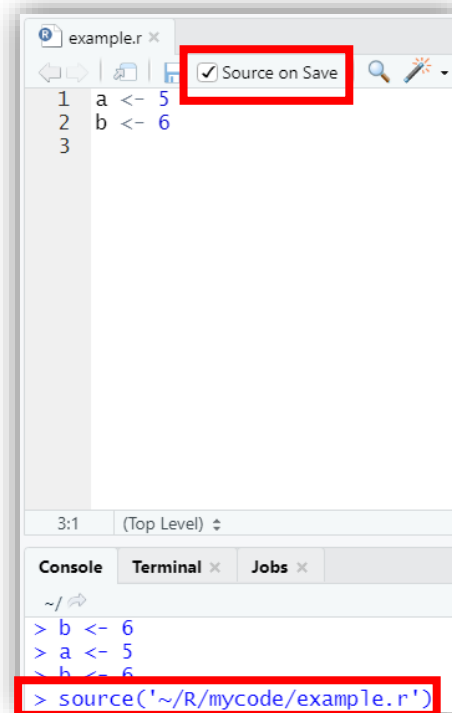
- 코드 작성하기
 - 새 script 파일(.r)을 생성
 - 작성 완료 후 실행
 - 아래 console 에서 출력 결과 확인
- 스크립트 파일 생성
 - File > New File > R script (Ctrl + Shift + N)



Rstudio

- 코드 실행하기

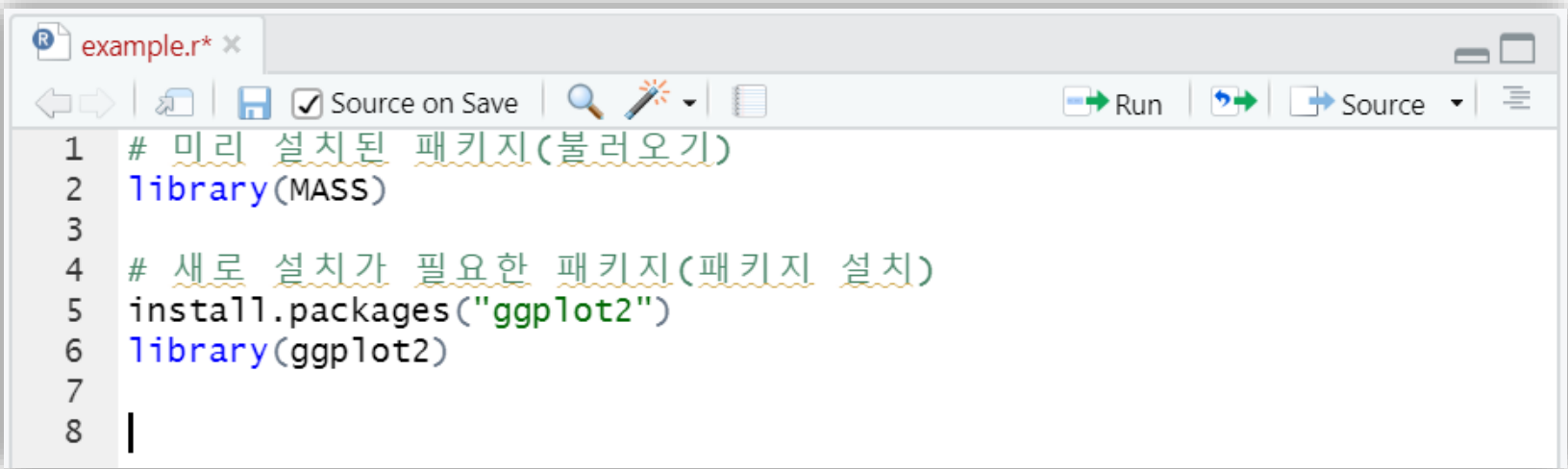
- Ctrl + Enter 를 통해 라인 별로 실행 가능
 - 해당 실행 라인이 console 로 복사되어 실행됨
- Ctrl + A 를 통해 모든 코드를 선택한 뒤 Ctrl + Enter 로 전체 실행
- 'Source on save' 에 체크 표시를 하면 저장할 때, source 함수로 실행










R 패키지

- R 패키지

- 패키지를 통해 다양한 데이터 및 구현된 함수를 불러올 수 있음
- 설치 시 자동으로 다운로드 되거나, 직접 다운로드 해야 함
- `install.packages(...)`
 - 필요한 패키지를 다운로드
- `library(...)`
 - 패키지 불러오기 (사용)

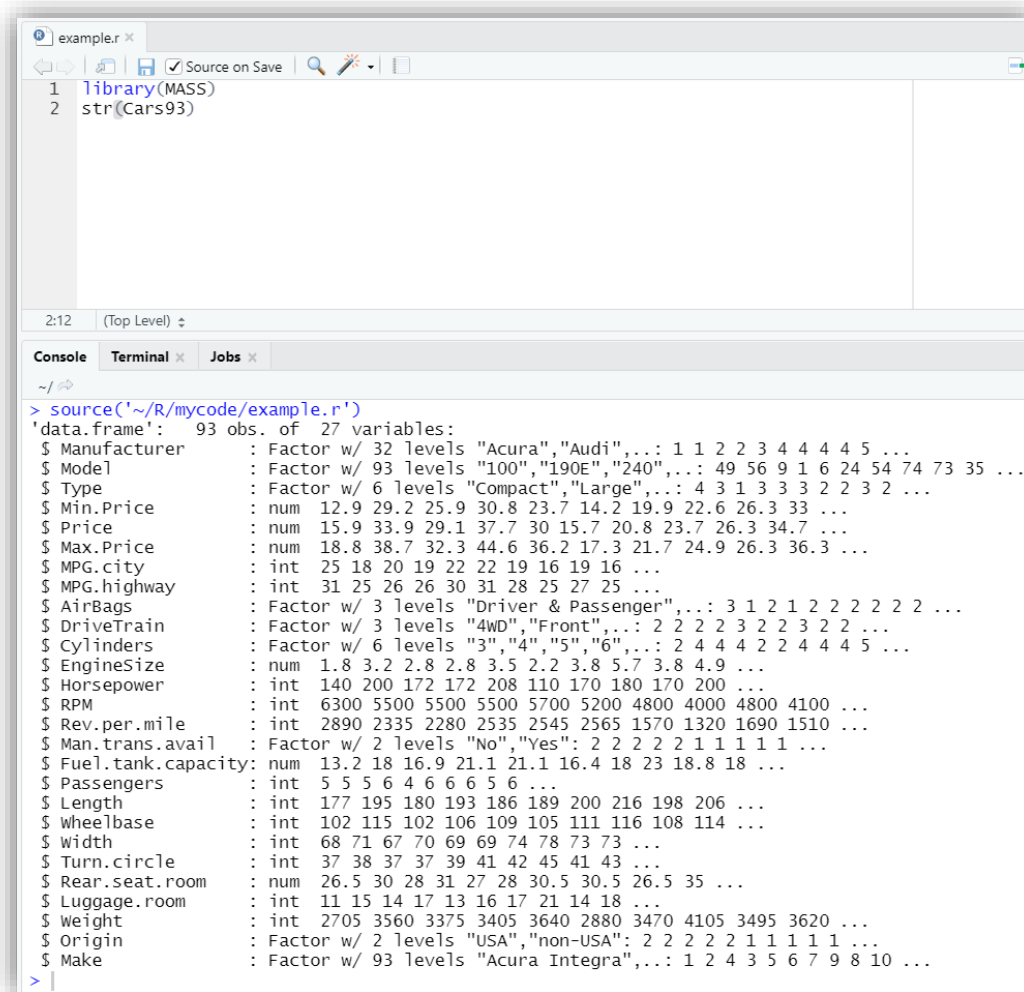


```
example.r* x
← → | ↻ |  ☒ Source on Save |   | 
 Run |  Source ▾ | 

1 # 미리 설치된 패키지(불러오기)
2 library(MASS)
3
4 # 새로 설치가 필요한 패키지(패키지 설치)
5 install.packages("ggplot2")
6 library(ggplot2)
7
8 |
```

R 패키지

- MASS::Cars93 데이터셋 예시



```
example.r x
1 library(MASS)
2 str(Cars93)

2:12 (Top Level) ⚡

Console Terminal Jobs x
~/
> source('~/.R/mycode/example.r')
'data.frame': 93 obs. of 27 variables:
 $ Manufacturer : Factor w/ 32 levels "Acura","Audi",...: 1 1 2 2 3 4 4 4 4 5 ...
 $ Model        : Factor w/ 93 levels "100","190E","240",...: 49 56 9 1 6 24 54 74 73 35 ...
 $ Type         : Factor w/ 6 levels "Compact","Large",...: 4 3 1 3 3 3 2 2 3 2 ...
 $ Min.Price    : num 12.9 29.2 25.9 30.8 23.7 14.2 19.9 22.6 26.3 33 ...
 $ Price       : num 15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ...
 $ Max.Price   : num 18.8 38.7 32.3 44.6 36.2 17.3 21.7 24.9 26.3 36.3 ...
 $ MPG.city    : int 25 18 20 19 22 22 19 16 19 16 ...
 $ MPG.highway : int 31 25 26 26 30 31 28 25 27 25 ...
 $ AirBags     : Factor w/ 3 levels "Driver & Passenger",...: 3 1 2 1 2 2 2 2 2 2 ...
 $ DriveTrain  : Factor w/ 3 levels "4WD","Front",...: 2 2 2 2 3 2 2 3 2 2 ...
 $ Cylinders   : Factor w/ 6 levels "3","4","5","6",...: 2 4 4 4 2 2 4 4 4 5 ...
 $ EngineSize  : num 1.8 3.2 2.8 2.8 3.5 2.2 3.8 5.7 3.8 4.9 ...
 $ Horsepower  : int 140 200 172 172 208 110 170 180 170 200 ...
 $ RPM         : int 6300 5500 5500 5500 5700 5200 4800 4000 4800 4100 ...
 $ Rev.per.mile : int 2890 2335 2280 2535 2545 2565 1570 1320 1690 1510 ...
 $ Man.trans.avail : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 1 1 1 1 ...
 $ Fuel.tank.capacity: num 13.2 18 16.9 21.1 21.1 16.4 18 23 18.8 18 ...
 $ Passengers   : int 5 5 5 6 4 6 6 6 5 6 ...
 $ Length      : int 177 195 180 193 186 189 200 216 198 206 ...
 $ Wheelbase   : int 102 115 102 106 109 105 111 116 108 114 ...
 $ Width       : int 68 71 67 70 69 69 74 78 73 73 ...
 $ Turn.circle : int 37 38 37 37 39 41 42 45 41 43 ...
 $ Rear.seat.room : num 26.5 30 28 31 27 28 30.5 30.5 26.5 35 ...
 $ Luggage.room : int 11 15 14 17 13 16 17 21 14 18 ...
 $ Weight      : int 2705 3560 3375 3405 3640 2880 3470 4105 3495 3620 ...
 $ Origin      : Factor w/ 2 levels "USA","non-USA": 2 2 2 2 2 1 1 1 1 1 ...
 $ Make       : Factor w/ 93 levels "Acura Integra",...: 1 2 4 3 5 6 7 9 8 10 ...
>
```

데이터 다루기

- 외부파일에서 데이터 불러오기

- `read.csv("파일경로")`

- R에서는 CSV, EXCEL, SPSS, SAS 등으로 저장된 외부 데이터를 불러올 수 있음
 - 일반적인 테이블 형태의 데이터 (행과 열이 존재하는 데이터) 모양으로 저장
 - "파일경로" 입력을 위해 `setwd(...)` 함수를 통해 working directory 변경이 가능

Number	Name	Score
1	Sam	90
2	Casey	97
3	Victor	57
4	William	88
5	John	62
6	Kim	41
7	Ryan	56
8	Krystal	77
9	Rem	84
10	Luke	31



```
> data <- read.csv("example.csv")
> data
  Number      Name Score
1       1       Sam   90
2       2     Casey   97
3       3    Victor   57
4       4  William   88
5       5     John   62
6       6      Kim   41
7       7     Ryan   56
8       8   Krystal   77
9       9      Rem   84
10      10     Luke   31
```

데이터 다루기

- 외부파일로 데이터 내보내기
 - write.csv(저장할 변수, "저장경로")
 - R 프로그램 내부에서 다루고 있는 데이터를 다양한 파일로 저장 가능
 - 또는 R 프로그램에서 사용하는 데이터(.RData)를 작업공간에 둘 수 있음
- Sequence 생성
 - num1:num2
 - num1부터 num2까지의 일련의 숫자 sequence 생성

```
> data <- 5:30
> data
[1] 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
> write.csv(data, "example2.csv")
```



	x
1	5
2	6
3	7
4	8
5	9
6	10
7	11
8	12
9	13
10	14
11	15
12	16
13	17
14	18
15	19
16	20
17	21
18	22
19	23
20	24
21	25
22	26
23	27
24	28
25	29
26	30

R의 자료형

- 변수 생성 및 접근

- R에서는 각 라인 뒤에 세미콜론(;)을 붙이지 않아도 됨
 - 한 줄에 여러 실행문을 사용하는 경우에는, 세미콜론으로 구분 가능
- 변수에 데이터를 저장(생성)하는 과정에서 자동 선언됨
 - 변수에 데이터를 저장할 때는 '=' 대신 '<-'를 사용
 - 일시적 동일함을 나타내는 것과 할당하는 것의 구분을 위하여 '='는 함수 매개변수 입력 등에서 사용하고, 실질적인 값의 할당은 '<-'를 이용
- 다음과 같은 변수명들은 허용되지 않음
 - 숫자로 시작하는 변수 이름, (ex) 2v
 - .(점), 숫자가 이어지는 변수 이름, (ex) .2v
 - 하이픈이 포함된 변수 이름, (ex) v-v

R의 자료형

```
> x1 <- 30
> y1 <- 1:100
> x1
[1] 30
> y1
  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 $
 [42] 42 43 44 45 46 47 48 49 50 51 52 53 54 55 $
 [83] 83 84 85 86 87 88 89 90 91 92 93 94 95 96 $
> x2
에러: 객체 'x2'를 찾을 수 없습니다
> sum(1, x2=2)
[1] 3
> x2
에러: 객체 'x2'를 찾을 수 없습니다
> sum(1, x2<-2)
[1] 3
> x2
[1] 2
```

• 변수 생성 및 접근 예시

- x1, y1에 각각 값을 할당하고 출력

- 출력에서 [1], [42], [83] 등의 표시는 현재 배열(데이터의 행)에서 가장 앞에 있는 값이 전체에서 가지는 index를 표현 (1에서부터 시작)
- 즉, 가장 앞에 있는 값이 전체 데이터 배열 내에서 1, 42, 83번째 값이라는 뜻

R의 자료형

```
> x1 <- 30
> y1 <- 1:100
> x1
[1] 30
> y1
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 $
[42] 42 43 44 45 46 47 48 49 50 51 52 53 54 55 $
[83] 83 84 85 86 87 88 89 90 91 92 93 94 95 96 $
> x2
에러: 객체 'x2'를 찾을 수 없습니다
> sum(1, x2=2)
[1] 3
> x2
에러: 객체 'x2'를 찾을 수 없습니다
> sum(1, x2<-2)
[1] 3
> x2
[1] 2
```

- 변수 생성 및 접근 예시

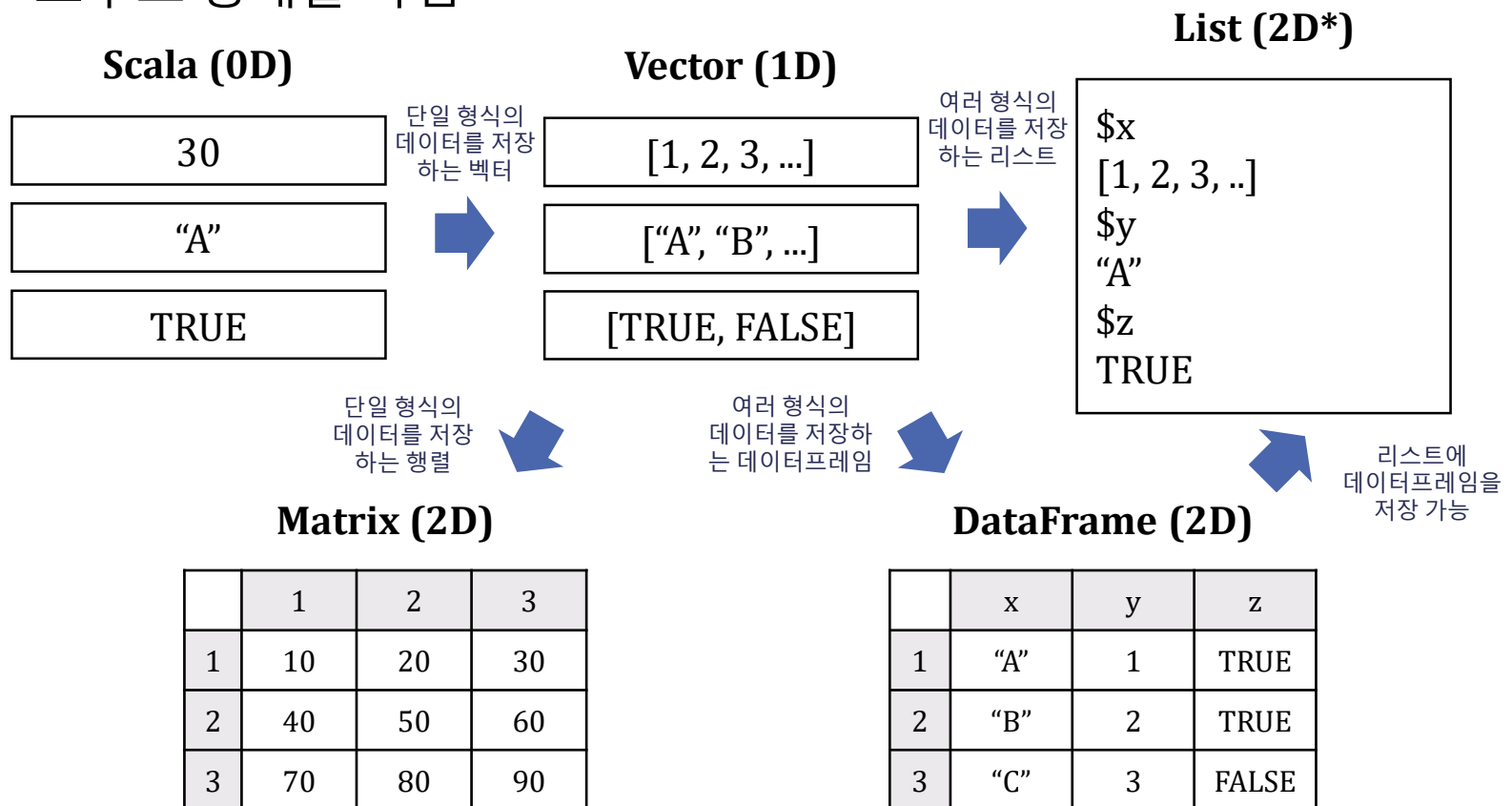
- x1, y1에 각각 값을 할당하고 출력

- '='를 이용하면 일시적인 할당을 수행하고 실제 값의 변화를 수행하지 않음
 - 반면, '<-'는 실질적으로 할당 또한 포함하고 있음

R의 자료형

- R의 자료형

- 다른 프로그래밍 언어들과 같이 '값'과 '여러 값'들을 저장할 수 있는 자료구조 형태를 가짐



R의 자료형

- Scala

- 자료형의 기본 단위
- 단일 자료형을 가지는 하나의 변수
- R에서는 사실 한 개의 원소를 가지는 벡터로 표현됨

```
> x1 <- 30
```

- 'x1'은 사실 하나의 원소 scala가 아니라, '30'이라는 값 하나만을 가진 벡터
- 즉, `x1 == x1[1] && x1[1] == 30`

R의 자료형

- 기본 자료형

- Numeric: 숫자 자료형
- Character: 문자열 자료형
- Logical(Bool): 미리 선언된 TRUE, FALSE 값
 - T, F로도 접근 및 사용 가능
- Factor: 범주 데이터 자료형 (데이터셋 내부 값들의 목록 = Levels)

- 특수한 자료형

- NA: 관측 내에서 결측된 값 (통계에서 '취합되지 못한 값'을 표현)
- NULL: 실제로 존재하지 않는 undefined 값 (데이터형과 길이가 없음)
- NaN: 수학적으로 표현이 불가능한 수
- Inf: 무한대 값 (infinite)

R의 자료형

```
> library(MASS)
> class(Cars93)
[1] "data.frame"
> cars93_manuf <- Cars93$Manufacturer
> nlevels(cars93_manuf)
[1] 32
> levels(cars93_manuf)
 [1] "Acura"      "Audi"      "BMW"      "Buick"      "Cadillac"  "Chevrolet"
 [7] "Chrysler"  "Chrysler"  "Dodge"    "Eagle"     "Ford"      "Geo"
[13] "Honda"     "Hyundai"   "Infiniti" "Lexus"     "Lincoln"   "Mazda"
[19] "Mercedes-Benz" "Mercury"  "Mitsubishi" "Nissan"    "Oldsmobile" "Plymouth"
[25] "Pontiac"    "Saab"     "Saturn"   "Subaru"    "Suzuki"    "Toyota"
[31] "Volkswagen" "Volvo"
> levels(cars93_manuf)[1]
[1] "Acura"
```

- MASS::Cars93 데이터셋을 이용한 Factor 자료형 예시
 - Manufacturer 변수컬럼은 Factor 자료형
 - DataFrame 자료형은 '\$'를 이용하여 변수명으로 자료에 접근 가능
 - 주어진 데이터 목록(범주) 내의 데이터가 저장되어 있음

R의 자료형

```
> library(MASS)
> class(Cars93)
[1] "data.frame"
> cars93_manuf <- Cars93$Manufacturer
> nlevels(cars93_manuf)
[1] 32
> levels(cars93_manuf)
 [1] "Acura"          "Audi"           "BMW"            "Buick"          "Cadillac"       "Chevrolet"
 [7] "Chrysler"       "Chrysler"       "Dodge"          "Eagle"          "Ford"           "Geo"
[13] "Honda"          "Hyundai"        "Infiniti"       "Lexus"          "Lincoln"        "Mazda"
[19] "Mercedes-Benz"  "Mercury"        "Mitsubishi"    "Nissan"          "Oldsmobile"     "Plymouth"
[25] "Pontiac"        "Saab"           "Saturn"         "Subaru"         "Suzuki"         "Toyota"
[31] "Volkswagen"     "Volvo"
> levels(cars93_manuf)[1]
[1] "Acura"
```

- MASS::Cars93 데이터셋을 이용한 Factor 자료형 예시
 - class(...)
 - 변수의 자료형 타입 반환
 - levels(...)
 - 변수의 범주 목록 반환
 - nlevels(...)
 - 변수의 범주 개수 반환

과제

- 없음