제5주 : 리스트

5.1. 리스트(list)

- 서로 다른 기본 자료형을 가질 수 있는 자료구조들의 모임
- 원소들이 서로 다른 기본자료형을 가질 수 있다.
- 생성방법

(1) 원소들이 이름이 없는 리스트 만들기

myList <- list(자료구조1, 자료구조2, ……)

(2) 원소들이 이름을 갖는 리스트 만들기

myList <- list(name1=자료구조1, name2=자료구조2, ……)

- 리스트의 원소들은 순서를 갖고 있으며, 위의 예에서 첫 번째 리스트의 경우 각 순서를 list() 함수 안에 집어 넣은 순서대로 갖습니다. 두번째 예의 경우에도 순서는 list() 함수 안에 배치한 순서로 결정되지만 이름을 주어 나타낼 수 있습니다. 사용 예를 통해 자세히 살펴보겠습니다.

- 사용예 및 접근방법

```
> title <- "My List"
> ages <- c(31, 41, 21)
> numbers <- matrix(1:9, nrow=3)
> names <- c("Baby", "Gentle", "none")
> listEx <- list(title, ages, numbers, names)
```

➔ 순서대로 title, ages, numbers, names의 값들이 순서대로 묶인 list를 생성합니다.

```
> listEx
[[1]]
[1] "My List"
[[2]]
[1] 31 41 21
```

```
[[3]]
[,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
[[4]]
[1] "Baby"   "Gentle" "none"
```

➔ 리스트에 포함된 하위 원소들이 순서대로 저장됩니다.
하지만 list로 묶인 각 값들에는 이름이 없습니다.

➔ 각 원소들에 접근하기 위해서는 대괄호 두 개([[]사이에 숫자로 지정합니다.(자료값 접근방법 1)

```
> listEx[[1]]
[1] "My List"
```

➔ 해당 리스트의 첫 번째 원소를 가져옵니다.

```
> listEx2 <- list(title=title, age=ages, number=numbers, name=names)
```

➔ 리스트를 구성하는 자료구조에 이름을 주어 배치합니다. 순서는 앞선 예제와 똑같지만 실제사용에 있어 이름을 지정하면 순서보다 이름을 통해 값을 가져오는 것이 일반적입니다.

```
> listEx2
$title
[1] "My List"
$age
[1] 31 41 21
$number
[,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
$name
[1] "Baby"   "Gentle" "none"
```

```
> listEx2[[1]]
[1] "My List"
```
➔ 첫번째 원소의 값을 가져옵니다. 원소들 간에 순서가 있으므로, 값을 불러오기 위해 대괄호 두 개를 사용했습니다: *리스트이름*[[*원소의위치*]]

```
> listEx2$title
[1] "My List"
```
➔ 원소들 각각에 이름을 지정한 경우, 원소의 값을 불러오기 위해 리스트 자료명 뒤에 달러표시($)를 붙이고 지정한 이름을 적어 해당 자료를 가져올 수 있습니다. (자료값 접근 방법 2)

```
> listEx2$age
[1] 31 41 21
> listEx2$number
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> listEx2$name
[1] "Baby"   "Gentle" "none"
```

- 같이 사용할 수 있는 함수(1)
  - is.list(x): 주어진 자료가 list 자료구조이면 TRUE 그렇지 않으면 FALSE
  - as.list(x): 주어진 자료를 list로 변환하는 함수

```
> x <- list(c(1,2,3,4), c(3, 2, 1))
> v <- c(1, 2, 3, 4)
> is.list(x)
[1] TRUE
```
➔ x가 리스트이므로 TRUE 반환

```
> is.list(v)
[1] FALSE
```
➔ x가 리스트가 아니므로 FALSE 반환

```
> v_list <- as.list(v)
> v_list
```

```
[[1]]
[1] 1
[[2]]
[1] 2
[[3]]
[1] 3
[[4]]
[1] 4
```
➔벡터v를 리스트로 변환 시 각 원소가 리스트의 원소로 변환된다.

- 같이 사용할 수 있는 함수(2)
  - length(x): 리스트 x에 포함된 원소들의 개수를 반환
  - names(x): 리스트 x에 포함된 원소들의 이름을 문자벡터형태로 반환.
  - names(x) <- : 리스트 x에 포함된 원소들의 이름을 설정해줄 수 있다.

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

[[2]]
[1] 3 2 1
> length(x)
[1] 2
```
➔ x의 원소의 개수는 2이다.
```
> names(x)
[1] NULL
```
➔ x의 원소들이 이름이 없으므로 NULL을 반환한다.
```
> names(x) <- c("ID", "Height")
> x
$ID
[1] 1 2 3 4
```

```
$Height
[1] 3 2 1
```
➔ 리스트x의 각 원소에 이름을 부여한다.

- 같이 사용할 수 있는 함수(3): lapply(), sapply()

```
lapply(X, FUN, ...)

sapply(X, FUN, ...,
       simplify = TRUE, USE.NAMES=TRUE)
```

- 리스트의 각 원소들마다 함수FUN을 적용한다.
- X: 적용할 리스트
- FUN: 적용할 함수
- ...: FUN 함수에 추가적으로 전달되는 전달인자
- simplify: TRUE이면 결과를 스칼라로 반환하고, FALSE이면 리스트형으로 반환한다. 기본값은 TRUE
- USE.NAMES: TRUE이고 X가 문자열로 구성되어 있을 때 결과의 이름이 정해지지 않으면 X를 이름으로 결과 반환

```
> x <- list(a = 1:10, beta = exp(-3:3),
+           logic = c(TRUE, FALSE, FALSE, TRUE))
> lapply(x, mean)
$a
[1] 5.5
$beta
[1] 4.535125
$logic
[1] 0.5
```
➔ 리스트의 각 원소에 대하여 평균을 구하고 결과를 리스트로 반환
```
> lapply(x, sample, 3, replace = TRUE)
$a
[1] 4 6 5
```

```
$beta
[1] 2.7182818 0.3678794 7.3890561

$logic
[1] FALSE FALSE  TRUE
```

➔ 리스트의 각 원소에 대하여 sample(*, 3, replace = TRUE)을 통해 복원 추출 결과를 리스트로 반환

```
> sapply(x, mean)
a    beta   logic
5.500000 4.535125 0.500000
```

➔ 리스트의 각 원소에 대하여 평균을 구하고 결과를 벡터로 반환

##**********##
##***개별/조별 활동: sapply()의 다른 전달인자의 역할을 조사하시오.**##
##**********##

(1) 전달인자 … 의 역할을 설명할 수 있는 예제를 만들어 설명하시오.

(2) 전달인자 simplify= 의 역할을 설명할 수 있는 예제를 만들어 설명하시오.