R의 자료구조

R에서 5가지 자료구조가 있다:

- 벡터(vector)
- 행렬(matrix)
- 배열(array)
- 리스트(list)
- 데이터프레임(data.frame)

Vector는 한 가지 타입의(숫자형 또는 문자형)의 데이터만 입력이 가능하고, 동일한 데이터 타입의 Vector가 여러 개 있는 것이 Matrix 및 Array(배열)이고, 한 가지 유형의 데이터로만 구성되는 Vector와 달리 여러 가지 유형의 데이터를 동시에 가질 수 있는 데이터 구조가 바로 List이다. 그리고 상이한 데이터 유형의 Vector가 여러 개 있는 것이 Data.frame이라고 보면 된다.

각각에 대해서 살펴보도록 하자.

1. 벡터(vector)

c( )는 Combind의 약자를 나타내는 명령어로, 벡터를 만드는데 사용된다. 여기서 벡터라고 하면, 데이터에서 하나의 '열(Column)'을 의미한다.

① 벡터는 c( ) 안에 값을 나열하는 방법으로 생성하거나, scan( ) 함수를 이용하면 직접 데이터를 입력하여 생성할 수 있다.
다음 코드에서 x <- c(…) 부분이 다시 괄호( ) 안에 쌓여 있음에 주목하기 바란다. 이렇게 괄호( )로 코드를 묶으면 괄호 안의 문장을 수행하고, 그 결과 값을 화면에 출력한다.

```
x <- c(1, 2, 3, 4, 5)                  # 변수 x에 벡터 c(1,2,3,4,5)를 할당한다.
x
[1] 1 2 3 4 5
c(1, 2, 3, 4, 5)
[1] 1 2 3 4 5
assign("x", c(1,2,3,4,5)) ; x # 변수 x에 벡터 c(1,2,3,4,5)를 할당한다.
[1] 1 2 3 4 5
```

```
(x <- c(1, 2, 3, 4, 5))      # 변수 x에 벡터 1,2,3,4,5를 넣는다.
[1] 1 2 3 4 5
print(x)                     # 변수 x에 저장된 데이터를 출력하라.
```

② 벡터는 동일한 데이터 유형으로 구성된 데이터 집합. 만일 서로 다른 유형의 데이터를 섞어서 벡터에 저장하면, 이들 데이터는 한 가지 유형으로 자동 변환된다.
예를 들어, 정수형과 문자형이 섞여 있다면 모두 문자형으로 변환된다. 예를 들어, 아래 코드에서 숫자형 데이터(numeric)인 2는 "2"라는 문자형로 자동으로 변환되어 x 안에는 **문자형 형태의 데이터**만 나열된다(문자형이 숫자형보다 우선한다).

logical(논리형) < integer(정수형) < double(실수형) < character(문자형)

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

③ 벡터는 중첩할 수 없다. 따라서 벡터 안에 벡터를 생성하면 단일 차원의 벡터로 변경된다. 중첩된 구조가 필요하다면 **리스트**(list)를 사용해야 한다.

```
c(1, 2, 3)
[1] 1 2 3
c(1, 2, 3, c(1, 2, 3))
[1] 1 2 3 1 2 3
list(1, 2, 3, c(1, 2, 3))
```

④ 벡터의 각 셀에는 names( ) <- c( ) 함수를 사용해 이름을 부여할 수 있다. names( )의 반환 값에 원하는 이름을 문자열 벡터로 할당하면 된다.

```
x <- c(79, 94, 80)
names(x) <- c("kim", "seo", "park")
x
kim seo park
79  94  80
```

벡터 데이터 접근

벡터의 데이터를 접근하는 데는 색인을 사용하는 방법과 이름을 사용하는 방법이 있다. 또, 벡터에서 특정 요소를 제외한 나머지 데이터를 가져오거나, 동시에 여러 셀의 데이터를 접근하는 것 역시 가능하다.

```
x <- c(1, -3, 0, -5, 2, 4, 0)
x
length(x)  # 벡터 x의 길이를 출력
x[3]       # x의 3번째 값을 출력하라.
x[n] : 벡터 x의 n번째 요소. n은 숫자 또는 셀의 이름을 뜻하는 문자열
x[c(1,3)]  # x의 1,3번째 값을 출력하라.
x[1,3]     # x[1,3]은 행렬 x의 1행3열 성분을 출력하는 명령어
Error in x[1, 3] : incorrect number of dimensions
x[2:4]     # x의 2번째 원소부터 4번째 원소까지 출력하라.
x[start:end] : 벡터 x의 start부터 end까지의 값을 반환함. 반환 값은 start 위치의 값과 end 위치의 값을 모두 포함함
x[-1]      # x의 첫 번째 원소를 빼고 출력하라.
x[c(-1,-3)]  # x의 첫 번째 원소와 세 번째 원소를 빼고 출력하라.
x[-1:-2]  # x의 첫 번째부터 세 번째 원소까지를 제외하고 나머지를 출력.
x[3:length(x)]  # 3번째 원소부터 마지막 원소까지 출력
x[-length(x)]   # 마지막 원소를 제거하고 싶다면...
x[x>5]   # x값이 5 보다 큰 원소들만 추려내서 출력하라는 코드
numeric(0)
x[x>=2]     # x값이 2 이상의 원소들만 추려내서 출력하라는 코드
which(x*x>8)    # x의 제곱, 즉 8보다 큰 9, 25, 16 원소들의 위치를 출력하라는 코드[1]
[1] 2 4 6  # 2번째, 4번째, 6번째
```

벡터의 정렬(오름차순/내림차순)

sort( )를 사용하면 간단하다. 만일, 내림차순으로 정렬하고 싶다면 [decreasing=TRUE] 옵션을 사용하자.

```
x <- c(-3,7,2,-1,0)
sort(x)
[1] -3 -1  0  2  7  # 오름차순(디폴트)
sort(x, decreasing = TRUE)
[1]  7  2  0 -1 -3
sort(-x)
[1] -7 -2  0  1  3 # 벡터의 반대부호로 하여 오름차순으로 정렬
```

1) which 의문대명사 (문장 첫머리에서) 어느 것, 어느 쪽, 어느 사람, (한 무리 중의) 누구《한정된 수의 것으로부터의 선택. 불특정의 것으로부터의 선택은 what임》

order( ) 함수만 단독으로 쓰는 경우, 각 값의 순위(rank)가 출력된다.[2)]

```
x <- c(2,5,3,1,4)
order(x)
[1] 4 1 3 5 2  # 각 값에 대한 순위값 출력
```

2) 데이터 프레임의 정렬에서 order( ) 함수