Part 1. R 프로그래밍 (데이터 분석 전문가 양성과정)

03

R의 기본문법

경북대학교 배준현 교수
(joonion@knu.ac.kr)

03. R의 기본문법

- R 언어의 기본문법
  - 변수: *variables*
  - 할당문: *assignment*
  - 연산자: *operators*
  - 조건문: *conditionals*
  - 반복문: *iterations*
  - 함수: *functions*

03. R의 기본문법

- **변수:** *variable*
  - 프로그래밍에서 변수는 어떤 데이터를 저장하는 저장소
  - <-: 할당 연산자 (*assignment operator*)
    - R에서는 <- 외에도 ->와 =를 할당 연산자로 사용할 수 있음

03. R의 기본문법

```
> x <- 10
> x
[1] 10

> 20 -> y
> y
[1] 20

> z = x + y
> z
[1] 30
```

✓ R에서는 할당문에 <- 를 사용할 것을 권장함

변수 이름의 규칙

- 첫 글자는 반드시 문자나 마침표로 시작해야 함
- 첫 글자 이후에는 문자, 마침표, 숫자, 밑줄을 사용할 수 있음
- 주의: 알파벳 대문자와 소문자를 구분함

```
> 1.val <- 200
Error: unexpected symbol in "1.val"

> var.1 <- 100
> Var.1
Error: object 'Var.1' not found
```

- **원시 자료형:** *primitive* data types

구분	자료형	리터럴
논리형	*logical*	TRUE FALSE T F
숫자형	*numeric*	1 2 3 4 5 … 3.14 1.414 …
문자형	*character*	"A" "B" "C" … "Hello" 'A' 'B' 'C' … 'Hello'
특수형		NA NULL NaN Inf -Inf

```
> class(TRUE)
[1] "logical"

> class(3)
[1] "numeric"

> class("Hello")
[1] "character"
```

```
> var <- TRUE
> class(var)
[1] "logical"

> var <- 3
> class(var)
[1] "numeric"

> var <- "Hello"
> class(var)
[1] "character"
```

■ 연산자: *operators*

구분	연산자	구분
산술 연산자	+ - * /	덧셈 뺄셈 곱셈 나눗셈 연산
	%% %/%	나머지, 몫 연산
	^ 또는 **	거듭제곱 연산
논리 연산자	< > <= >=	대소 비교 연산
	== !=	같은가, 다른가
	! & |	NOT AND OR (벡터를 리턴)
	&& ||	AND OR (단일값을 리턴)

03. R의 기본문법

```
> x <- 2
> y <- 2 * ((x+2)-(x+4)) / 2 + 3
> y
[1] 1

> 7 %/% 3
[1] 2
> 7 %% 3
[1] 1

> 10 ^ 2
[1] 100
> 2 ** 10
[1] 1024
```

```
> 3 < 4
[1] TRUE
> 5 > 7
[1] FALSE

> x <- 5
> y <- x
> x == y
[1] TRUE
> x != y
[1] FALSE

>!TRUE & TRUE | FALSE
[1] FALSE

> x <- 3
> (x > 3) && (x < 4)
[1] FALSE
```

- **조건문: *conditional* statement**
  - **어떤 조건에 따라 코드 블록의 실행 여부를 결정하는 구문**

```
if (condition) {
    condition.true.statements
}

if (condition) {
    condition.true.statements
} else {
    condition.false.statements
}
```

```
if (condition1) {
    condition1.true.statements
} else if (condition2) {
    condition2.false.statements
} else {
    other.statements
}
```

- **학생의 점수 score 에 따라 학점 grade 부여하기**

```
> score <- 91

> if (score >= 90) {
+     grade <- "A"
+ } else if (score >= 80) {
+     grade <- "B"
+ } else {
+     grade <- "F"
+ }

> grade
[1] "A"
```

✓ **주의: else는 반드시 } 가 있는 줄에 함께 있어야 함**

- **반복문:** *loop* statement (*iteration*)
  - **특정한 코드 블록이 여러 번 반복적으로 수행하도록 하는 구문**

```
while (condition) {
    condition.true.statement
}
```

```
for (variable in start:end) {
    loop.statements
}
```

- **반복문을 이용하여 1에서 10까지 자연수의 합 구하기**

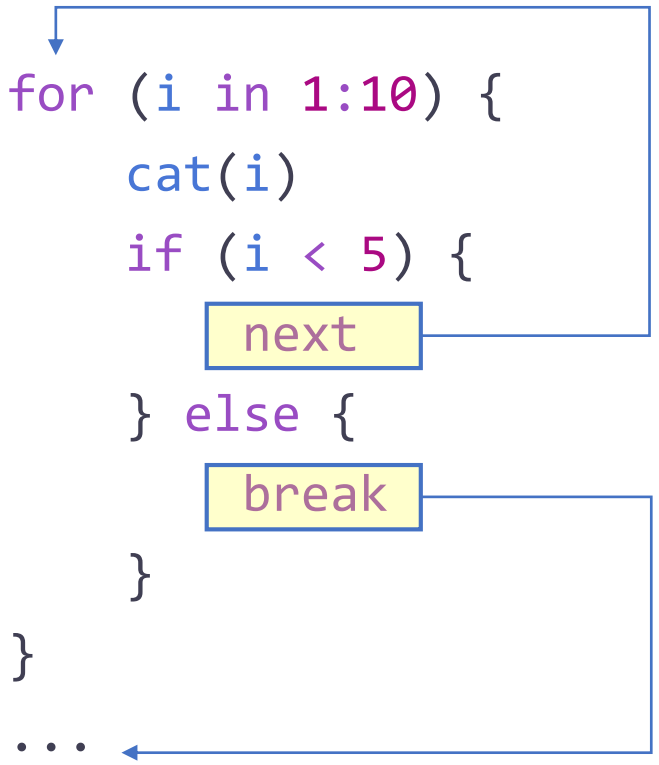
```
> s <- 0
> i <- 1
> while (i <= 10) {
+     s <- s + i
+     i <- i + 1
+ }
> s
[1] 55
```

```
> s <- 0
> for (i in 1:10) {
+     s <- s + i
+ }
> s
[1] 55
```

- 임의의 자연수 n에 대하여, n의 약수를 출력하고 약수의 개수 구하기

```
> n <- 32
> count <- 0
> for (i in 1:n) {
+     if (n %% i == 0) {
+         cat(i, " ")
+         count <- count + 1
+     }
+ }
1  2  4  8  16  32
> count
[1] 6
```

- break와 next: **반복문의 중단과 계속**
  - *break*: **현재 수행 중인 반복문을 중단하고 코드 블록을 빠져나감**
  - *next*: **현재 수행 중인 반복문의 코드 블록에서 다음 반복으로 되돌아감**



- 소수($prime$): 1과 자기 자신(n)으로만 나누어 떨어지는 1이 아닌 자연수
- 임의의 자연수 n이 소수인지 아닌지 판단하기

```
> n <- 17
> is.prime = TRUE
> for (i in 2:(n-1)) {
+     if (n %% i == 0) {
+         is.prime <- FALSE
+         break
+     }
+ }
> is.prime
[1] TRUE
```

- **중첩 반복문: *nested* for-loop**
  - **조건문이나 반복문은 여러 단계로 중첩될 수 있음**

```
> for (i in 1:3) {
+     cat(i, ": ")
+     for (j in 1:5) {
+         cat(j, " ")
+     }
+     cat("\n")
+ }
1 : 1  2  3  4  5
2 : 1  2  3  4  5
3 : 1  2  3  4  5
```

- **연습문제 3.1:**
  - 한 변의 길이가 x인 정사각형의 넓이 $area$를 구하는 수식을 만드시오.
    - x가 $5, 10, 15$일 때 $area$의 값을 확인해 보시오.
  - 반지름의 길이가 r인 원의 둘레 $round$와 넓이 $area$를 구하는 수식을 만드시오.
    - r이 $5, 10, 15$일 때 $round$와 $area$의 값을 확인해 보시오.

- **연습문제 3.2: 피자나라의 치킨공주**
  - 양의 정수 n의 값에 따라 $order$에 다른 값을 할당하시오.
    - n이 3의 배수이면 $order$는 "피자"
    - n이 5의 배수이면 $order$는 "치킨"
    - n이 3과 5의 배수이면 $order$는 "피자나라치킨공주"
    - n이 그 이외의 수이면 $order$는 "다이어트"
  - n 이 6, 10, 13, 15일 때 $order$ 의 값을 확인하시오.

- **연습문제 3.3:**
  - 임의의 자연수 n에 대하여
    - $S = 1 + 2^3 + 3^3 + \cdots + (n-1)^3 + n^3$를 구하는 코드를 작성하시오.
    - n이 10, 15, 20일 때 S의 값을 확인하시오.
  - 임의의 자연수 n에 대하여
    - $n!$을 구하는 코드를 작성하시오.
    - n이 10, 15, 20일 때 $n!$의 값을 확인하시오.

- **연습문제 3.4:**
  - 피자나라의 치킨공주 문제에서 임의의 자연수 n이 주어지면
    - 1부터 n까지 $order$를 아래 예시($n = 15$)와 같이 출력하시오.
    - 치킨, 피자, 피자나라치킨공주, 다이어트의 횟수를 각각 아래와 같이 출력하시오.

```
1 다이어트
2 다이어트
3 피자
4 다이어트
5 치킨
6 피자
7 다이어트
8 다이어트
9 피자
10 치킨
11 다이어트
12 피자
13 다이어트
14 다이어트
15 피자나라치킨공주
```

```
> cat("pizza =", pizza, "\n")
pizza = 4
> cat("chicken =", chicken, "\n")
chicken = 2
> cat("combo =", combo, "\n")
combo = 1
> cat("diet =", diet, "\n")
diet = 8
```

- **연습문제 3.5:**
  - 임의의 홀수 n에 대하여
    - 이중 for-loop를 이용하여 아래와 같이 별(*) 모양을 찍어보시오.
    - 아래 예시는 $n = 5$일 때이다.

```
*****        *        *****
*****       **        *
*****      ***       *****
*****     ****       *
*****    *****       *****
```

- **연습문제 3.6:**
  - 임의의 자연수 n에 대하여 1부터 n까지의 수 중에서
    - 소수를 모두 출력하고,
    - 소수의 개수를 구하는 코드를 작성하시오.
  - n이 10, 100, 1000일 때 소수와 소수의 개수를 확인하시오.

```
2 3 5 7
```

```
> count
[1] 4
```

- **연습문제 3.7:**
  - 1부터 n까지의 자연수에 대하여
    - 약수의 개수를 모두 출력하고,
    - 약수의 개수가 가장 많은 숫자를 찾는 코드를 작성하시오.
    - 단, 약수의 개수가 가장 많은 숫자가 여러 개이면 가장 큰 수를 찾으시오.
    - n이 10, 100, 1000일 때 약수의 개수가 가장 많은 숫자를 확인하시오.

```
1 : 1
2 : 2
3 : 2
4 : 3
5 : 2
6 : 4
7 : 2
8 : 4
9 : 3
10 : 4
```

```
> maxval
[1] 10
```

Any Questions?