



문공 A0015

R 프로그래밍

김 태 완

kimtwan21@dongduk.ac.kr

함수

- apply() 함수

- apply() 함수의 개념

: 매트릭스나 데이터프레임에 있는 '행들이나 열들'을 차례로 꺼내어 작업을 수행하고자 할 때 사용

대상 매트릭스나 데이터프레임의 이름

apply(데이터셋, 행/열방향 지정, 적용 함수)

행 방향 작업 = 1, 열 방향 작업 = 2

1	2	3	4
5	6	7	8
9	10	11	12

행 방향 계산
출력값 -> 3개

열 방향 계산
출력값 -> 4개

apply() 함수

사용자 정의 함수

데이터 위치 찾기

함수

- apply() 함수
 - apply() 함수의 적용

```
> apply(iris[, 1:4], 1, mean)
```

```
# iris 데이터프레임의 1열부터 4열까지 행방향으로 평균을 계산
```

Sepal.Length	Sepal.width	Petal.Length	Petal.width	
5.1	3.5	1.4	0.2	mean()
4.9	3.0	1.4	0.2	mean()
4.7	3.2	1.3	0.2	
4.6	3.1	1.5	0.2	
5.0	3.6	1.4	0.2	
5.4	3.9	1.7	0.4	
4.6	3.4	1.4	0.3	
5.0	3.4	1.5	0.2	
4.4	2.9	1.4	0.2	
4.9	3.1	1.5	0.1	
5.4	3.7	1.5	0.2	
4.8	3.4	1.6	0.2	
4.8	3.0	1.4	0.1	
4.3	3.0	1.1	0.1	
5.8	4.0	1.2	0.2	mean()

apply() 함수

사용자 정의 함수


데이터 위치 찾기

함수

- apply() 함수
 - apply() 함수의 적용

```
> apply(iris[, 1:4], 2, mean)
```

```
# iris 데이터프레임의 1열부터 4열까지 열방향으로 평균을 계산
```



Sepal.Length	Sepal.width	Petal.Length	Petal.width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
5.0	3.6	1.4	0.2
5.4	3.9	1.7	0.4
4.6	3.4	1.4	0.3
5.0	3.4	1.5	0.2
4.4	2.9	1.4	0.2
4.9	3.1	1.5	0.1
5.4	3.7	1.5	0.2
4.8	3.4	1.6	0.2
4.8	3.0	1.4	0.1
4.3	3.0	1.1	0.1
5.8	4.0	1.2	0.2
mean()	mean()	mean()	mean()

apply() 함수

사용자 정의 함수

데이터 위치 찾기

함수

- apply() 함수
 - apply() 함수의 적용

```
a <- matrix(1:20, 4,5)
```

```
apply(a,1,mean)
```

```
[1] 9 10 11 12
```

함수

- 사용자 정의 함수
 - 예시 : 약수 구하기

```
a ← 24

for (i in 1:a) {
  if (a%%i == 0){
    cat(i, end="")
  }
}
```

```
a ← 10

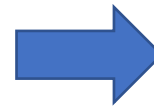
for (i in 1:a) {
  if (a%%i == 0){
    cat(i, end="")
  }
}
```

```
a ← 36

for (i in 1:a) {
  if (a%%i == 0){
    cat(i, end="")
  }
}
```

```
a ← 158

for (i in 1:a) {
  if (a%%i == 0){
    cat(i, end="")
  }
}
```



```
yaksu ← function(a){
  for (i in 1:a) {
    if (a%%i == 0){
      cat(i, end="")
    }
  }
}

yaksu(24)
yaksu(10)
yaksu(36)
yaksu(158)
```

함수

- 사용자 정의 함수 : 자신만의 함수를 만들어 사용할 수 있는 기능
 - 사용자 정의 함수 만들기

```
함수명 ← function(매개변수 목록) {  
  실행할 명령문(들)  
  return(함수의 실행 결과)  
}
```

- **함수명** : 사용자 정의 함수의 이름으로 사용자가 만들 수 있다.
- **매개변수 목록** : 함수에 입력할 매개변수 이름을 지정한다.
- **실행할 명령문(들)** : 함수에서 처리하고 싶은 내용을 작성한다.
- **함수의 실행 결과** : 함수의 실행 결과를 반환하며, 반환 결과가 없으면 return() 함수를 생략한다.

함수

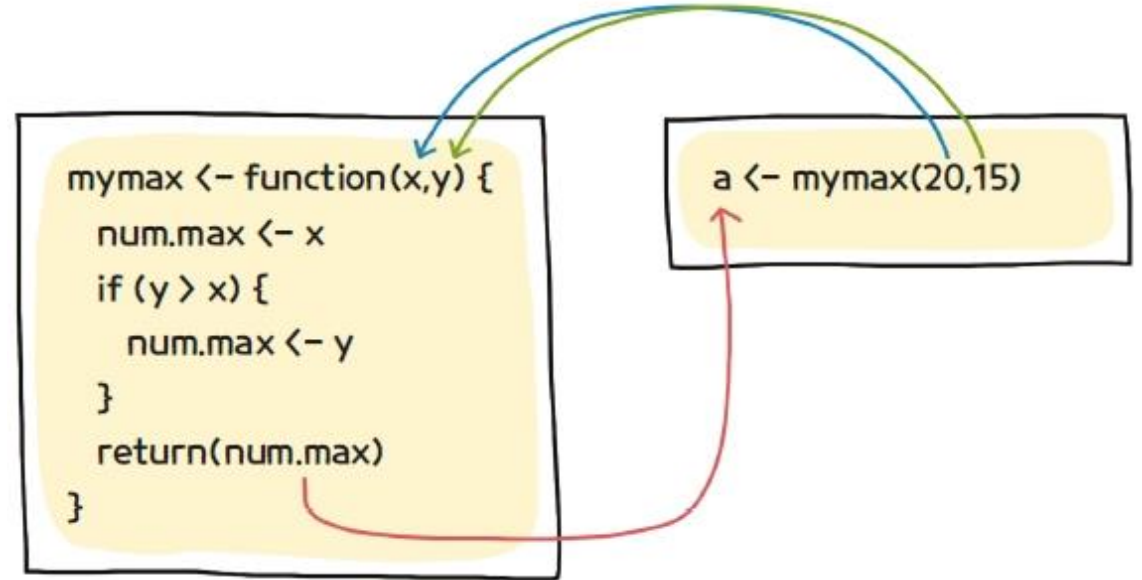
- 사용자 정의 함수 : 자신만의 함수를 만들어 사용할 수 있는 기능
 - 사용자 정의 함수 만들기
 - 사용자 정의 함수를 만들고 사용하기 : 두 개의 값을 입력 받아 큰 수를 결과값으로 돌려주는 함수

```
mymax ← function(x, y) {  
  num.max ← x  
  if (y > x) {  
    num.max ← y  
  }  
  return(num.max)  
}
```


함수

- 사용자 정의 함수 : 자신만의 함수를 만들어 사용할 수 있는 기능
 - 사용자 정의 함수 만들기
 - 사용자 정의 함수를 만들고 사용하기 : 두 개의 값을 입력 받아 큰 수를 결과값으로 돌려주는 함수

```
mymax(10, 15)  
  
a ← mymax(20, 15)  
  
b ← mymax(31, 45)  
  
print(a+b)
```



함수

- 사용자 정의 함수 : 자신만의 함수를 만들어 사용할 수 있는 기능
 - 사용자 정의 함수의 매개변수에 초기값 설정하기

```
mydiv ← function(x, y=2){  
  result ← x/y  
  return(result)  
}
```

```
mydiv(x=12, y=3)
```

```
mydiv(12,3)
```

```
mydiv(12)
```

함수

- 사용자 정의 함수 : 자신만의 함수를 만들어 사용할 수 있는 기능
 - 함수가 반환하는 결과값이 여러 개일 때의 처리 : list() 함수를 이용하여 결과값을 하나로 묶음

```
myfunc ← function(x, y) {  
  val.sum ← x+y  
  val.mul ← x*y  
  return(list(sum = val.sum, mul = val.mul))  
}
```

```
result ← myfunc(5, 8)  
s ← result$sum  
m ← result$mul
```

```
cat('5+8=',s,'\n')  
cat('5*8=',m,'\n')
```

함수

- 사용자 정의 함수 : 자신만의 함수를 만들어 사용할 수 있는 기능
 - 사용자 정의 함수의 저장 및 호출
 - 자주 사용하게 될 사용자 정의 함수는 파일에 따로 모아두었다가 필요시 호출

```
myfunc ← function(x, y) {  
  val.sum ← x+y  
  val.mul ← x*y  
  return(list(sum = val.sum, mul = val.mul))  
}
```

myfunc.R



```
source('myfunc.R')  
result <- myfunc(5, 8)  
s <- result$sum  
m <- result$mul  
cat('5+8=',s,'\n')  
cat('5*8=',m,'\n')
```

main.R

함수

- 데이터 위치 찾기 : `which()`, `which.max()`, `which.min()` 함수 이용

함수	설명
<code>which()</code>	찾고자 하는 값의 인덱스(위치)를 알아내는 함수
<code>which.max()</code>	최댓값의 인덱스(위치)를 알아내는 함수
<code>which.min()</code>	최솟값의 인덱스(위치)를 알아내는 함수

```
score <- c(76, 84, 69, 50, 95, 60, 82, 71, 88, 84)
which(score==69)      # 성적이 69인 학생은 몇 번째에 있나?
which(score>=85)      # 성적이 85 이상인 학생은 몇 번째에 있나?
max(score)            # 최고 점수는 몇 점인가?
which.max(score)       # 최고 점수는 몇 번째에 있나?
min(score)             # 최저 점수는 몇 점인가?
which.min(score)       # 최저 점수는 몇 번째에 있나?
```

함수

- 데이터 위치 찾기 : `which()`, `which.max()`, `which.min()` 함수 이용
 - 벡터에서의 활용

```
> score <- c(76, 84, 69, 50, 95, 60, 82, 71, 88, 84)
> which(score==69)                # score 벡터에서 값이 69인 데이터의 위치
[1] 3
> which(score>=85)                # score 벡터에서 값이 85 이상인 데이터의 위치
[1] 5 9
> max(score)                      # score 벡터의 최댓값
[1] 95
> which.max(score)                # score 벡터의 최댓값의 위치
[1] 5
> min(score)                      # score 벡터의 최솟값
[1] 50
> which.min(score)                # score 벡터의 최솟값의 위치
[1] 4
```

함수

- 데이터 위치 찾기 : which(), which.max(), which.min() 함수 이용
 - 벡터에서의 활용

```
> score ← c(76, 84, 69, 50, 95, 60, 82, 71, 88, 84)
> idx ← which(score<=60)
> score[idx] ← 61
> score
[1] 76 84 69 61 95 61 82 71 88 84

> idx ←which(score >=80)
> score.high ← score[idx]
> score.high
[1] 84 95 82 88 84
```

함수

- 데이터 위치 찾기 : `which()`, `which.max()`, `which.min()` 함수 이용
 - 매트릭스/데이터프레임에서의 활용 : 행의 인덱스(위치)가 출력됨

```
> idx ← which(iris$Petal.Length>5.0) # iris에서 Petal.Length열의 값들 중 5.0보다 큰 데이터의 위치
> idx                                # 행의 값이 출력됨
[1] 84 101 102 103 104 105 106 108 109 110 111 112 113 115 116 117 118 119 121
[20] 123 125 126 129 130 131 132 133 134 135 136 137 138 140 141 142 143 144 145
...
> iris.big ← iris[idx, ]              # iris에서 Petal.Length열의 값들 중 5.0보다 큰
> iris.big                           데이터의 값들을 iris.big에 저장
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
84           6.0         2.7          5.1         1.6 versicolor
101          6.3         3.3          6.0         2.5  virginica
...
```


예시

- 예시 1 : apply() 함수 이용해서 mtcars 데이터셋 분석하기
 - 실행 결과

```
apply( , MARGIN=2, FUN=mean) # mtcars 데이터셋의 열별 합계
apply( , MARGIN=1, FUN=mean) # mtcars 데이터셋의 행별 평균
apply( , MARGIN=2, FUN=max)  # mtcars 데이터셋의 열별 최댓값
```

```
mpg      cyl      disp      hp      ...
642.900 198.000 7383.100 4694.000 ...
```

```
Mazda RX4   Mazda RX4 Wag Datsun 710 ...
 29.90727   29.98136      23.59818    ...
...
```

```
mpg      cyl      disp      hp      ...
33.900  8.000  472.000  335.000 ...
```

예시

- 예시 2 : 직삼각형의 빗변의 길이를 구하는 tri() 함수 만들기
 - 실행 결과

```
tri <- {  
  z <-  
    
}
```

```
tri(3,4)
```

```
[1] 5
```

```
tri(12,5)
```

```
[1] 13
```

```
# 함수의 매개변수 2개 필요  
# 빗변의 길이 = z  
# z를 반환
```

예시

- 예시 3 : 두 정수의 최대공약수를 구하는 함수 `lcm()` 만들기
 - 실행 결과

```
lcm ← {  
  res ← 0  
  for {  
    if {  
      res = i  
    }  
  }  
  return(res)  
}
```

```
lcm(10, 8)
```

```
[1] 2
```

```
lcm(10, 20)
```

```
[1] 10
```

함수의 매개변수 2개 필요(두 정수)
최대공약수를 저장할 res값을 미리 설정
for문을 이용
조건 : 최대공약수: x의 약수이면서(동시에)
y의 약수인 값의 최대값

예시

- 예시 4 : 벡터의 최대값과 최소값을 구하는 maxmin() 함수 만들기

- 실행 결과

```
maxmin ← {  
  ma ←  
  mi ←  
  return(  
}  
}
```

```
v1 ← c(7, 1, 2, 8, 9)
```

```
result ← maxmin(v1)
```

```
result$max
```

```
[1] 9
```

```
result$min
```

```
[1] 1
```

함수의 매개변수 1개 필요(벡터 하나)

변수의 최대값을 구함

변수의 최소값을 구함

변수의 최대값과 최소값을 각각 max, min
변수에 저장한 뒤 두 값을 모두 반환

result의 max값(최대값)

result의 min값(최소값)

예시

- 예시 5 : 벡터 인덱스 찾기
 - 실행 결과

```
weight ← c(69, 50, 55, 71, 89, 64, 59, 70, 71, 80)
```

```
[1] 5
```

```
# weight 벡터의 최댓값의 위치
```

```
[1] 2
```

```
# weight 벡터의 최솟값의 위치
```

```
[1] 1 6
```

```
# weight 벡터값 중 61에서 69인 데이터의 위치
```

```
w ← [1] 50 55 59
```

```
# weight 벡터값 중 60 이하인 값의 위치를 w에 저장
```

```
weight.2 ← [1] 50 55 59
```

```
# weight 벡터값 중 60 이하인 값을 weight.2에 저장
```

```
weight.2
```

```
[1] 50 55 59
```

예시

- 예시 6 : iris 데이터셋에서 Petal.Length가 가장 큰 관측값(행)의 내용 출력하기
 - 실행 결과

```
ma <-
```

```
# Petal.Length가 가장 큰 행의 위치를 ma에 저장
```

```
iris_ma <-
```

```
# 가장 큰 행의 내용을 iris_ma에 저장
```

```
iris_ma
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
119	7.7	2.6	6.9	2.3	virginica

감사합니다

kimtwan21@dongduk.ac.kr

김 태 완