My first R Markdown

nahyun

2020 4 11

####  
#### R의 기본연산  
####  
  
## 2-1 산술연산과 주석   
  
2+3

## [1] 5

(3+6)\*8

## [1] 72

2^3 # 2의 세제곱

## [1] 8

## 2-2   
  
7+4

## [1] 11

# 2^3   
  
## 2-3 산술연산 함수   
  
log(10)+5 # 로그함수

## [1] 7.302585

sqrt(25) # 제곱근

## [1] 5

max(5,3,2) # 가장 큰 값

## [1] 5

####  
#### 변수  
####  
  
  
## 2-4 변수   
  
a <- 10  
b <- 20   
c <- a+b  
print(c)

## [1] 30

## 2-5 변수의 값 저장과 확인   
  
a <- 125   
a

## [1] 125

print(a)

## [1] 125

## 2-6 변수의 자료형, 변수의 값 변경   
  
a <- 10 # a에 숫자 저장  
b <- 20  
a+b # a+b의 결과 출력

## [1] 30

a <- "A" # a에 문자 저장  
#a+b # a+b의 결과 출력. 에러 발생  
  
####  
#### 벡터의 이해  
####  
  
  
## 2-7 벡터 만들기   
  
x <- c(1,2,3) # 숫자형 벡터  
y <- c("a","b","c") # 문자형 벡터  
z <- c(TRUE,TRUE, FALSE, TRUE) # 논리형 벡터  
x # x에 저장된 값을 출력

## [1] 1 2 3

y

## [1] "a" "b" "c"

z

## [1] TRUE TRUE FALSE TRUE

## 2-8 벡터 만들기   
  
w <- c(1,2,3, "a","b","c")  
w

## [1] "1" "2" "3" "a" "b" "c"

## 2-9 연속적인 숫자로 이루어진 벡터의 생성   
  
v1 <- 50:90   
v1

## [1] 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74  
## [26] 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90

v2 <- c(1,2,5, 50:90)   
v2

## [1] 1 2 5 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71  
## [26] 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90

## 2-10 일정한 간격의 숫자로 이루어진 벡터 생성   
  
v3 <- seq(1,101,3)   
v3

## [1] 1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55  
## [20] 58 61 64 67 70 73 76 79 82 85 88 91 94 97 100

v4 <- seq(0.1,1.0,0.1)   
v4

## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

## 2-11 반복된 숫자로 이루어진 벡터 생성   
  
v5 <- rep(1,times=5) # 1을 5번 반복  
v5

## [1] 1 1 1 1 1

v6 <- rep(1:5,times=3) # 1에서 5까지 3번 반복  
v6

## [1] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5

v7 <- rep(c(1,5,9), times=3) # 1, 5, 9를 3번 반복  
v7

## [1] 1 5 9 1 5 9 1 5 9

## 2-12 벡터의 원소값에 이름 지정   
  
score <- c(90,85,70) # 성적  
score

## [1] 90 85 70

names(score) # score에 저장된 값들의 이름을 보이시오

## NULL

names(score) <- c("John","Tom","Jane") # 값들에 이름을 부여  
names(score) # score에 저장된 값들의 이름을 보이시오

## [1] "John" "Tom" "Jane"

score # 이름과 함께 값이 출력

## John Tom Jane   
## 90 85 70

# 2-13 벡터에서 원소값 추출   
  
d <- c(1,4,3,7,8)  
d[1]

## [1] 1

d[2]

## [1] 4

d[3]

## [1] 3

d[4]

## [1] 7

d[5]

## [1] 8

d[6]

## [1] NA

## 2-14 벡터에서 여러개의 값 한꺼번에 추출하기   
  
d <- c(1,4,3,7,8)  
d[c(1,3,5)] # 1, 3, 5번째 값 출력

## [1] 1 3 8

d[1:3] # 처음 세 개의 값 출력

## [1] 1 4 3

d[seq(1,5,2)] # 홀수 번째 값 출력

## [1] 1 3 8

d[-2] # 2번째 값 제외하고 출력

## [1] 1 3 7 8

d[-c(3:5)] # 3~5번째 값은 제외하고 출력

## [1] 1 4

## 2-15 벡터에서 이름으로 값 추출하기   
  
GNP <- c(2090,2450,960)   
GNP

## [1] 2090 2450 960

names(GNP) <- c("Korea","Japan","Nepal")  
GNP

## Korea Japan Nepal   
## 2090 2450 960

GNP[1]

## Korea   
## 2090

GNP["Korea"]

## Korea   
## 2090

GNP[c("Korea","Nepal")]

## Korea Nepal   
## 2090 960

## 2-16 벡터에 저장된 원소값 변경   
  
v1 <- c(1,5,7,8,9)  
v1

## [1] 1 5 7 8 9

v1[2] <- 3 # v1의 2번째 값을 3으로 변경  
v1

## [1] 1 3 7 8 9

v1[c(1,5)] <- c(10,20) # v1의 1, 5번째 값을 각각 10, 20으로 변경  
v1

## [1] 10 3 7 8 20

####  
####벡터의 연산   
####  
  
## 2-17 벡터와 숫자값 연산   
  
d <- c(1,4,3,7,8)  
2\*d

## [1] 2 8 6 14 16

d-5

## [1] -4 -1 -2 2 3

3\*d+4

## [1] 7 16 13 25 28

## 2-18 벡터와 벡터 간의 연산   
  
x <- c(1,2,3)  
y <- c(4,5,6)  
x+y # 대응하는 원소끼리 더하여 출력

## [1] 5 7 9

x\*y # 대응하는 원소끼리 곱하여 출력

## [1] 4 10 18

z <- x + y # x, y를 더하여 z에 저장  
z

## [1] 5 7 9

## 2-19 벡터에 적용가능한 함수   
  
d <- c(1,2,3,4,5,6,7,8,9,10)  
sum(d) # d에 포함된 값들의 합

## [1] 55

sum(2\*d) # d에 포함된 값들에 2를 곱한 후 합한 값

## [1] 110

length(d) # d에 포함된 값들의 개수

## [1] 10

mean(d[1:5]) # 1~5번째 값들의 평균

## [1] 3

max(d) # d에 포함된 값들의 최댓값

## [1] 10

min(d) # d에 포함된 값들의 최솟값

## [1] 1

sort(d) # 오름차순 정렬

## [1] 1 2 3 4 5 6 7 8 9 10

sort(d, decreasing = FALSE) # 오름차순 정렬

## [1] 1 2 3 4 5 6 7 8 9 10

sort(d, decreasing = TRUE) # 내림차순 정렬

## [1] 10 9 8 7 6 5 4 3 2 1

v1 <- median(d)  
v1

## [1] 5.5

v2 <- sum(d)/length(d)  
v2

## [1] 5.5

## 2-20 벡터에 논리연산자 적용   
  
d <- c(1,2,3,4,5,6,7,8,9)  
d>=5

## [1] FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

d[d>5] # 5보다 큰 값

## [1] 6 7 8 9

sum(d>5) # 5보다 큰 값의 개수를 출력

## [1] 4

sum(d[d>5]) # 5보다 큰 값의 합계를 출력

## [1] 30

d==5

## [1] FALSE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE

condi <- d > 5 & d < 8 # 조건을 변수에 저장  
d[condi] # 조건에 맞는 값들을 선택

## [1] 6 7

####   
#### 팩터   
####  
  
## 2-22 팩터  
  
bt <- c('A', 'B', 'B', 'O', 'AB', 'A') # 문자형 벡터 bt 정의  
bt.new <- factor(bt) # 팩터 bt.new 정의  
bt # 벡터 bt의 내용 출력

## [1] "A" "B" "B" "O" "AB" "A"

bt.new # 팩터 bt.new의 내용 출력

## [1] A B B O AB A   
## Levels: A AB B O

bt[5] # 벡터 bt의 5번째 값 출력

## [1] "AB"

bt.new[5] # 팩터 bt.new의 5번째 값 출력

## [1] AB  
## Levels: A AB B O

levels(bt.new) # 팩터에 저장된 값의 종류를 출력

## [1] "A" "AB" "B" "O"

as.integer(bt.new) # 팩터의 문자값을 숫자로 바꾸어 출력

## [1] 1 3 3 4 2 1

bt.new[7] <- 'B' # 팩터 bt.new의 7번째에 'B' 저장  
bt.new[8] <- 'C' # 팩터 bt.new의 8번째에 'C' 저장

## Warning in `[<-.factor`(`\*tmp\*`, 8, value = "C"): invalid factor level, NA  
## generated

bt.new # 팩터 bt.new의 내용 출력

## [1] A B B O AB A B <NA>  
## Levels: A AB B O

####  
#### 리스트  
####  
  
## 2-21 리스트   
  
ds <- c(90, 85, 70, 84)  
my.info <- list(name='Tom', age=60, status=TRUE, score=ds)  
my.info # 리스트에 저장된 내용을 모두 출력

## $name  
## [1] "Tom"  
##   
## $age  
## [1] 60  
##   
## $status  
## [1] TRUE  
##   
## $score  
## [1] 90 85 70 84

my.info[[1]] # 리스트의 첫 번째 값을 출력

## [1] "Tom"

my.info$name # 리스트에서 값의 이름이 name인 값을 출력

## [1] "Tom"

my.info[[4]] # 리스트의 네 번째 값을 출력

## [1] 90 85 70 84