pr2_201921487_김서준

김서준

2023년 09년 14일

1. R로 계산하기

1.1 기본연산

```
31 + 3 # 31 더하기 3
```

```
## [1] 34
```

```
15 - 3 + 7 # 순서대로 15 빼기 3 더하기 7
```

```
## [1] 19
```

```
13 * 2 - 6 / 2 # 계산순서는 사칙연산의 순서와 같다
```

```
## [1] 23
```

```
13 * (2 - 6) / 2 # 수학의 괄호와 같이 괄호를 사용하면 괄호 안 연산부터 계산한다.
```

```
## [1] -26
```

```
8 %/% 2 # 8 나누기 2의 몫
```

```
## [1] 4
```

```
11 %% 3 # 11 나누기 3의 나머지, 결과값은 2
```

```
## [1] 2
```

```
n <- 21 %% 4 # 21 나누기 4의 나머지 1을 n이라는 변수에 할당
print(n)
```

```
## [1] 1
```

1.2 수학함수 사용

```
log(2) #로그함수 log()
```

```
## [1] 0.6931472
```

```
log(exp(1)) #exp()는 지수함수 log()는 로그함수
```

```
## [1] 1
```

```
sqrt(4) #squre root 제곱근을 구하는 함수 sqrt()
```

```
## [1] 2
```

```
4^5 #4의 5승
```

```
## [1] 1024
```

```
4**5 #4의 5승
```

```
## [1] 1024
```

```
round(9.13) #반올림함수 round(), 9.13을 반올림 하면 결과값은 9
```

```
## [1] 9
```

```
ceiling(1.41) #올림함수 ceiling(), 1.41을 올림하면 결과값은 2
```

```
## [1] 2
```

```
floor(1.95) #내림함수 floor(),  1.95를 내림하면 결과값은 1
```

```
## [1] 1
```

```
pi #원주율을 호출해준다.
```

```
## [1] 3.141593
```

2. 수치 요약하기

2.1 벡터 생성 및 출력

- 정수형 값이 저장된 벡터를 생성하기
- 벡터 출력해보기

```
v1 <- 3 # v1변수에 3을 할당
v2 <- c(4,5) # v2라는 변수에 4와 5가 있는 벡터 할당
v3 <- 3:11 # v3 이라는 변수에 3에서 11까지가 있는 벡터 할당
v4 <- c(v1, v2, v3) # v1, v2, v3 각각의 변수들이 가진 값들을 모두 합친 새로운 벡터 할당
print(v1)
```

```
## [1] 3
```

```
print(v2)
```

```
## [1] 4 5
```

```
print(v3)
```

```
## [1]  3  4  5  6  7  8  9 10 11
```

```
print(v4)
```

```
##  [1]  3  4  5  3  4  5  6  7  8  9 10 11
```

```
#각각의 변수 출력
v1 * 2 #v1 변수에 저장된 값에 2 곱하기
```

```
## [1] 6
```

```
v1/v3 #v1 변수에 저장된 값을 2 v3 변수에 저장된 값으로 나누기
```

```
## [1] 1.0000000 0.7500000 0.6000000 0.5000000 0.4285714 0.3750000 0.3333333
## [8] 0.3000000 0.2727273
```

2.2 평균구하기

- 평균을 구하는 여러가지 방법

```
(1+2+3+4+5+6+7+8+9) / 9 #괄호를 통해 먼저 1부터 9까지 모두 더하고 그 값을 9로 나눔
```

```
## [1] 5
```

```
sum(1, 2, 3, 4, 5, 6, 7, 8, 9) / 9 #괄호안에 있는 값들을 더하는 함수 sum()
```

```
## [1] 5
```

```
v5 <- 1:9 #v5라는 변수에 1부터 9까지의 수들을 할당
sum(v5) / length(v5) # v5에 포함된 모든 수들의 합을 v5의 길이(변수 안에 있는 것들의 갯수)로 나누기
```

```
## [1] 5
```

```
mean(v5) # v5의 평균을 구하는 함수 mean()
```

```
## [1] 5
```

2.3 함수활용

```
mean(v5) # 평균
```

```
## [1] 5
```

```
var(v5) # 분산
```

```
## [1] 7.5
```

```
sd(v5) # 표준편차
```

```
## [1] 2.738613
```

```
median(v5) # 중앙값
```

```
## [1] 5
```

```
max(v5) # 최댓값
```

```
## [1] 9
```

```
min(v5) # 최솟값
```

```
## [1] 1
```

```
v6 <- 1:10 # v6라는 변수에 1부터 10까지의 수들을 할당
median(v6) #v6안에 있는 수들의 중앙값
```

```
## [1] 5.5
```

3. 문자값이 저장된 벡터 생성

```
myEmail <- "rlatjwns234@ajou.ac.kr"
birthday <- c("2000년", "3월", "20일")
birthday2 <- paste("2000년", "3월", "20일")
birthday3 <- paste0("2000년", "3월", "20일")

print(myEmail)
```

```
## [1] "rlatjwns234@ajou.ac.kr"
```

```
print(birthday)
```

```
## [1] "2000년" "3월"    "20일"
```

```
print(birthday2)
```

```
## [1] "2000년 3월 20일"
```

```
print(birthday3)
```

```
## [1] "2000년3월20일"
```

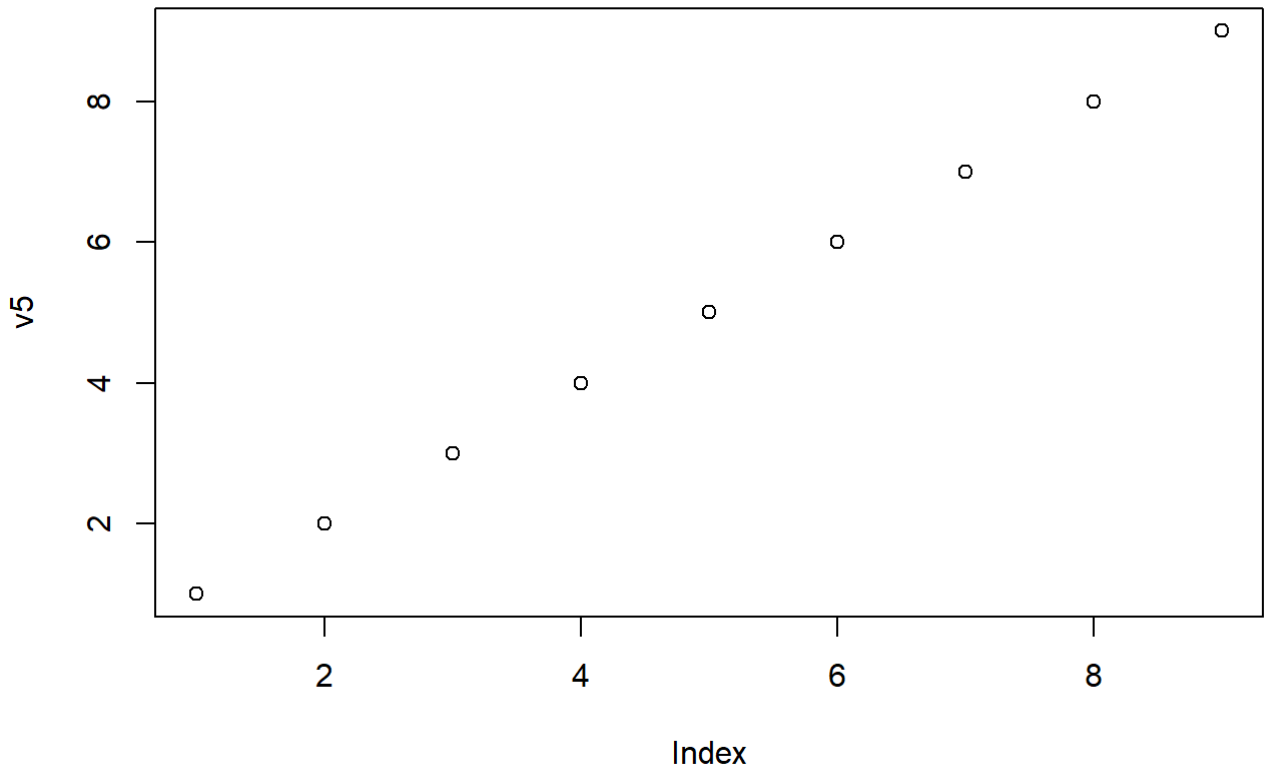
4. 기타 R 활용법

4.1 세미콜론 활용 및 변수명만으로 출력하기

```
mean(v5) ; sd(v5) ; plot(v5) # 한줄에 3개 명령문 실행
```

```
## [1] 5
```

```
## [1] 2.738613
```




```
myEmail ; birthday # print명령문 없이, 변수만 실행해도 출력됨
```

```
## [1] "rlatjwns234@ajou.ac.kr"
```

```
## [1] "2000년" "3월"    "20일"
```

4.2 작업폴더 확인 및 변경

- 변경할 폴더는 사전에 만들어져 있는 폴더여야함
- 본인이 작업할 폴더의 경로를 setwd(“”) 의 따옴표 사이에 입력
- 작업할 폴더는 본인이 원하는 경로로 지정하기
- 예) setwd(“c:/data”)

```
getwd() # 작업 폴더 확인
```

```
## [1] "C:/학업/학교 수업/23-2/R-programming/과제/PR2"
```

```
#setwd("C:W학업W학교 수업W23-2WR-programmingW과제WPR2") #작업 폴더 변경
getwd() # 변경된 폴더 위치 확인
```

```
## [1] "C:/학업/학교 수업/23-2/R-programming/과제/PR2"
```

PR2 연습문제

문제1. min-max normalization

```
student_num <- c(9, 3, 8, 9, 8, 6, 6, 13) #student_num이라는 변수에 값들을 벡터로 할당한다.
min <- min(student_num) #min이라는 변수에 student_num의 최솟값 할당
max <- max(student_num) #max라는 변수에 student_num의 최댓값 할당
min_max <- ((student_num - min)/(max-min)) #min-max normalization을 구하는 식으로 구한 값을 min
_max 라는 변수에 할당
print(min_max) #min_max 출력
```

```
## [1] 0.6 0.0 0.5 0.6 0.5 0.3 0.3 1.0
```

문제2. NDVI 구하기

```
NIR <- 11 # NIR 이라는 변수에 11 할당
RED <- 14 # RED 라는 변수에 14 할당
NDVI <- ((NIR-RED)/(NIR+RED)) # NDVI라는 변수에 NDVI 를 구하는 식을 통해 구한 값을 할당
print(NDVI) # NDVI 를 출력
```

```
## [1] -0.12
```

문제3. DoV와 DoD 계산하기

```
NN <- 10
TF <- 8
DF <- 3
tw <- 0.5
n <- 3
j <- 2
# 각각의 변수들에 주어진 값들을 할당
DoV <- (((TF/NN)*1) - (tw*(n-j)))
DoD <- (((DF/NN)*1) - (tw*(n-j)))
# DoV와 DoD라는 변수에 각각의 값을 계산하는 식을 통해 계산된 값을 할당
print(DoV)
```

```
## [1] 0.3
```

```
print(DoD)
```

```
## [1] -0.2
```

```
# 저장된 값들을 출력
```