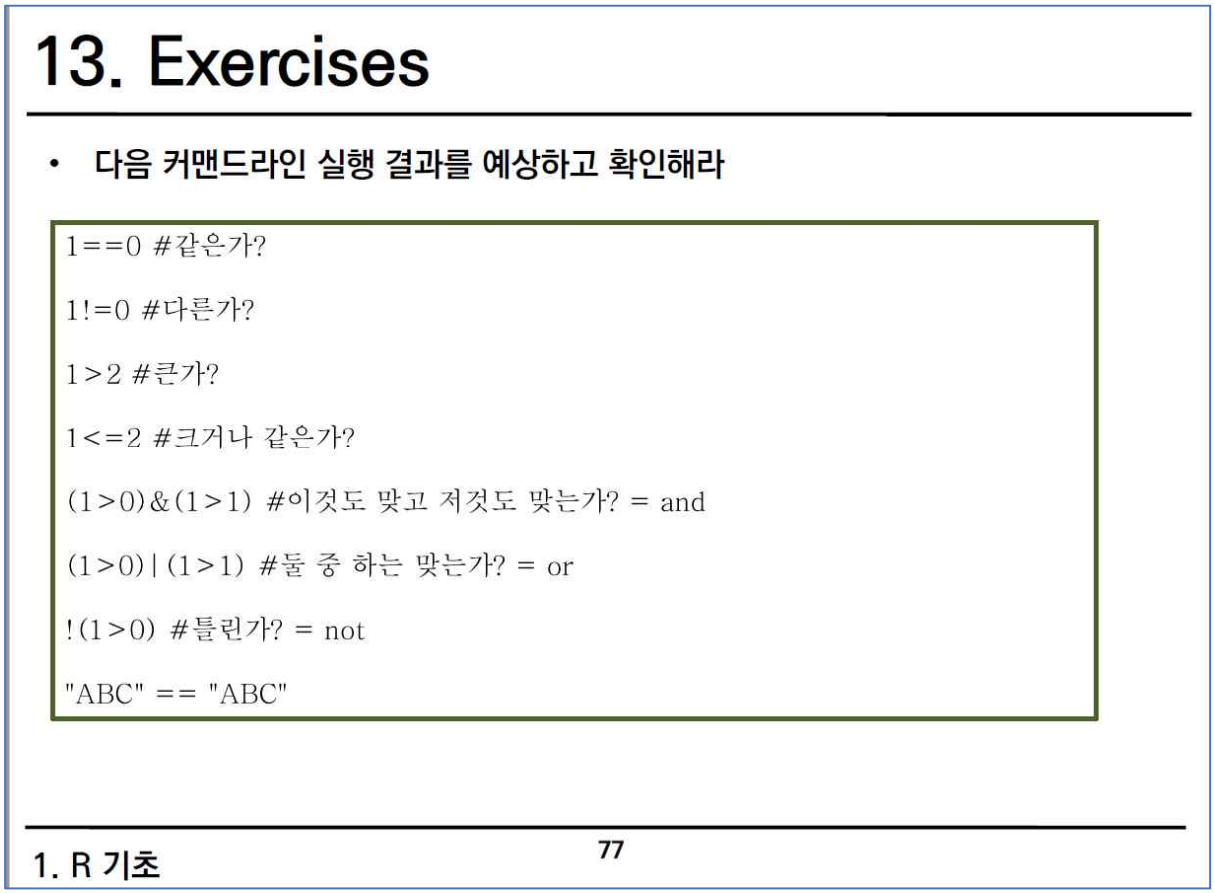
R기초_이근백교수님 1일차 연습문제 풀이

#Day1 연습문제 p77



13. Exercises

- 다음 커맨드라인 실행 결과를 예상하고 확인해라

```
1==0 #같은가?
1!=0 #다른가?
1>2 #큰가?
1<=2 #크거나 같은가?
(1>0)&(1>1) #이것도 맞고 저것도 맞는가? = and
(1>0)|(1>1) #둘 중 하는 맞는가? = or
!(1>0) #틀린가? = not
"ABC" == "ABC"
```

1. R 기초 77

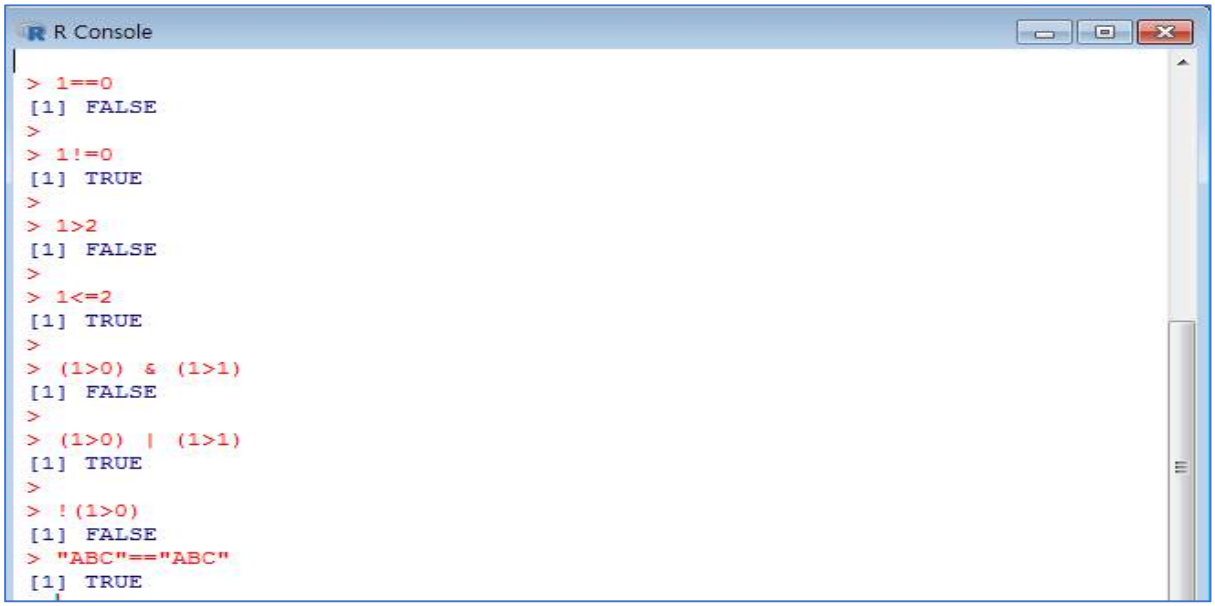
<코드>

```
1==0
1!=0
1>2
1<=2
(1>0) & (1>1)
(1>0) | (1>1)
!(1>0)
"ABC"=="ABC"
```

#answer: 위 코드를 R에 작성해보면 아래와 같이 결과를 확인 가능하다.

<해설>

아래 결과창을 통해서 위 문제에 대한 답을 알 수 있다.



<코드 실행결과>

#Day1 연습문제 p78

13. Exercises

- R 함수 'unique'는 무슨 일을 할까?
- R 함수 'seq'와 'rep'를 이용하여 다음의 벡터를 생성하고 싶다.

  369369369

  아래의 (a)와 (b) 중 적절한 것은?

  (a) rep(seq(3,9,by=3),3)
  (b) seq(3,9,by=3)[rep(1:3,3)]

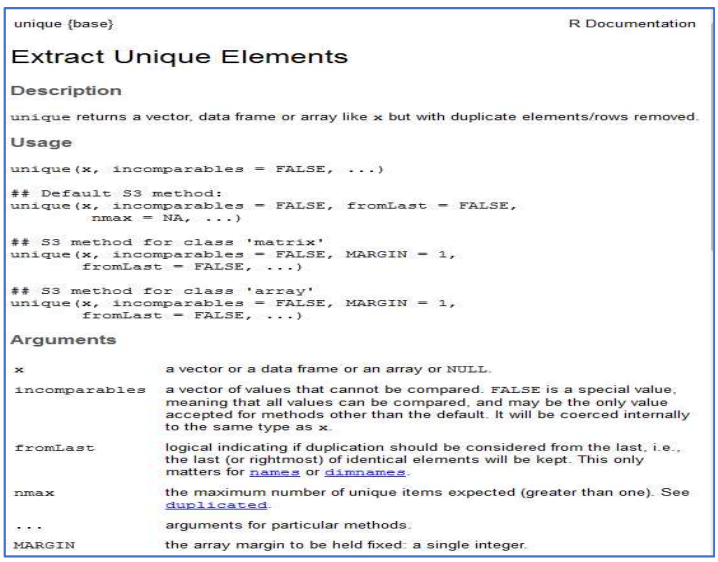
- 수치 벡터가 주어질 때 벡터의 최소값, 최대값을 주는 R 함수를 작성해보아라
- 수치 벡터가 주어질 때 벡터의 최빈값을 주는 R 함수를 작성해보아라

<코드>

```
1.
?unique

2.
rep(seq(3,9, by=3),3)
seq(3,9,by=3)[rep(1:3,3)]

3.
a <-sample(1:10,5, replace=TRUE)
a
max(a)
min(a)
```

<해설>

1. ?unique를 통해서 unique함수의 기능을 파악할 수 있으며, 아래와 같은 창을 발견할 수 있다. 이를 통해 unique함수는 중복값을 제거한 후 return해주는 함수임을 알 수 있다.



unique {base} R Documentation

Extract Unique Elements

Description

unique returns a vector, data frame or array like x but with duplicate elements/rows removed.

Usage

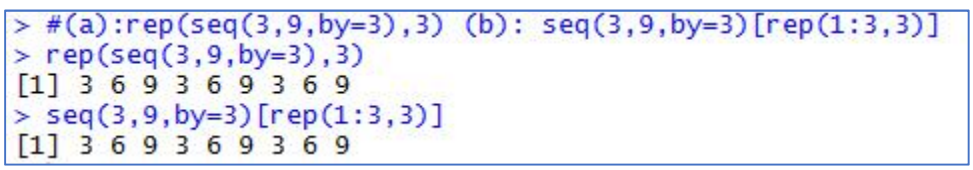
```
unique(x, incomparables = FALSE, ...)

## Default S3 method:
unique(x, incomparables = FALSE, fromLast = FALSE,
       nmax = NA, ...)

## S3 method for class 'matrix'
unique(x, incomparables = FALSE, MARGIN = 1,
       fromLast = FALSE, ...)

## S3 method for class 'array'
unique(x, incomparables = FALSE, MARGIN = 1,
       fromLast = FALSE, ...)
```

Arguments

x	a vector or a data frame or an array or NULL.
incomparables	a vector of values that cannot be compared. FALSE is a special value, meaning that all values can be compared, and may be the only value accepted for methods other than the default. It will be coerced internally to the same type as x.
fromLast	logical indicating if duplication should be considered from the last, i.e., the last (or rightmost) of identical elements will be kept. This only matters for names or dimnames.
nmax	the maximum number of unique items expected (greater than one). See duplicated.
...	arguments for particular methods.
MARGIN	the array margin to be held fixed: a single integer.

<?unique 결과창>

2. 코드를 작성함으로 똑 같은 결과를 확인할 수 있다. 즉 어느 방법을 사용해도 무방하다.



```
> #(a):rep(seq(3,9,by=3),3) (b): seq(3,9,by=3)[rep(1:3,3)]
> rep(seq(3,9,by=3),3)
[1] 3 6 9 3 6 9 3 6 9
> seq(3,9,by=3)[rep(1:3,3)]
[1] 3 6 9 3 6 9 3 6 9
```

<code 결과창>

3.

1) 먼저 sample함수를 통해서 a라는 오브젝트에 벡터를 생성한다. sample(1:10,5, replace=TRUE) 이 함수의 의미는 다음과 같다. 1~10 사이의 숫자들 중 5개를 추출할 것이며, 반복추출(replace)를 허용한다.

2) a라는 벡터를 다시 call함으로 a 벡터 속에 어떤 숫자가 들어갔는지 확인한다. 여기서는 5 1 4 4 2가 들어갔음을 확인 가능하다.

3) Max(a), min(a)를 통해서 최대값, 최소값이 return됨을 확인할 수 있다.

```
> #make any vector using sample function
> a <- sample(1:10,5, replace=TRUE)
> a
[1] 5 1 4 4 2
> #return max in vector (a)
> max(a)
[1] 5
> #return min in vector (a)
> min(a)
[1] 1
```

<위 코드의 결과창>

4.

<코드>

```
# random number generation
x <- sample(1:10, 30, T)
x

# table
table <- table(x)

# 가장 큰 빈도수 출력
order <- max(table)

# 최빈값이 여러개일 수 있으니, 다음과 같은 논리 vector를 만듦
v <- which(table(x)==order)

mode <- as.numeric(names(table)[v])
mode
```

<해설>

코드는 자신이 직접 작성하기 나름임을 다시 한 번 말씀드리며, 이 문제에서는 제가 짠 코드를 설명하겠습니다.

1) X 오브젝트에 앞선 문제에서 설명했던 것처럼 sample을 넣습니다. 이는 실제적으로 제가 짠 코드가 올바로 작동하는지 보기 위함입니다. 그리고 x를 다시 한번 타이핑하면서 x 속에 어떤 숫자들이 들어갔는지 확인합니다. 아래 결과창에서 결과를 확인할 수 있습니다.

2) Table(x)라는 코드를 통해서 x 오브젝트 속에 나열된 숫자들의 빈도표를 만들 수 있습니다. 이 결과를 table이라는 오브젝트에 저장합니다.

3) Max(table)을 통해서 가장 많은 빈도수를 order 오브젝트에 저장합니다.

4) Table(x)==order함수라는 논리를 통해서 true와 false를 구분합니다. 즉, v<-which(table(x)==order)의 의미는 각 숫자의 빈도수가 3)에서 저장한 order와 동일한 숫자를 v라는 오브젝트에 저장하라는 의미입니다.

5) mode <- as.numeric(names(table)[v])를 통해서 최빈값을 mode 오브젝트에 저장합니다.

   names(table)은 table의 각 빈도에 해당되는 이름을 말합니다. 아래 결과창을 보면 저희의 경우는 1 2 3 4 5 6 7 8 9 10이 이에 해당합니다. 그리고 names(table)[v]를 통해서 v에 해당되는 names를 return시킵니다. 마지막으로 as.numeric()함수를 통해서 이 결과값을 숫자로 바꾸어줍니다.

위 코드를 통해서 아래와 같은 결과가 확인 가능하다.

```
> x <- sample(1:10, 30, T)
> x
 [1]  5  1  5  3 10  8  2  5  7  5 10  8  4  2  3  9  5  5  7  8  2  9 10  6  6  2  1  9  2  1
> table <- table(x)
> order <- max(table)
> v <- which(table(x)==order)
> mode <- as.numeric(names(table)[v])
> table
x
 1  2  3  4  5  6  7  8  9 10
 3  5  2  1  6  2  2  3  3  3
> mode
[1] 5
```

위 코드가 잘 이해가 되지 않으시면 구글에서 최빈값을 산출하는 R function을 무수히 찾을 수 있습니다.