

Foundations of R Software

Lecture 29

Sorting, Ordering and Mode

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Sorting

sort function sorts the values of a vector in ascending order (by default) or descending order.

Syntax

```
sort(x, decreasing = FALSE, ...)
```

```
sort(x, decreasing = FALSE, na.last = NA, ...)
```

x Vector of values to be sorted

decreasing Should the sort be increasing or decreasing

na.last for controlling the treatment of **NA**s.
If **TRUE**, missing values in the data are put last;
if **FALSE**, they are put first;
if **NA**, they are removed.

Sorting

Example

```
> y = c(8,5,7,6)
```

```
> y
```

```
[1] 8 5 7 6
```

```
> sort(y)
```

```
[1] 5 6 7 8
```

```
> sort(y, decreasing = TRUE)
```

```
[1] 8 7 6 5
```

R Console

```
> y = c(8,5,7,6)
```

```
> y
```

```
[1] 8 5 7 6
```

```
> sort(y)
```

```
[1] 5 6 7 8
```

```
>
```

```
> sort(y, decreasing = TRUE)
```

```
[1] 8 7 6 5
```

```
> |
```

Ordering

order function sorts a variable according to the order of variable.

Syntax

```
order(x, decreasing = FALSE, ..., )
```

```
order(x, decreasing = FALSE, na.last = TRUE, ...)
```

x Vector of values to be sorted

decreasing Should the sort be increasing or decreasing

na.last for controlling the treatment of **NAs**.
If **TRUE**, missing values in the data are put last;
if **FALSE**, they are put first;
if **NA**, they are removed.

Ordering

Example

```
> y = c(9, 8, 5, 7, 6)
```

```
> y
```

```
[1] 9 8 5 7 6
```

Value	9	8	5	7	6
Position	1	2	3	4	5

Ordered value	5	6	7	8	9
Position	3	5	4	2	1

Ordering

Example

```
> y
```

```
[1] 9 8 5 7 6
```

Value	9	8	5	7	6
Position	1	2	3	4	5

Ordered value	5	6	7	8	9
Position	3	5	4	2	1

```
> order(y)
```

```
[1] 3 5 4 2 1
```

```
> order(y, decreasing = TRUE)
```

```
[1] 1 2 4 5 3
```

Ordering

```
R Console  
  
> y = c(9,8,5,7,6)  
> y  
[1] 9 8 5 7 6  
> order(y)  
[1] 3 5 4 2 1  
>  
> order(y, decreasing = TRUE)  
[1] 1 2 4 5 3  
> |
```

Mode

Every object has a mode.

The mode indicates how the object is stored in memory: as a

- ✓ **number,**
- ✓ **character string,**
- ✓ **list of pointers to other objects,**
- ✓ **function etc.**

Mode

mode function gives us such information.

Syntax

mode()

Mode

Object	Example	Mode
Number	<code>1.234</code>	numeric
Vector of numbers	<code>c(6, 7, 8, 9)</code>	numeric
Character string	<code>"India"</code>	character
Vector of character strings	<code>c("India", "CANADA")</code>	character
Factor	<code>factor(c("MP", "UP"))</code>	numeric
List	<code>list("India", "CANADA")</code>	list
Data frame	<code>data.frame(x=1:2, y=c("India", "CANADA"))</code>	list
Function	<code>print</code>	function

Mode

Example

```
> mode(2.432)
```

```
[1] "numeric"
```

```
> mode(c(3,4,5,6,7,8))
```

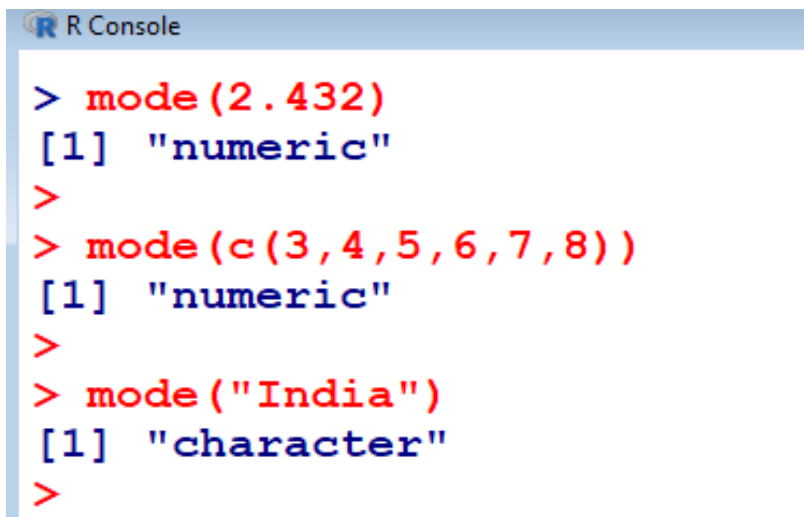
```
[1] "numeric"
```

```
> mode("India")
```

```
[1] "character"
```

```
> mode(c("India", "CANADA"))
```

```
[1] "character"
```

A screenshot of the R Console window. The title bar says "R Console". The console shows the following commands and their outputs:

```
> mode(2.432)
[1] "numeric"
>
> mode(c(3,4,5,6,7,8))
[1] "numeric"
>
> mode("India")
[1] "character"
>
```

Mode:

Example

```
> mode(factor(c("UP", "MP"))) )
```

```
[1] "numeric"
```

```
> mode(list("India", "USA"))
```

```
[1] "list"
```

```
> mode(data.frame(x=1:2, y=c("India", "USA")))
```

```
[1] "list"
```

```
> mode(print)
```

```
[1] "function"
```

Mode

R Console

```
> mode(factor(c("UP", "MP")))
[1] "numeric"
> mode(list("India", "USA"))
[1] "list"
> mode(data.frame(x=1:2, y=c("India", "USA")))
[1] "list"
> mode(print)
[1] "function"
```