

Introduction to R Software

Basics of Calculations

:::

Missing Data and Logical Operators

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Missing data

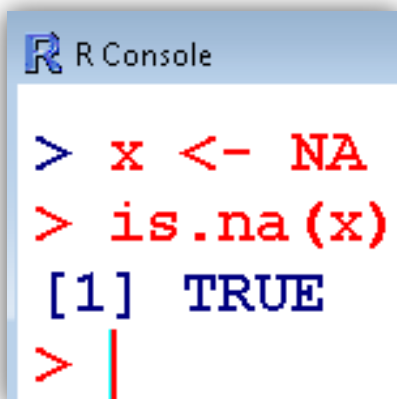
R represents missing observations through the data value **NA**

We can detect missing values using **is.na**

```
> x <- NA      # assign NA to variable x
```

```
> is.na(x)     # is it missing?
```

```
[1] TRUE
```

A screenshot of an R Console window. The title bar is light blue with the R logo and the text 'R Console'. The console area is white and contains the following text: a red prompt character followed by 'x <- NA', a red prompt character followed by 'is.na(x)', a blue output '[1] TRUE', and a red prompt character followed by a vertical bar '|'.

```
R Console  
> x <- NA  
> is.na(x)  
[1] TRUE  
> |
```

Missing data

Now try a vector to know if any value is missing?

```
> x <- c(11, NA, 13)

> is.na(x)

[1] FALSE TRUE FALSE
```

R Console

```
> x <- c(11,NA,13)
> is.na(x)
[1] FALSE TRUE FALSE
> |
```

Example : How to work with missing data

```
> x <- c(11,NA,13) # vector
```

```
> mean(x)
[1] NA
```

$$\frac{11+NA+13}{2}$$

```
> mean(x, na.rm = TRUE) # NAs can be removed
[1] 12
```

$$\frac{11+13}{2}=12$$

The null object, called **NULL**, is returned by some functions and expressions.

Note that **NA** and **NULL** are not the same.

NA is a placeholder for something that exists but is missing.

NULL stands for something that never existed at all.

Logical Operators and Comparisons

The following table shows the operations and functions for logical comparisons (True or False).

TRUE and **FALSE** are reserved words denoting logical constants.

Operator	Executions
>	Greater than
>=	Greater than or equal
<	Less than
<=	Less than or equal
==	Exactly equal to
!=	Not equal to
!	Negation (not)

Logical Operators and Comparisons

Operator	Executions
<code>& , &&</code>	and
<code> , </code>	or

- The shorter form performs element-wise comparisons in almost the same way as arithmetic operators.
- The longer form evaluates left to right examining only the first element of each vector. Evaluation proceeds only until the result is determined.
- The longer form is appropriate for programming control-flow and typically preferred in if clauses (conditional).

Logical Operators and Comparisons

TRUE and FALSE are reserved words denoting logical constants

Operator	Executions
<code>xor()</code>	either... or (exclusive)
<code>isTRUE(x)</code>	test if <code>x</code> is TRUE
<code>TRUE</code>	true
<code>FALSE</code>	false

Examples:

```
> 8 > 7  
[1] TRUE
```

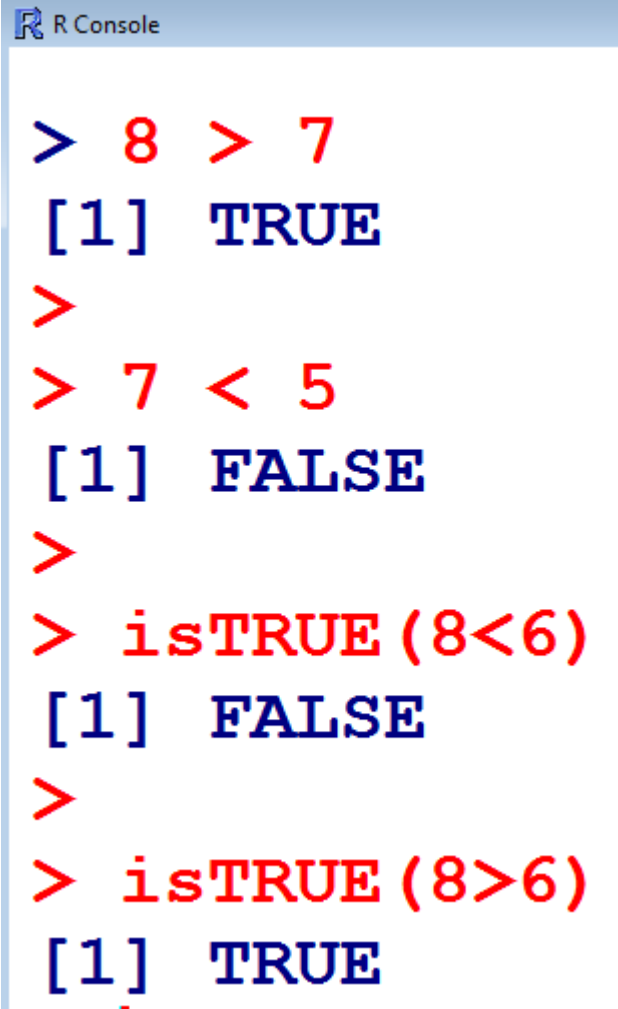
```
> 7 < 5  
[1] FALSE
```

Is 8 less than 6?

```
> isTRUE(8<6)  
[1] FALSE
```

Is 8 greater than 6?

```
> isTRUE(8>6)  
[1] TRUE
```




R Console

```
> 8 > 7  
[1] TRUE  
>  
> 7 < 5  
[1] FALSE  
>  
> isTRUE(8<6)  
[1] FALSE  
>  
> isTRUE(8>6)  
[1] TRUE
```


Examples:

```
> x <- 5  
> (x < 10) && (x > 2)      # && means AND  
[1] TRUE
```

 R Console

```
> x <- 5  
> (x < 10) && (x > 2)  
[1] TRUE
```

Examples:


```
> x <- 5
```

Is **x** less than 10 or **x** is greater than 5 ?

```
> (x < 10) || (x > 5)      # || means OR  
[1] TRUE
```

Is **x** greater than 10 or **x** is greater than 5 ?

```
> (x > 10) || (x > 5)  
[1] FALSE
```

 RGui (64-bit)

```
>  
> (x < 10) || (x > 5)  
[1] TRUE  
>  
> (x > 10) || (x > 5)  
[1] FALSE  
>
```

Examples:

```
> x = 10
```

```
> y = 20
```

Is **x** equal to 10 and is **y** equal to 20?

```
> (x == 10) & (y == 20)
```

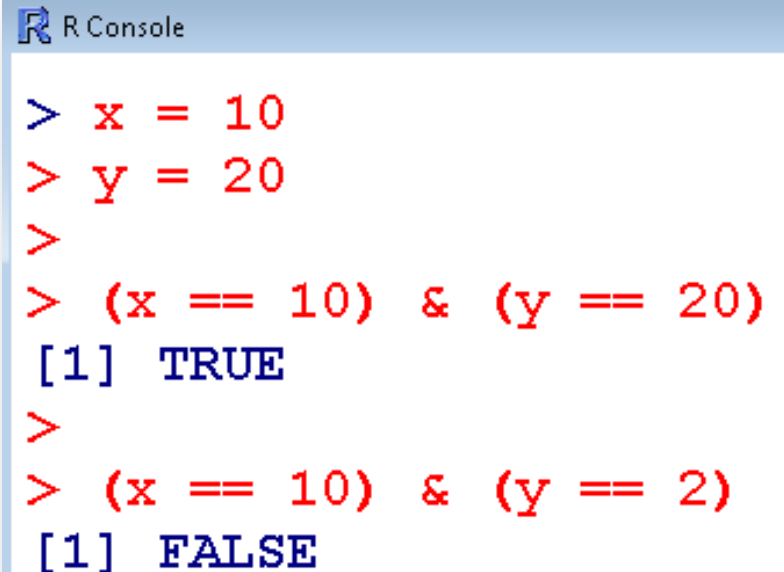
```
[1] TRUE
```

== means exactly
equal to

Is **x** equal to 10 and is **y** equal to 2?

```
> (x == 10) & (y == 2)
```

```
[1] FALSE
```

A screenshot of an R console window with a blue header bar containing the R logo and the text "R Console". The console shows a series of commands and their outputs. The commands are: x = 10, y = 20, a blank line, (x == 10) & (y == 20), [1] TRUE, a blank line, (x == 10) & (y == 2), and [1] FALSE. The commands are in red text, and the outputs are in blue text.

```
R Console
> x = 10
> y = 20
>
> (x == 10) & (y == 20)
[1] TRUE
>
> (x == 10) & (y == 2)
[1] FALSE
```

Examples:

```
> x = 10
```

```
> y = 20
```

Is **x** equal to 1 and is **y** equal to 20?

```
> (x == 1) & (y == 20)
```

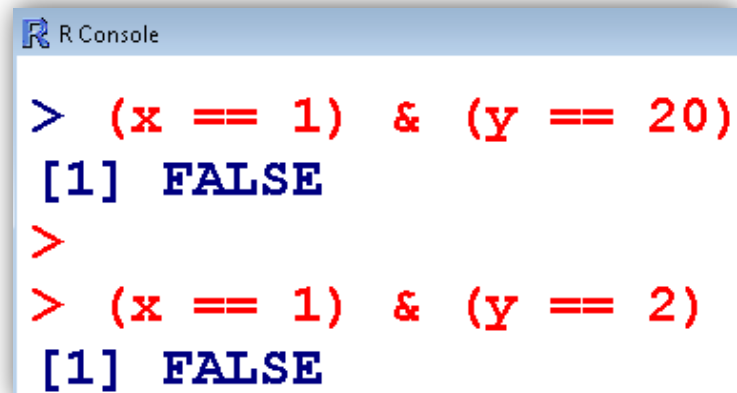
== means exactly
equal to

```
[1] FALSE
```

Is **x** equal to 1 and is **y** equal to 2?

```
> (x == 1) & (y == 2)
```

```
[1] FALSE
```

A screenshot of an R console window with a blue header bar containing the R logo and the text "R Console". The console shows two lines of code and their outputs. The first line is "> (x == 1) & (y == 20)" in red, followed by the output "[1] FALSE" in blue. The second line is "> (x == 1) & (y == 2)" in red, followed by the output "[1] FALSE" in blue.

```
R Console
> (x == 1) & (y == 20)
[1] FALSE
>
> (x == 1) & (y == 2)
[1] FALSE
```