R Lesson 3

R Operators

- Arithmetic and Assignment Operators

Operator	**Description**
+	addition
-	subtraction
*****	multiplication
/	division
^ or **	exponentiation
x %% y	modulus (x mod y) 5%%2 is 1 → gives us remainder.
x %/% y	integer division 5%/%2 is 2
x<-y or y->x	assignment; x gets y
:	create series (1:10)

R Operators

- Logical Operators

Operator	**Description**
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	exactly equal to
!=	Boolean not equal to
!x	Not x
x | y	x OR y (for vectors - x , y are logical tests)
||	OR for scalars (use with IF: IF x or y)
x & y	x AND y (for vectors)
&&	AND for scalars (use with IF: IF x and y)
isTRUE(x)	test if X is TRUE

Command Syntax

- functionname(x,arg=0)
- x – positional argument
  - usually required
  - must be in expected location (order)
- arg – keyword argument
  - often optional
  - usually has a default value
- Multiple parameters separated by commas

Command Syntax

- How do you specify an argument with multiple values that are separated by a comma like plot(x,y) with multiple points?
  - c(1,2)
  - Known as the combine function
  - plot(c(1,3,5,7),c(2,4,6,8))
  - Works with common data types
  - Blt <- c(‘bacon’, ‘lettuce’, ‘tomato’)

Some Commonly Used R Functions

- length()
- sum(), cumsum(), prod(), cumprod()
- mean(), sd(), var(), median(), min(), max(), range(), summary()
- exp(), log(), sin(), cos(), tan() [radians, not degrees]
- round(), ceiling(), floor(), signif()
- sort(), order(), rank(), rev()
- which(), which.max(), which.min()
- any(), all()
- apply(), tapply(), lapply()

Working with Vectors & Data Structures

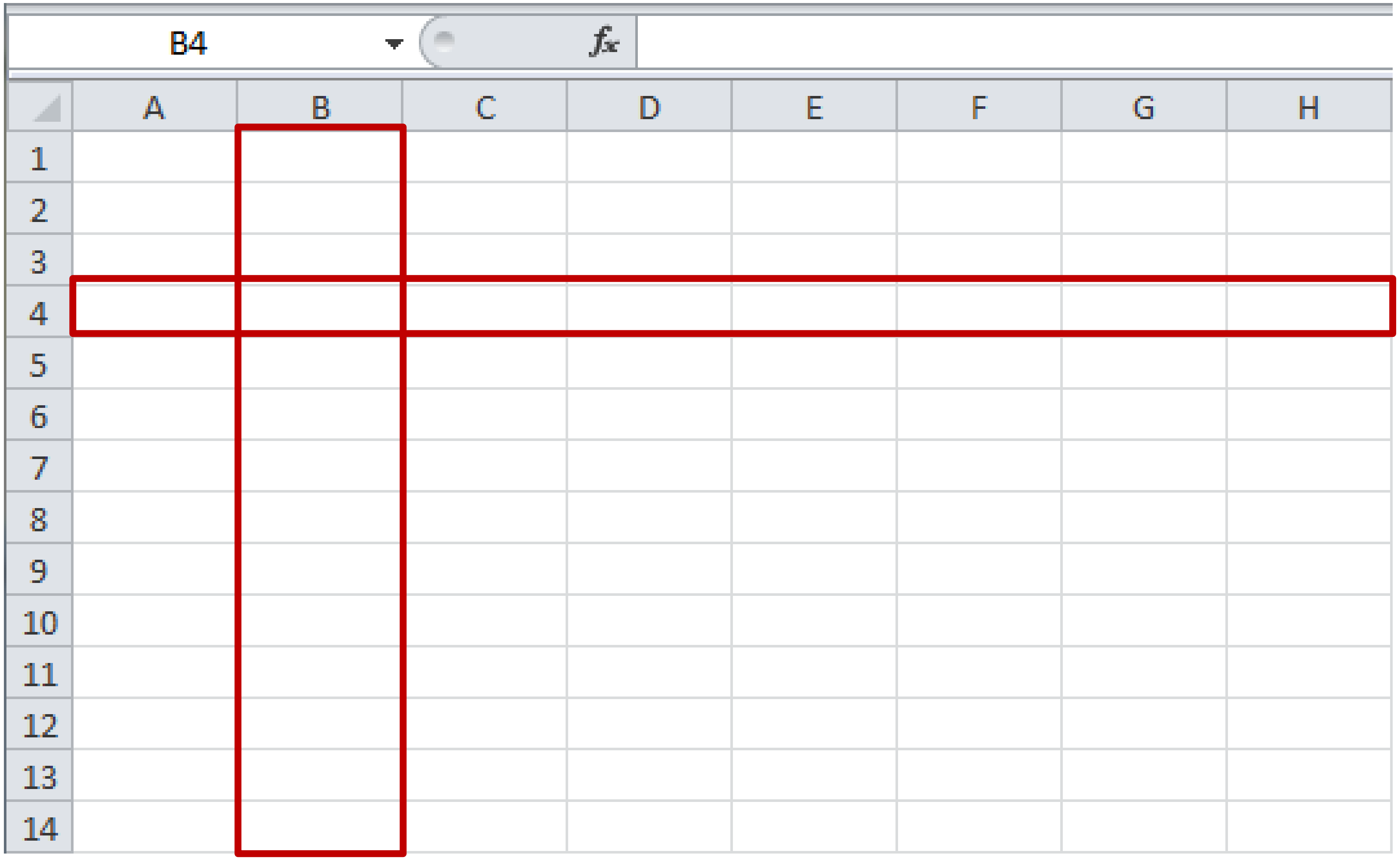
Command Syntax

- The devil is in the details!
- This is different from a function!!!
  - dataobject[indices]
  - dataobject – name of data frame, vector, etc.
  - indices – vector, formula, or function to specify members to use
    - Numbering starts at 1
    - Negative indices remove the specified members

[ ] - can extract more than 1 element of an object

[[ ]] - returns only 1 element of an object.

Working With Vectors



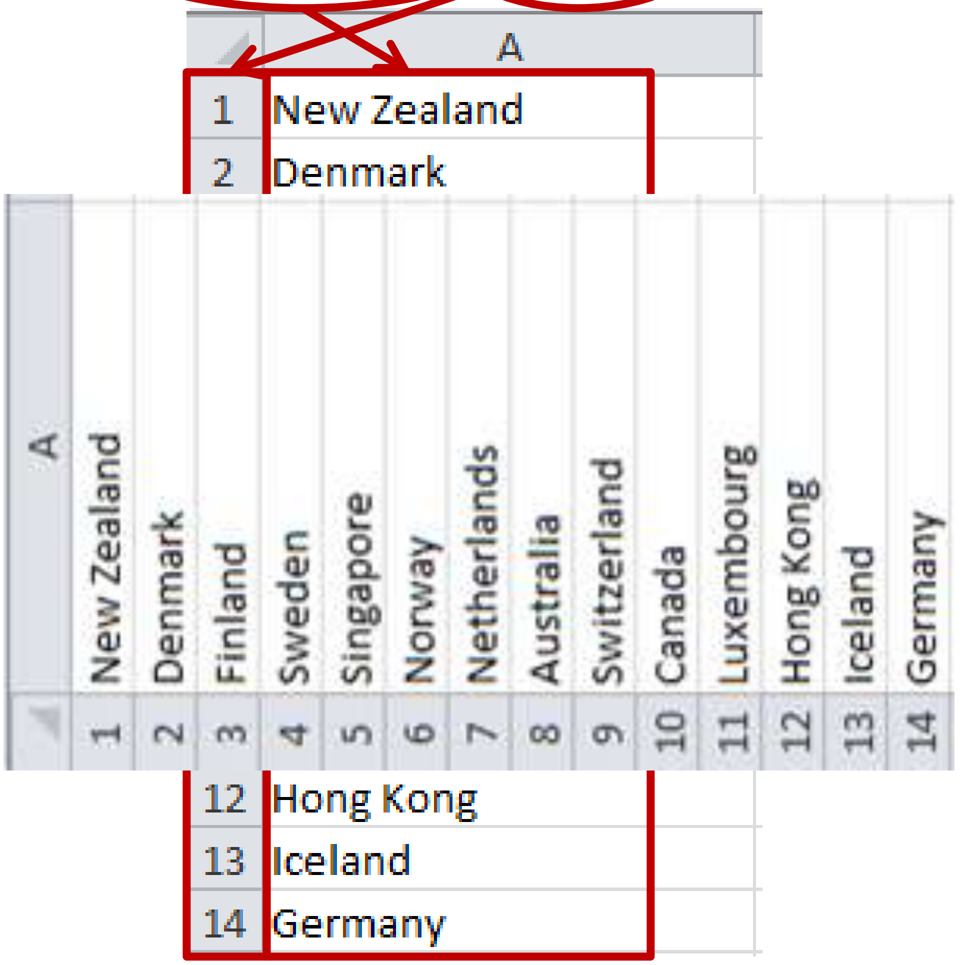
Working With Vectors

- A vector is a series of values
- Single dimension
- Not necessarily part of a data frame or matrix
- Frequently are a subset of data frame or matrix
- (V <- 1:14) # () send results to console
- [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14

Accessing Data in a Vector

Working With Vectors

countries[1:14]

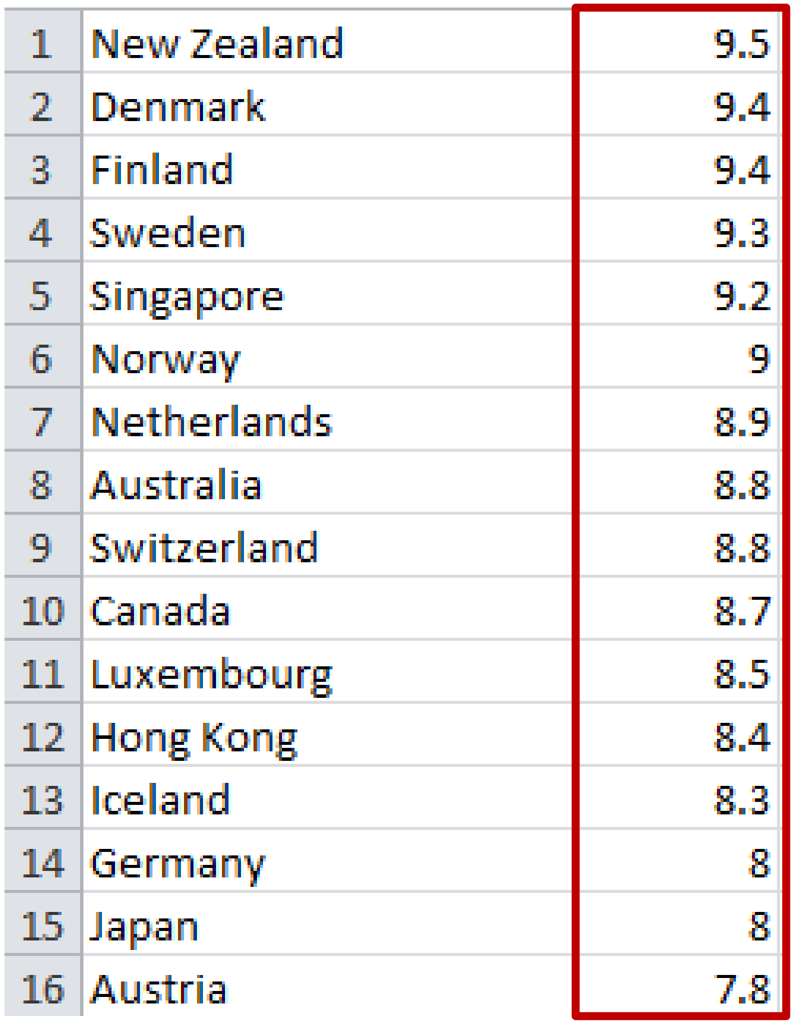


	A
1	New Zealand
2	Denmark
3	Finland
4	Sweden
5	Singapore
6	Norway
7	Netherlands
8	Australia
9	Switzerland
10	Canada
11	Luxembourg
12	Hong Kong
13	Iceland
14	Germany

Working With Missing Data

- NA
- <NA> (among characters without quotes)
- Function arguments
  - na.rm=TRUE instructs function to remove missing
  - na.strings= specify values to identify as missing in raw data
- Functions
  - na.omit() removes cases from action inside ()
  - is.na() tests to see if a value is missing
- NaN "Not a number" i.e.: Square root -4

Working With Vectors



1	New Zealand	9.5
2	Denmark	9.4
3	Finland	9.4
4	Sweden	9.3
5	Singapore	9.2
6	Norway	9
7	Netherlands	8.9
8	Australia	8.8
9	Switzerland	8.8
10	Canada	8.7
11	Luxembourg	8.5
12	Hong Kong	8.4
13	Iceland	8.3
14	Germany	8
15	Japan	8
16	Austria	7.8

Combining Vectors into one Object

- cbind(*V1, V2*) – as matrix columns
- rbind(*V1, V2*) – as matrix rows
- Matrix
  - A vector of equal length vectors
  - All values must be of same type

Combining Vectors into one Object

- data.frame(*V1, V2)* – into a data frame
- Data Frame
  - Matrix like structure
  - Ideal for mixed data types
- Recycling occurs when vectors of unequal length are combined

STOP 9/6/21

Using the matrix Function

- Creates a matrix from a single vector
- General form

  matrix(*data*, *nrow*=n, *ncol*=n, *byrow*=FALSE)

  – *data:* a data vector to be converted

  – *nrow:* specify desired number of rows

  – *ncol:* specify desired number of columns

  – *byrow:* if FALSE matrix filled by columns, otherwise by rows

Using the matrix Function

- Example:

```
mat1 <- matrix(1:12, nrow=4, byrow=TRUE)
```