The R Basics

Your Turn

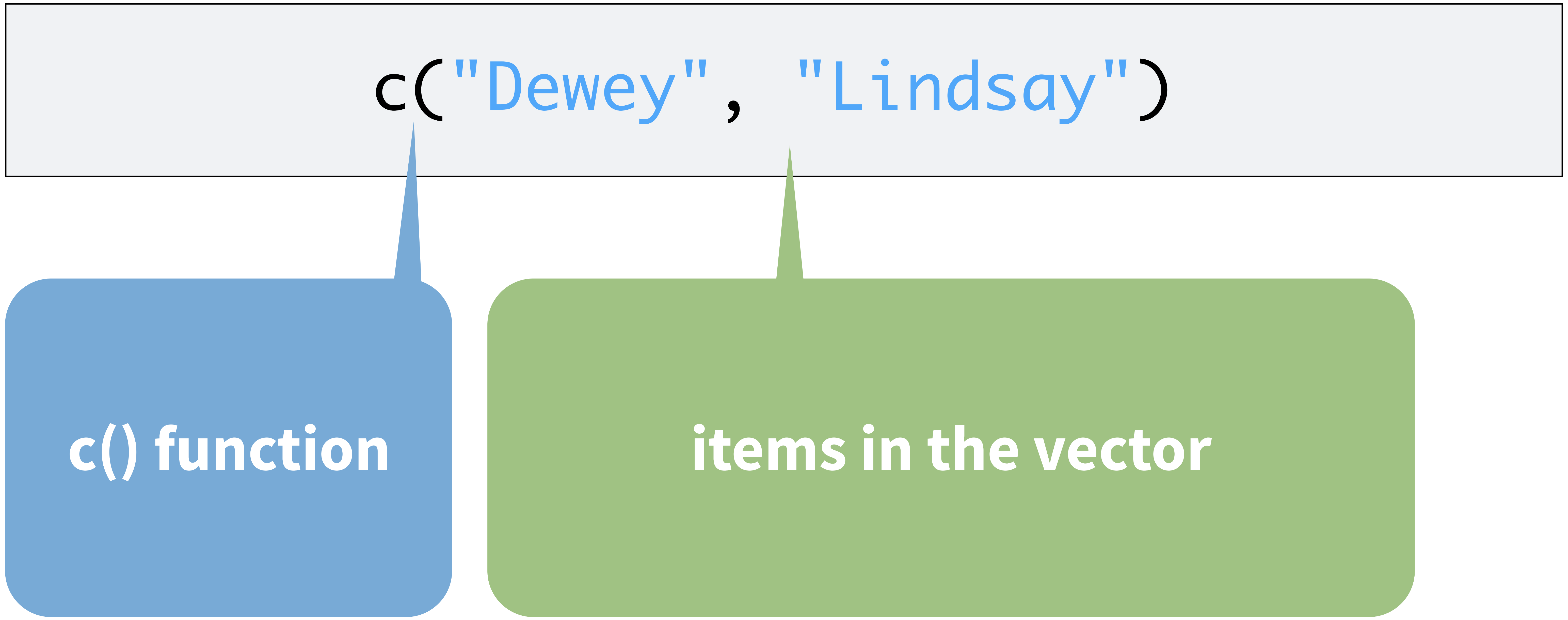
Open **02-Basic-R.Rmd**.

01:00

Vectors

Vectors

c() - create **vectors**



```
c("Dewey", "Lindsay")
```

c() function

items in the vector

Vectors

c() - create **vectors**

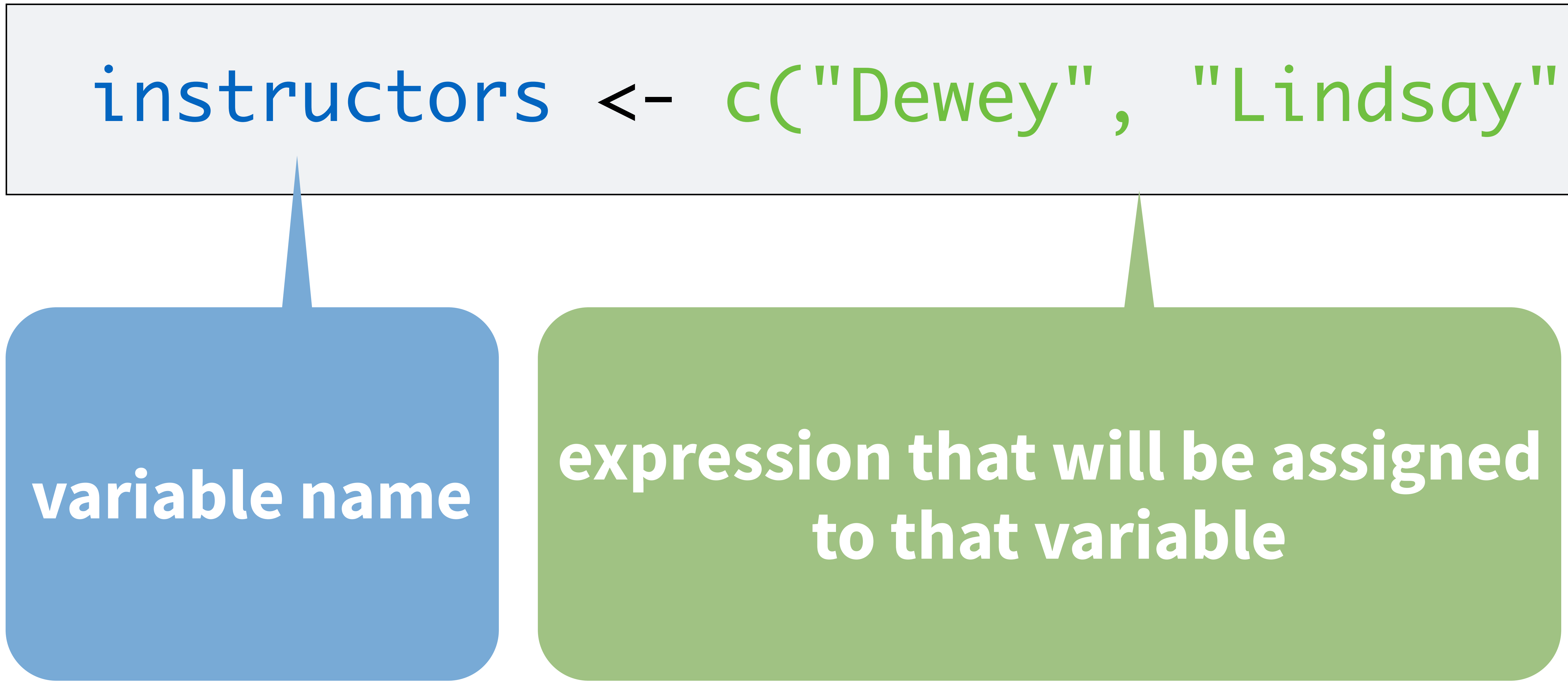
```
c("Dewey", "Lindsay")
```

Pro tip: quotes surround text!

Variables

Variables

<- - assign **variables**



```r
instructors <- c("Dewey", "Lindsay")
```

variable name

expression that will be assigned to that variable

Your Turn 1

Create a character vector of 3 fruit names and assign it to the variable `fruits`.

Your Turn 1

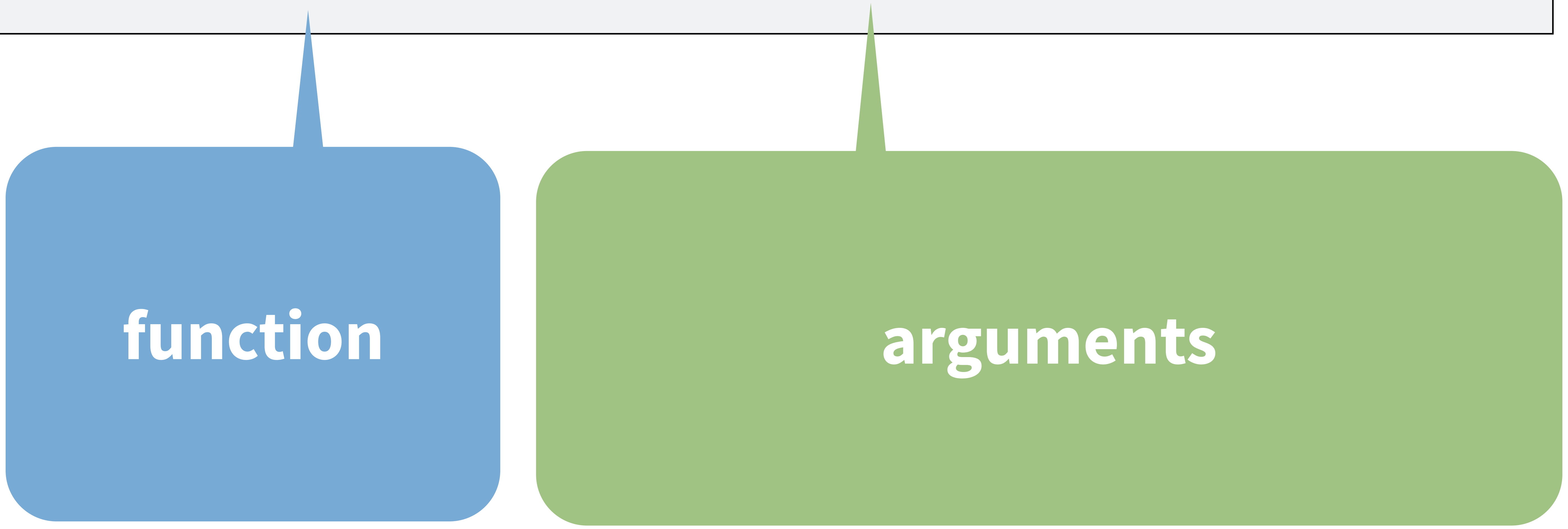
```
fruits <- c("apple", "orange" "banana")
fruits
```

```
[1] "apple"  "orange" "banana"
```

Functions

Functions

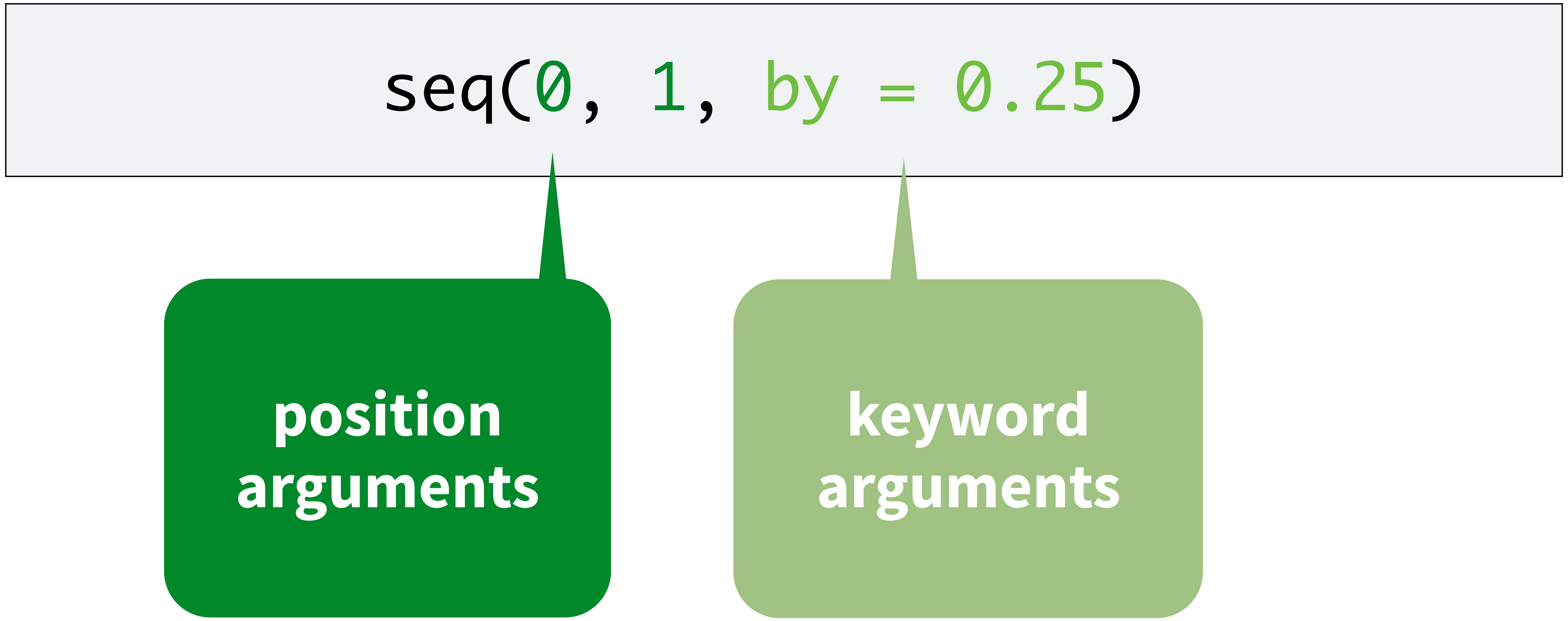
```
seq(0, 1, by = 0.25)
```



function

arguments

Functions

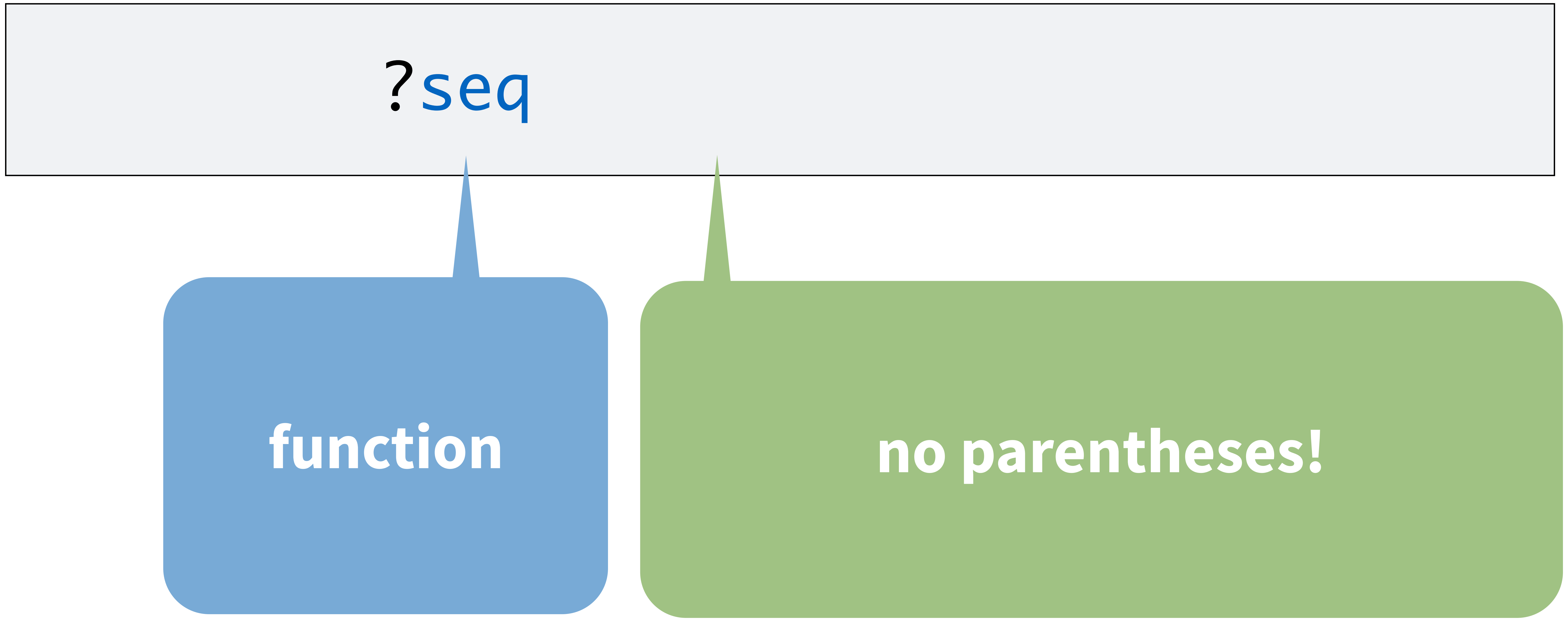


```
seq(0, 1, by = 0.25)
```

position arguments

keyword arguments

Function Help

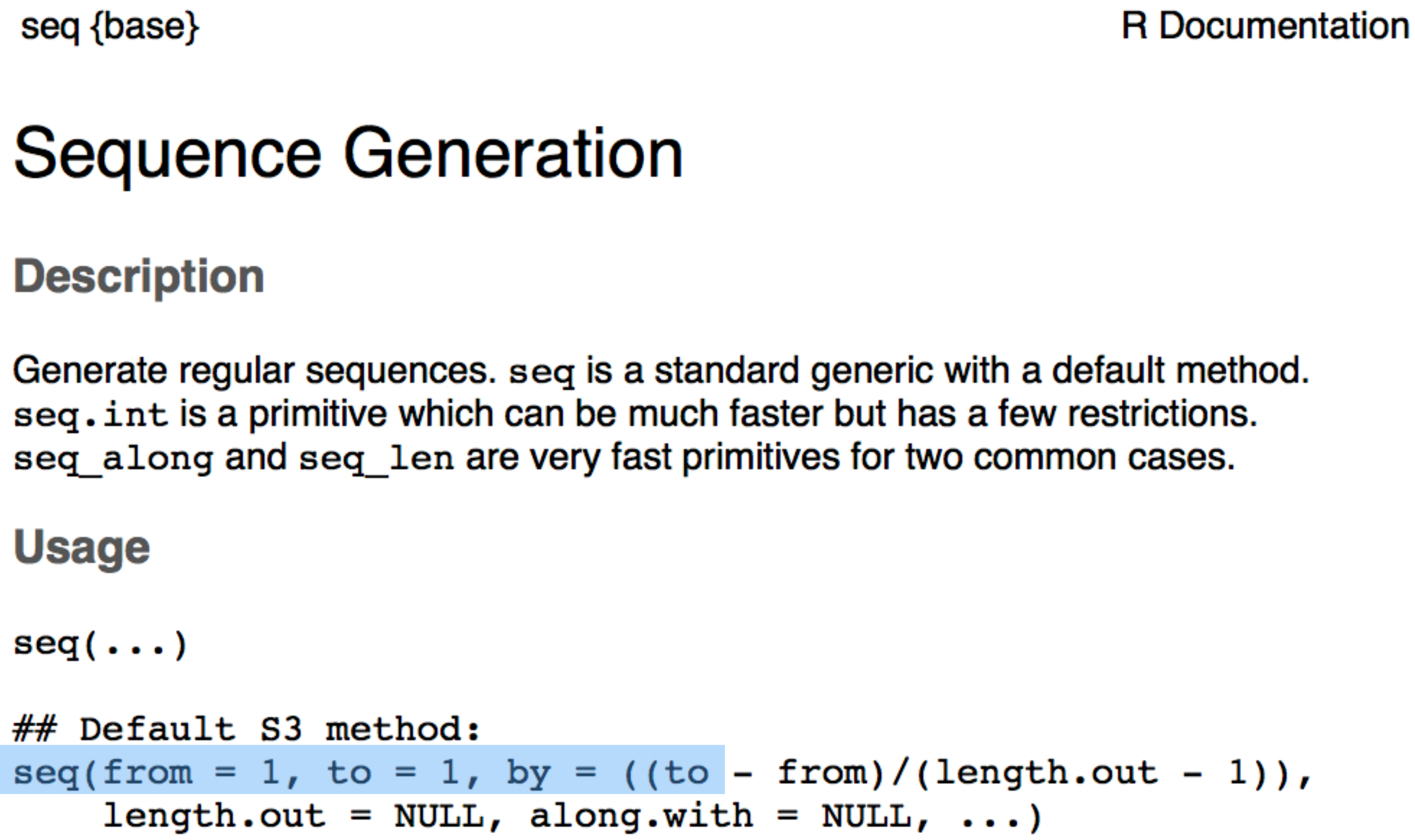


```
?seq
```

function

no parentheses!

Function Help



seq {base}　　　　R Documentation

Sequence Generation

Description

Generate regular sequences. `seq` is a standard generic with a default method. `seq.int` is a primitive which can be much faster but has a few restrictions. `seq_along` and `seq_len` are very fast primitives for two common cases.

Usage

```
seq(...)

## Default S3 method:
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)),
    length.out = NULL, along.with = NULL, ...)
```

Your Turn 2

Bring up the help for the `seq()` function. Then, use it to generate a numeric vector from 1 to 8 by 0.5 (1, 1.5, ..., 7.5, 8). Assign it to a variable called `numbers`.

02:00

Your Turn 2

```r
numbers <- seq(1, 8, by = 0.5)
numbers
```

```
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 [...] 7.5 8.0
```

Vectorized Functions

Your Turn 3

How many elements are in the vectors produced by:

- `max(numbers)`
- `mean(numbers)`
- `numbers * 3`
- `paste("Fruit:", fruits)`
- `numbers == 4`
- `fruits %in% c("banana", "pear")`

02:00

Your Turn 3

How many elements are in the vectors produced by:

- `max(numbers) [1] 8`
- `mean(numbers) [1] 4.5`
- `numbers * 3 [1]  3.0  4.5  6.0  7.5  9.0 10.5...`
- `paste("Fruit:", fruits) [1] "Fruit: apple"...`
- `numbers == 4 [1] FALSE FALSE FALSE FALSE...`
- `fruits %in% c("banana", "pear") [1] FALSE FALSE...`

Your Turn 3

How many elements are in the vectors produced by:

- `max(numbers)` **Length 1**
- `mean(numbers)` **Length 1**
- `numbers * 3` **Length of `numbers`**
- `paste("Fruit:", fruits)` **Length of `fruits`**
- `numbers == 4` **Length of `numbers`**
- `fruits %in% c("banana", "pear")` **Length of `fruits`**

Missing Values (NA)

Missing Values

NA - represents a **missing value**

```
instructors <- c("Dewey", NA)
```

NA represents a missing/ unknown/blank value

Missing Values

missing values propagate

```
sum(c(1, 2, NA))
```

```
[1] NA
```

Missing Values

missing values propagate by default

```r
sum(c(1, 2, NA), na.rm = TRUE)
```

```
[1] 3
```

Your Turn 4

The following code evaluates to NA. Fix it to get the mean of all the non-missing arguments.

```
mean(c(1, 2, 3, NA))
```

```
  [1] NA
```

01:00

Your Turn 4

The following code evaluates to NA. Fix it to get the mean of all the non-missing arguments.

```
mean(c(1, 2, 3, NA), na.rm = TRUE)
```

```
  [1] 2
```

Importing Data

Importing from Excel

`read_excel()` - from the *readxl* package

```
library(readxl)
```

First you have to load the package

Importing from Excel

`read_excel()` - from the *readxl* package

```
read_excel("warwick.xlsx")
```

The filename of the excel file

Importing from Excel

`read_excel()` - from the *readxl* package

```
read_excel("warwick.xlsx", sheet = "Sheet1")
```

The filename of the excel file

The sheet you meant (the first sheet is the default)

Importing from Excel

`read_excel()` - from the *readxl* package

```
read_excel("warwick.xlsx", sheet = "Sheet1", range = "A1:G7")
```

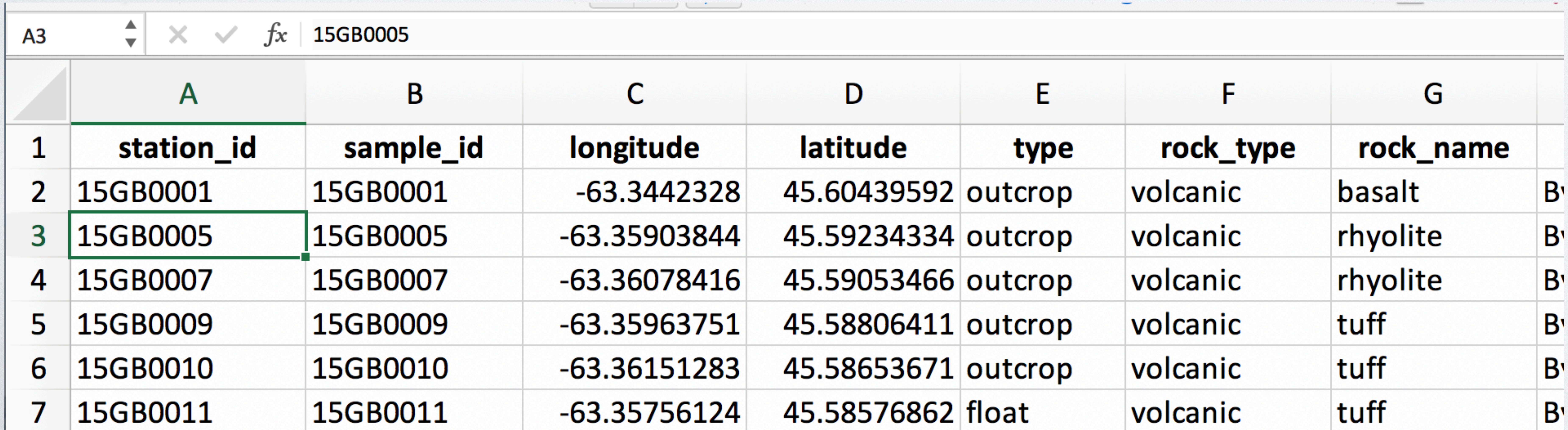
The filename of the excel file

It is often helpful to specify a specific range of cells

Your Turn 5

Modify the `range` argument in this call to `read_excel()` to read different regions of the sheet.

```
read_excel("warwick.xlsx", sheet = "Sheet1", range = ???)
```



A3 fx 15GB0005

	A	B	C	D	E	F	G	
1	**station_id**	**sample_id**	**longitude**	**latitude**	**type**	**rock_type**	**rock_name**	
2	15GB0001	15GB0001	-63.3442328	45.60439592	outcrop	volcanic	basalt	B
3	15GB0005	15GB0005	-63.35903844	45.59234334	outcrop	volcanic	rhyolite	B
4	15GB0007	15GB0007	-63.36078416	45.59053466	outcrop	volcanic	rhyolite	B
5	15GB0009	15GB0009	-63.35963751	45.58806411	outcrop	volcanic	tuff	B
6	15GB0010	15GB0010	-63.36151283	45.58653671	outcrop	volcanic	tuff	B
7	15GB0011	15GB0011	-63.35756124	45.58576862	float	volcanic	tuff	B

Your Turn 5

```
read_excel("warwick.xlsx", sheet = "Sheet1", range = "A1:G7")
```

station_id	sample_id	longitude	latitude	type	rock_type	rock_name
15GB0001	15GB0001	-63.3442328	45.60439592	outcrop	volcanic	basalt
15GB0005	15GB0005	-63.35903844	45.59234334	outcrop	volcanic	rhyolite
15GB0007	15GB0007	-63.36078416	45.59053466	outcrop	volcanic	rhyolite
15GB0009	15GB0009	-63.35963751	45.58806411	outcrop	volcanic	tuff
15GB0010	15GB0010	-63.36151283	45.58653671	outcrop	volcanic	tuff
15GB0011	15GB0011	-63.35756124	45.58576862	float	volcanic	tuff

Other formats

read_csv() for CSV files

```
read_csv("warwick.csv")
```

(see R for Data Science for more!)

The R Basics