2. Vector

A *vector* (or one-dimensional *array*) `v` is a collection of *values* (or *elements*) of the same type, each identified by an *index* in the range `1` to `length(v)`. *Combine* values into a vector with `c(...)`. e.g.

```
v <- c(2.71, 5, 3.14)
length(v)
v
```

index i	value v[i]
1	2.71
	5
3	

```
words <- c("tree", "ant", "chainsaw")
length(words)
words
```

index i	value words[i]
1	
2	"ant"
	"chainsaw"

Basic Vector Types, and Specifying Constants of These Types

- `numeric` (real number): digits with optional decimal point, with optional suffix of `E` or `e` for *exponent* digits (scientific notation); e.g. `3.14e2` is _____
- `character` (which should have been called *character string*): a *string* (or word) in double or single quotes, `"..."` or `'...'`. (*Escape sequences* include `\"` (double quote), `\'` (single quote), `\n` (newline), `\t` (tab), and `\\` (backslash).)

  `paste(..., sep = " ")` makes a string from its arguments, separated by `sep`. e.g.

  ```
  oak <- 70
  text = paste(sep="", "Tree names include \"oak.\"\nOak weighs ", oak, " lbs/ft^3.\n")
  ```

  `cat(..., sep = " ")` pastes and writes to console, interpreting escape sequences. e.g.

  ```
  cat(text)
  cat(sep = "", "oak=", oak, "\n") # display variable with helpful label
  ```

- `logical`: `TRUE` and `FALSE` (which become `1` and `0` when used in arithmetic)

  `any(v)` is `TRUE` if any of the values in `v` is `TRUE`; `all(v)` is `TRUE` if all are

  e.g. `v > 3`, `words == "ant"`, `sum(v > 3)`, `sum(words == "ant")`

`vector(mode="logical", length=0)` creates a vector of the given `mode` and `length`.

To change a vector's type, use `as.numeric()`, `as.character()`, or `as.logical()`. (There are three other basic types we will not use much: `integer`, `complex`, and `raw`.)

Names attribute

`names(x)` gets or sets a vector of `character` (strings) corresponding to values in x. e.g.

```
names(v) = c("e", "five", "pi"); v  # set names
names(v) = NULL; v                  # remove names
```

Names can also be set with `c()` by specifying "name=value" pairs. e.g. `y = c(burger=2.50, fries=1.50); y`

A Few Functions

e.g. `x <- c(12, 11, 16, 11)`

`sum(x)`, `max(x)`, `mean(x)`, `median(x)`, `sd(x)`

Operators (which act element-wise on vectors)

- arithmetic: `+ - * / ^` (and, for integer division, `%/%` is quotient, `%%` is remainder)

  e.g. The sample standard deviation of $x_1, x_2, \cdots, x_n$ is $s_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$:

  ```
  n = length(x)
  sqrt(sum((x - mean(x))^2) / (n-1))
  ```

- relation: `> >= < <= == !=` (last two are *equals* and *is not equal to*)

- logic:

! ("not")	T	F
	F	T

& ("and")	T	F
T	T	F
F	F	F

| ("or")	T	F
T	T	T
F	T	F

  e.g. `!(v > 3)`, `v < 4`, `(v > 3) & (v < 4)`, `(v > 3) | (v < 4)`

- assignment: `<-` (or `=`, which is not `==`)
- sequence: `:` (colon); e.g. `11:14` is `c(  ,  ,  ,  )`

  `seq()` is a related function:
  - `seq(from=1, to=1, by)`, e.g. `seq(10, 15, by=2)` is `c(  ,  ,  )`
  - `seq(from=1, to=1, length.out)`, e.g. `seq(10, 15, length.out=3)` is `c(  ,  ,  )`
- matching: `%in%`, e.g. `1:3 %in% c(2, 7)` is `c(  ,  ,  )`

Indexing

- For a vector `v` of positive `integer`, `x[v]` is those elements of `x` with indices in `v`; e.g. for `x <- 11:20` and `v <- c(1, 2, 10)`, `x[v]` is `c(  ,  ,  )`; `x[3]` is `c(  )` (or _____)
- For a vector `v` of negative `integer`, `x[v]` is those elements of `x` *excluding* those with indices in `v`; e.g. for `x <- 11:20` and `v <- c(-1, -2, -10)`, `x[v]` is _____
- For a vector `v` of `logical`,
  - `which(v)` is a vector of *indices* for which `v[i]` is `TRUE`; e.g.

    ```
    indices = which(x < 14)  # c(  ,  ,  )
    ```

    Now use the indices: `x[indices]` is `c(  ,  ,  )`
  - `x[v]` is those elements of `x` corresponding to `TRUE` values in `v`. e.g. `x[x < 14]` is `c(  ,  ,  )` (so "`which`" could have been omitted in previous example)

  e.g. `x[(x %% 2) == 0]`
- For a vector `v` of `character` names, `x[v]` is those elements of `x` whose names are in `v`; e.g.

  `x <- 1:3; names(x) <- c("one", "two", "Fred"); v <- c("Fred", "one"); x[v]` is _____