Introduction to R, Part II

Rebecca C. Steorts, STA 360

Agenda

- Data Structures
- Vectors

Data Structure

A **data structure** is group related data values that are stored in one object.

An example of a **data structure** is a **vector**, which is a sequence of values (all of the same type).

Vectors

```
x <- c(7, 8, 10, 45)
```

`c()` function returns a vector containing all its arguments in order

`x[1]` is the first element, `x[3]` is the 3rd element `x[-3]` is a vector containing all but the 3rd element

Exercise: What does x[-c(2:3)] return?

`vector(length=6)` returns an empty vector of length 6; helpful for filling things up later

Exercise on Vectors

```
weekly.hours <- vector(length=5)
weekly.hours[5] <- 8
weekly.hours
```

```
## [1] 0 0 0 0 8
```

Operators apply to vectors "pairwise" or "elementwise":

```
y <- c(-7, -8, -10, -45)
x+y
```

```
## [1] 0 0 0 0
```

```
x*y
```

```
## [1]   -49   -64  -100 -2025
```

Can also do pairwise comparisons:

```
x > 9
```

```
## [1] FALSE FALSE  TRUE  TRUE
```

Note: returns Boolean vector

Boolean operators work elementwise:

```
(x > 9) & (x < 20)
```

```
## [1] FALSE FALSE  TRUE FALSE
```

Functions on vectors

Many functions take vectors as arguments:

- `mean()`, `median()`, `sd()`, `var()`, `max()`, `min()`, `length()`, `sum()`: return single numbers
- `sort()` returns a new vector
- `hist()` takes a vector of numbers and produces a histogram, a highly structured object, with the side-effect of making a plot
- Similarly `ecdf()` produces a cumulative-density-function object
- `summary()` gives a five-number summary of numerical vectors
- `any()` and `all()` are useful on Boolean vectors

Addressing vectors

Vector of indices:

```
x[c(2,4)]
```

```
## [1]  8 45
```

Vector of negative indices

```
x[c(-1,-3)]
```

```
## [1]  8 45
```

(why that, and not `7 10`?)

Boolean vector:

```
x[x>9]
```

```
## [1] 10 45
```

```
y[x>9]
```

```
## [1] -10 -45
```

`which()` turns a Boolean vector in vector of TRUE indices:

```
x
```

```
## [1]  7  8 10 45
```

```
y
```

```
## [1]  -7  -8 -10 -45
```

```
places <- which(x > 9)
places
```

```
## [1] 3 4
```

```
y[places]
```

```
## [1] -10 -45
```

Named components

You can give names to elements or components of vectors

```
names(x) <- c("v1","v2","v3","fred")
names(x)
```

```
## [1] "v1"   "v2"   "v3"   "fred"
```

```
x[c("fred","v1")]
```

```
## fred   v1
##   45    7
```

note the labels is what R prints; not actually part of the value

`names(x)` is just another vector (of characters):

```
names(y) <- names(x)
sort(names(x))
```

```
## [1] "fred" "v1"   "v2"   "v3"
```

```
which(names(x)=="fred")
```

```
## [1] 4
```