Indexing

Part 1

Last time was objects

- **Objects** are the **nouns** of programming languages
- **Objects** store something
  - value
  - character string
  - **vector** of values or characters
  - entire data frame
  - model
  - etc...

So what?

- Making an object is only useful if you can use it later on
- We need to be able to access our objects
- And we need to be able to access bits and pieces of our objects
  - Example: what if you only want one or a few of the objects in a **vector**?

Indexing

Indexing is how we access items within a vector.

To index a vector, we're going to use square brackets `[ ]`

An example

Let's make a **vector** of *some* my favorite TV `comedies`:

```
comedies <- c("Parks and Rec", "Broad City", "I'm Sorry", "Avenue 5",
              "Bored to Death", "The Office", "Silicon Valley", "30 Rock
              "Big Mouth", "Futurama")
```

If you want to look at *all* the objects contained in the vector, all you need is your vector name

```
comedies
```

```
##  [1] "Parks and Rec"  "Broad City"     "I'm Sorry"      "Avenue 5"
##  [5] "Bored to Death" "The Office"     "Silicon Valley" "30 Rock"
##  [9] "Big Mouth"      "Futurama"
```

But, if I only want to know the 6th TV show in this `comedies` vector I would type:

```
comedies[6]
```

```
## [1] "The Office"
```

Indexing

An **index** is the *position* of an element in a vector

In R, indexing starts at 1. That is, the first element of a vector is located at position 1.

The basic syntax is to take the name of the vector and directly next to in, in square brackets, put the position you want to get.

- Something like `objectName[position number]`

Indexing Vectors

```
comedies <- c("Parks and Rec", "Broad City", "I'm Sorry", "Avenue 5",
              "Bored to Death", "The Office", "Silicon Valley", "30 Rock
              "Big Mouth", "Futurama")
```

To select objects that are sequential (in a row):

- `comedies[3:5]`
- You can think of `:` as *"through"*
  - `[3:5]` = "three *through* five"

```
comedies[3:5]
```

```
## [1] "I'm Sorry"      "Avenue 5"       "Bored to Death"
```

Indexing Vectores

To select objects that are *not* in a row:

```
comedies[c(1,7,10)]
```

```
## [1] "Parks and Rec"  "Silicon Valley" "Futurama"
```

Note that you need to wrap the position numbers inside of `c()`. What you're doing is making a mini-vector of position numbers! Vectors on vectors!

Indexing Vectors

Since you can make these mini-vectors, you can go crazy and combine sequential and non-sequential objects.

- Example:

```
comedies[c(1:4, 9)]
```

```
## [1] "Parks and Rec" "Broad City"    "I'm Sorry"     "Avenue 5"
## [5] "Big Mouth"
```

This reads as *"give me the elements that correspond to positions 1,2,3,4 (1 through 4), and 9"*

A more serious example

Let's say we have a vector that contains body mass indices (BMIs) for 20 participants in your study. A healthy BMI is between 18.5-24.9, and anything above 30 is considered "obese". You inspect your vector called `bmi`.

```
##  [1] 34.2 31.6 29.8 27.2 26.5 20.0 25.1 33.7 21.1 22.3 32.0 24.9 33.0 28.6
## [16] 55.8 55.8 60.9 81.0 65.2
```

The last 5 numbers are WAY off the scale! You recently implemented a new survey tool to collect the data, and only those 5 participants used the new method. You realize there's probably something wrong with the data collection!

You'll need those **index** positions in order to remove those 5 participants.

```
bmi[16:20]
```

```
## [1] 55.8 55.8 60.9 81.0 65.2
```

(We will talk more about removing participants in the coming weeks!)