# Functions II

Nathaniel MacNell

EPID 799B

# Overview

- Loops
- Conditional execution (if)

- Tour: function design (let’s write regression!)

- Activity: Solve a problem with a function

# Loops - Overview

- Same as loops from SAS and other languages
  - The DATA step is also essentially a loop
- R has better alternatives (but good to fall back on)
  - Avoid loops with apply() and vectorized operations

# Loops

- Used to do things repeatedly (same as SAS)
- Work by stepping through any vector rather than ***a*** *to* ***b*** *by* ***c*** (accomplish this with `seq(a,b,c)` )
- Loop "body" is enclosed in {}
  - Just like a function!

```
for(i in vector) {
     do something
}
```

# A Simple Loop

```
for(i in 1:10) {
     cat(
          paste(“Iteration”,i, “\n”)
     )
}
```

What does this loop do?

(Hint: cat() just prints something to the terminal)

# Loop examples – DATA Step Style!

```
x <- c(8,6,7,5,3,0,9) # some data

for(i in length(x) ) {
     x[i] <- x[i] + 1 # add one to each
}

# this is equivalent to (vectorized):
x = x + 1
```

# While loops (bad)

```
# repeat until style
x <- 0
while(x<0.5) {
     x <- x + runif(1) # R UNIFform
}
```

Please avoid **while** loops unless you are writing:

1. A simulation, or generating a custom distribution
2. A regression solver (or other recursive program)

# Repeat loops (worse)

```
# repeat forever style
repeat {
     x <- rnorm(n=1,mean=0,sd=1)
     if(x>2) break
}
```

Please avoid **repeat** loops unless you are writing:

1. A simulation, or generating a custom distribution
2. A regression solver (or other recursive program)

# Conditional Execution

- Useful for functions (so they can take multiple paths depending on the input).

```
if(condition) {
     do something
}
```

# Elseif and else: handle other cases

```
if(option=1) {          # the first thing to
     plot(1)             # try to match to
}

elseif(option=2) {       # not matching if, but
     plot(2)             # matching this one
}                         # (can have multiple)

else{                    # anything not already
     plot(3:100)         # matched (only one)
}
```

# When Not to use If and Loops

```
DATA sasisgreat
      IF age > 100 THEN
        old = “yes”;
      ELSE
        old= “no”;
RUN;
```

```
for (i in 1:length(age)) {
      if(age[i] > 100) {
        old[i] = “yes”}
      else {
        old[i] = “no”}
}
```

```
old <- rep(NA,length(age))        # BETTER: use []
old[x > 100]     <- “yes”
old[is.na(old)] <- “no”
```

```
old <- x > 100          # BEST: vectorized Boolean
```

# Class Activity