**R Programming – Loops**

Loops are known as “control structures.”

**If, else loops**: tests a condition

Example 1:

if(x >3){

Y <- 10

} else {

Y <- 0

}

Example 2:

Y<- if(x >3){

10

} else {

<- 0

}

If, else can also be run on more than 2 condiontions, for example if(){} else if(){}

If, else can also be run without the else big. It’s saying if x is true, do this, if x is not true, do nothing.

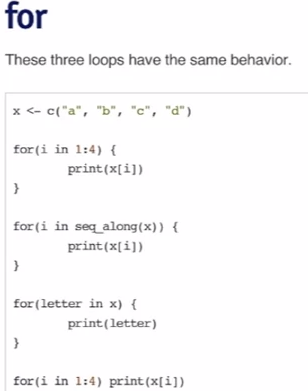
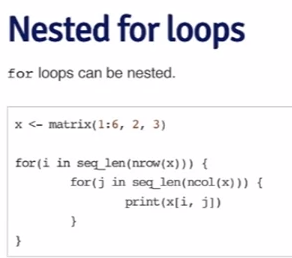
**For loops:** executes a loop “x” amounts of time. It’s an iterative loop

Example:

For (i in 1:x){

Do this

}

**While loops:** executes a loop while a condition is true.

Example:

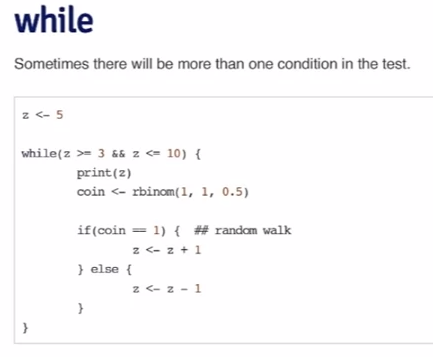
X <-1

While(x <10){

Print(x)

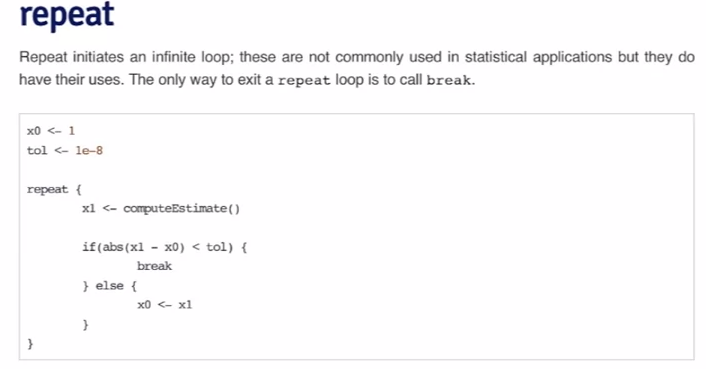
X <- x+1

}



this function flips a coin, if it’s heads (coin=1), z is z+1. If it’s tails (coin=0), z is z-1. For each iteration, z is printed. When z hits the bounds (less than 3 or greater than 10) the loop stops. This will print out a list of random sequential variables from 3 to 10 of undetermined length

**Repeat:** executes an infinite loop. Can be useful in optimization problems when you’re trying to get a value within a tolerance level.

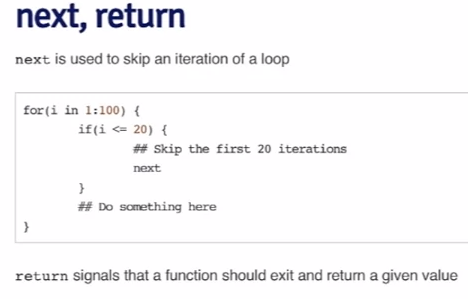


computeEstimate does not exist, but it’s meant to estimate the value of x0. When the difference of x0 and x1 are less than the tolerance of 10^-8, the loop breaks

Repeats have the danger of running forever, it’s better to use an extremely high limit for loops and then see if the loop converges.

**Break:** breaks the execution of a loop. Very useful for Repeat loops

**Next:** skips an iteration of a loop

­

**Return:** exits a function