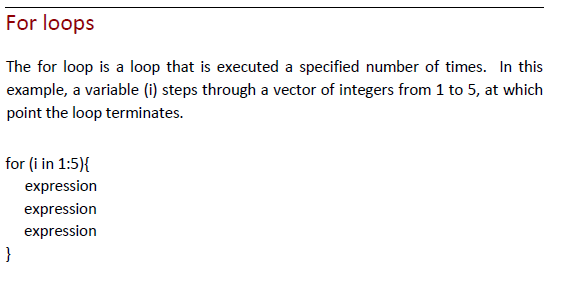
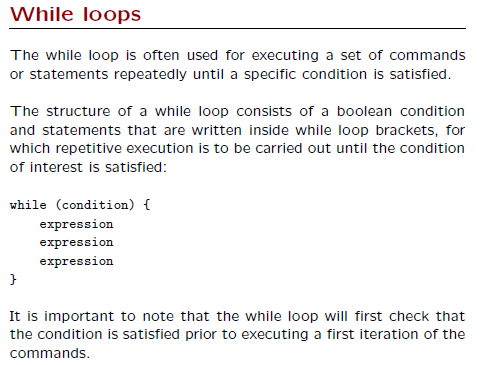
Loops in R

There are three kinds of loop in R. The *for* loop is probably the most popular. Figure 1 provides a quick overview of this loop, its syntax and operation.



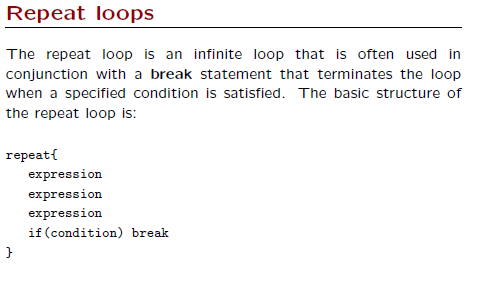
*Figure 1.* The For Loop.

Later in this module, we will examine some code that demonstrates how a *for* loop works. But before we do that, let’s take a quick look at R’s other two loops. Figure 2 illustrates the syntax and logical operation of the *while* looop.



*Figure 2.* The while loop. Adapted from *More Loops, Control Structures, and Bootstrapping*, by K. Rice & T. Thornton. Retrieved from http://faculty.washington.edu/kenrice/rintro/.

And finally, Figure 3 presents the logical operation and basic syntax of the *repeat* loop.



*Figure 3.* The repeat loop. Adapted from *More Loops, Control Structures, and Bootstrapping*, by K. Rice & T. Thornton. Retrieved from http://faculty.washington.edu/kenrice/rintro/.

Now that a concise outline of R’s three loop statements has been presented, we need to look at some actual code, to see how R loops work in the real world. To do that, we are going to load a data file and then step through it. The arsenic data file used in this example has two categorical variables (usedrink and usecook) with values of A, B, C, D, and E. The documentation that came with this data set tells us that each category represents an approximate percentage of well water used for drinking and cooking. It also provides the percentages for each category. However, it would be nice if we could add these percentages directly to the dataframe, explicitly defining each category. Thus we need to loop through the rows, assigning a value to a new variable, based on the category (A, B, C, D, and E) in either usedrink or usecook. The code shows how to do this for usedrink and the same logic applies for usecook.

Each of R’s loop constructs is demonstrated in the listing below, and the numbers correspond to sections after the listing that explain each block of code.

①

setwd (“…”) # Set the working directory to where the file is stored.

arsenic <- read.csv (file = "arsenic.txt", header = TRUE, sep = "\t", as.is = TRUE)

View (arsenic) # View the file to see what’s in it.

②

for(idx in 1:nrow(arsenic)){

arsenic$usedrinkpct[idx] <- "NA" # Assign a default to the variable.

if (arsenic$usedrink[idx] == "A") {arsenic$usedrinkpct[idx] <- "< 1/4"}

if (arsenic$usedrink[idx] == "B") {arsenic$usedrinkpct[idx] <- "~ 1/4"}

if (arsenic$usedrink[idx] == "C") {arsenic$usedrinkpct[idx] <- "~ 1/2"}

if (arsenic$usedrink[idx] == "D") {arsenic$usedrinkpct[idx] <- "~ 3/4"}

if (arsenic$usedrink[idx] == "E") {arsenic$usedrinkpct[idx] <- "> 3/4"}

}

③

idx <- 1

while (idx <= nrow(arsenic)) {

arsenic$usedrinkpct[idx] <- "NA" # Assign a default to the variable.

if (arsenic$usedrink[idx] == "A") {arsenic$usedrinkpct[idx] <- "< 1/4"}

if (arsenic$usedrink[idx] == "B") {arsenic$usedrinkpct[idx] <- "~ 1/4"}

# ... the other if statements.

idx <- idx + 1

}

④

idx <- 1

repeat {

arsenic$usedrinkpct[idx] <- "NA" # Assign a default to the variable.

if (arsenic$usedrink[idx] == "A") {arsenic$usedrinkpct[idx] <- "< 1/4"}

if (arsenic$usedrink[idx] == "B") {arsenic$usedrinkpct[idx] <- "~ 1/4"}

# ... the other if statements.

if (idx >= nrow(arsenic)) { # Break out of loop at last row.

break

}

idx <- idx + 1

}

Explanation of Code Segments

①

The arsenic data set contains data about arsenic levels in drinking water and toenails in New Hampshire. The values for the categorical variables usedrink and usecook are A, B, C, D, and E. As such, they represent an approximate percentage of well water used for drinking and cooking. The percentages for each code are as follows: A: < 1/4, B: ~ 1/4, C: ~ 1/2, D: ~ 3/4, E: > 3/4.

In the read.csv() line, the as.is = TRUE parameter instructs the function to read the file as it is and to not convert the variables into factors. The sep = “\t” parameter indicates that a tab character separates the variables. Character strings preceded by a \ are called “escape characters.”

②

R supports for, while, and repeat loops. The for loop executes a block of code a set number of times. At the beginning of this loop, the idx variable steps through a vector of integers from 1 to the number of rows (nrow) in the arsenic dataframe.

For each iteration of the loop, a series of if statements check the value of the usedrink variable. When usedrink equals one of the categories, the usedrinkpct variable is assigned a value – the approximate percentage of well water used for drinking.

At the top of the loop, we first assign a default value to usedrinkpct. In this case, we want to ensure that usedrinkpct has a value, in the unlikely event that all of the if statements fail. This would happen if usedrink had a value other than A, B, C, D, or E.

③

Unlike the for loop, a while loop checks that a logical condition is true before entering the loop. In this example, the while loop would not be entered if the idx variable was already greater than the number of rows (nrow) in the arsenic data frame.

Once in the loop, the initial condition is checked at each iteration of the loop, and execution continues until idx is greater than the number of rows in the arsenic dataframe. The same series of conditional if statements checks the usedrink category and assigns a value to usedrinkpct.

④

The repeat loop, unlike the previous two examples, does not have a logical check at the beginning of its execution. That is, it immediate begins to loop through and execute a block of code until a break statement is encountered. Absent such a statement, the loop continues forever, or until someone manually forces it to stop. Thus one MUST provide a way to break out of a repeat loop at some point.

Additional Resources

DataCamp has an extended loop tutorial that can be found at:

<https://www.datacamp.com/community/tutorials/tutorial-on-loops-in-r#gs.6ADfWng>

Two of the figures in this tutorial were taken from Ken Rice and Tim Thornton’s Introduction to R. The complete set of slides used in their course – including loops – can be found at:

<http://faculty.washington.edu/kenrice/rintro/>