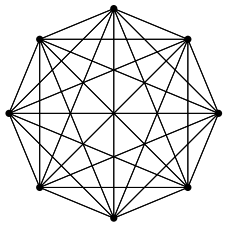
Chapter 18

Input / Output



So far, all of the interaction in an R session has stayed within that R session. This chapter concerns input and output, often abbreviated as I/O by computer scientists, in R. This topic is important because it allows the user to interface with external data sets within an R session. It also allows the user to write the results of a statistical analysis to an external file. This is how R interacts with the world outside of R. The topics considered in this chapter are (a) input and (b) output.

18.1 Input

One of the most versatile functions for reading data into R is the `scan` function. The syntax for the `scan` function is

```
scan(file = "", what = 0, n = -1, sep = "", skip = 0, ...)
```

All 22 of the arguments to `scan` are optional. The five arguments listed above are adequate for most applications.

- The `file` argument tells R to read data values from a filename surrounded by double quotes. The default of `""` indicates that the data values should be read from the keyboard.
- The `what` argument tells R the type of data to be read. The default value of `0` indicates that `numeric` data is to be read. All of the data types encountered thus far are supported by `scan`.
- The `n` argument tells R how many data values to read. The default value of `-1` indicates that R should continue reading from the file until an end-of-file character is encountered.
- The `sep` argument tells R what character delimits the data values. The default of `""` indicates that one or more spaces, to include tabs and end-of-line characters, should delimit the data values. Data files having comma-separated values (`csv`) are also common.
- The `skip` argument tells R the number of lines to skip before beginning to read data values. This argument is useful when there are non-data elements (column headings, metadata, text describing the data set, etc.) in the first few lines of the file that should be ignored.

The simplest application of the `scan` function is to read data from the keyboard. Pressing return once gives a newline; pressing return twice terminates the input. R provides a prompt in the form of an index followed by a colon at the beginning of each line so that the user knows how many values

have been keyed in. An example of the use of the `scan` function to read the ten Florida population values provided at the beginning of Chapter 4 is

```
> x = scan()                          # read data from keyboard into x
1: 529 753 968         1468 1897
6: 2771
7: 4952 6789 9746        12938
11:
Read 10 items
> x                                   # display x
 [1]   529   753   968  1468  1897  2771  4952  6789  9746 12938
```

This approach using `scan` with no arguments is adequate when reading numeric data from the keyboard. The `what` argument can be used for data containing differing modes. For example, the `scan` function can be used to read `numeric` and `character` data values about the Beatles into a list using the following command.

```
> Beatles = scan(what = list(FirstName = "", LastName = "", BirthYear = 0))
1: George Harrison 1943
2: John Lennon 1940
3: Paul McCartney 1942
4: Ringo Starr 1940
5:
Read 4 records
> Beatles                             # display the list named Beatles
$FirstName
[1] "George" "John"   "Paul"   "Ringo"

$LastName
[1] "Harrison"  "Lennon"    "McCartney" "Starr"

$BirthYear
[1] 1943 1940 1942 1940
```

The argument `FirstName = ""` indicates that the component of the list containing the first name has mode `character`; the argument `BirthYear = 0` indicates that the component of the list containing the birth year has mode `numeric`. When `NULL` is used in the `what` argument, the `scan` function skips the associated field.

In a similar fashion, the `readline` function is used to read a single line of data from the keyboard that is stored as a character string.

```
> y = readline()                      # read line of characters from keyboard
Instruct a wise man and he will be wiser still.
> y                                   # display y
[1] "Instruct a wise man and he will be wiser still."
```

The remaining illustrations in this chapter involve an external file. If you intend to key in the data from your computer, navigate to a directory and use a text editor to type the data into a file. I recommend an editor that works with plain ASCII characters and does not insert (often invisible)