Storing mixed data types

As items of different data types cannot be stored in a single vector, R provides a useful alternative “list” structure, whose elements can each contain values of any data type.

Lists are indexed starting at one, just like vectors, and values are assigned to list elements by specifying them as a comma-separated list of arguments to the built-in `list()` function. For example, you can create a list containing values of each main data type, like this:

```
data <- list( 12L, 3.14, “Mike”, TRUE )
```

The list length and structure type can be revealed using the `length()` function and `typeof()` function as with vectors, and there is an `is.list()` function to establish whether an object is a list.

Like vectors, you can address each individual list element by specifying its index number within `[ ]` square brackets. For example, `data[ 3 ]` to retrieve the string in the list created above.

Unlike vectors, lists are not flexible, which means you cannot assign a value to an index number beyond the list’s current length. You can, however, use the `c()` function to combine an existing list with additional values, or other list, to extend the list length.

Most importantly, you may optionally name each element in a list by specifying key=value pairs as a comma-separated list of arguments to the built-in `list()` function, like this:

```
data <- list( dozen=12L, pi= 3.14, user=“Mike”, flag=TRUE )
```

With a named element you can retrieve its value by specifying the list name and element name separated by the `$` dollar operator. For example, `data$user` to retrieve the string in the list above.

R provides two built-in functions especially for lists that contain key=value pairs. The `names()` function retrieves all the keys in the order they appear in the list. The `unlist()` function returns a vector of all keys and values in order, but the names can be explicitly ignored by including a `use.names=FALSE` argument.

The `sum()` function can be used to total up the numeric values contained in a vector, and the `mean()` function can be used to calculate an average of the numeric values contained in a vector.

A list structure in R is similar to the “associative array” (dictionary) structure found in other programming languages.

Named elements can also be addressed using their index number or their element name within square brackets, such as `data[ 3 ]` and `data[ “user” ]` – but remember that this will retrieve both the key and value, not just the value.

Don’t forget

Named elements can also be addressed using their index number or their element name within square brackets, such as `data[ 3 ]` and `data[ “user” ]` – but remember that this will retrieve both the key and value, not just the value.