R Vector: Create, Modify and Access Vector Elements

Perth Coding Club

Create, Modify and Access Vector Elements

In this article, you'll learn about vector in R programming. You'll learn to create them, access their elements using different methods, and modify them in your program.

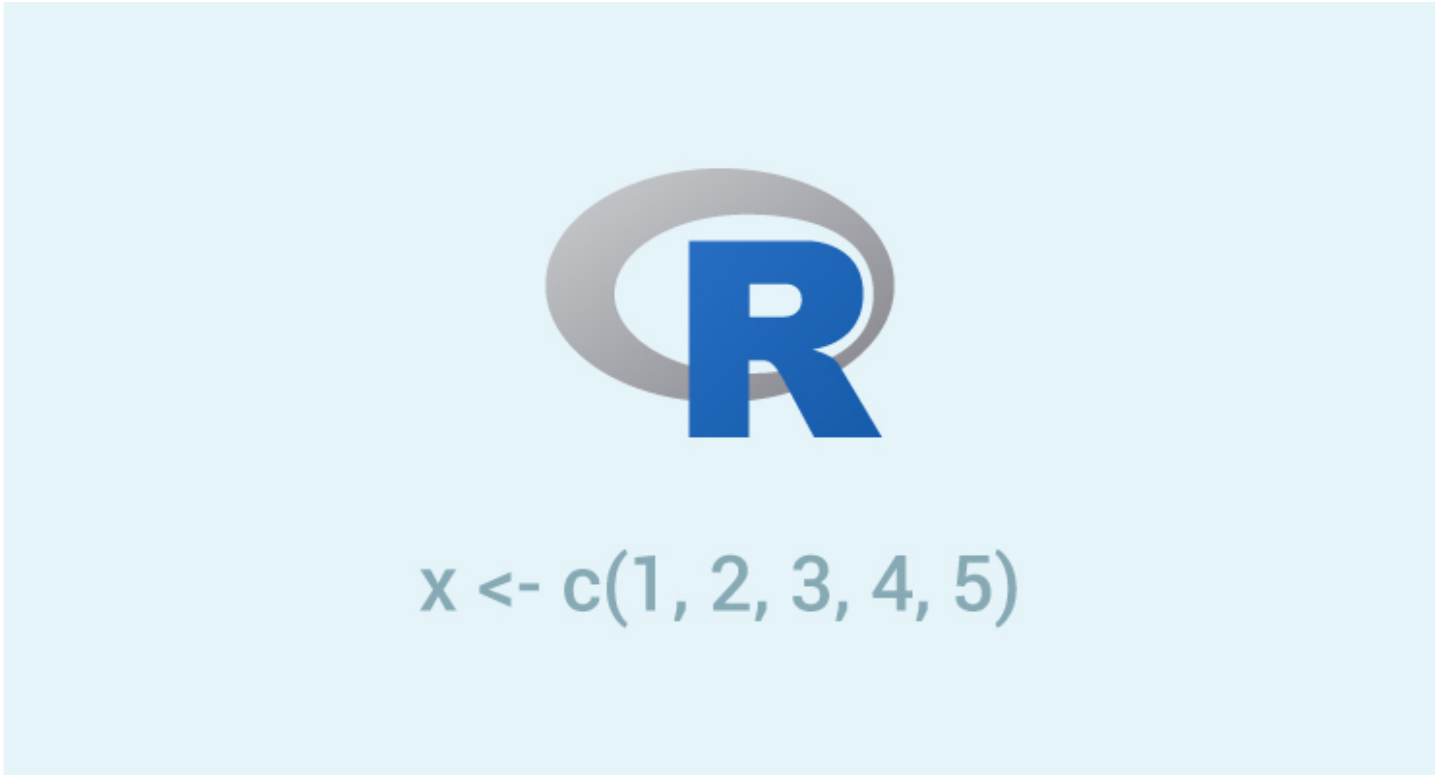
Vector is a basic data structure in R. It contains element of the same type. The data types can be logical, integer, double, character, complex or raw.

A vector's type can be checked with the `typeof()` function.

Another important property of a vector is its length. This is the number of elements in the vector and can be checked with the function `length()`.

How to Create Vector in R?

Vectors are generally created using the `c()` function.



Since, a vector must have elements of the same type, this function will try and coerce elements to the same type, if they are different.

Coercion is from lower to higher types from logical to integer to double to character.

```
> x <- c(1, 5, 4, 9, 0)
> typeof(x)
[1] "double"
> length(x)
[1] 5
> x <- c(1, 5.4, TRUE, "hello")
> x
[1] "1"     "5.4"   "TRUE"  "hello"
> typeof(x)
[1] "character"
```

If we want to create a vector of consecutive numbers, the `:` operator is very helpful.

Example 1: Creating a vector using : operator

```
> x <- 1:7; x
[1] 1 2 3 4 5 6 7
> y <- 2:-2; y
[1]  2  1  0 -1 -2
```

More complex sequences can be created using the `seq()` function, like defining number of points in an interval, or the step size.

Example 2: Creating a vector using seq() function

```
> seq(1, 3, by=0.2)         # specify step size
[1] 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0
> seq(1, 5, length.out=4)    # specify length of the vector
[1] 1.000000 2.333333 3.666667 5.000000
```

How to access Elements of a Vector?

Elements of a vector can be accessed using vector indexing. The vector used for indexing can be logical, integer or character vector.

Using integer vector as index

Vector index in R starts from 1, unlike most programming languages where index start from 0.

We can use a vector of integers as index to access specific elements.

We can also use negative integers to return all elements except that those specified.

But we cannot mix positive and negative integers while indexing and real numbers, if used, are truncated to integers.

```
> x
[1]  0  2  4  6  8 10
> x[3]          # access 3rd element
[1] 4
> x[c(2, 4)]    # access 2nd and 4th element
[1] 2 6
> x[-1]         # access all but 1st element
[1]  2  4  6  8 10
> x[c(2, -4)]   # cannot mix positive and negative integers
Error in x[c(2, -4)] : only 0's may be mixed with negative subscripts
> x[c(2.4, 3.54)]   # real numbers are truncated to integers
[1] 2 4
```

Using logical vector as index

When we use a logical vector for indexing, the position where the logical vector is `TRUE` is returned.

This useful feature helps us in filtering of vector as shown below.

```
> x[c(TRUE, FALSE, FALSE, TRUE)]
[1] -3  3
> x[x < 0]  # filtering vectors based on conditions
[1] -3 -1
> x[x > 0]
[1] 3
```

In the above example, the expression `x>0` will yield a logical vector `(FALSE, FALSE, FALSE, TRUE)` which is then used for indexing.

Using character vector as index

This type of indexing is useful when dealing with named vectors. We can name each elements of a vector.

```
> x <- c("first"=3, "second"=0, "third"=9)
> names(x)
[1] "first"  "second" "third"
> x["second"]
second
0
> x[c("first", "third")]
first third
3     9
```

How to modify a vector in R?

We can modify a vector using the assignment operator.

We can use the techniques discussed above to access specific elements and modify them.

If we want to truncate the elements, we can use reassignments.

```
> x
[1] -3 -2 -1  0  1  2
> x[2] <- 0; x        # modify 2nd element
[1] -3  0 -1  0  1  2
> x[x<0] <- 5; x   # modify elements less than 0
[1] 5 0 5 0 1 2
> x <- x[1:4]; x      # truncate x to first 4 elements
[1] 5 0 5 0
```

How to delete a Vector?

We can delete a vector by simply assigning a `NULL` to it.

```
> x
[1] -3 -2 -1  0  1  2
> x <- NULL
> x
NULL
> x[4]
NULL
```