



Data Structures in R

Part of Future Connect Media's IT Course





In R, data structures are fundamental for organizing, storing, and manipulating data. Here are some common data structures in R:

- Vectors
- Lists
- Matrices
- Data Frames
- Factors
- Arrays
- Time Series
- Environments





Vectors: A basic data structure containing elements of the same data type, like numbers or strings.

To combine the list of items to a vector, use **c()** and separate the items by a comma.

```
# Vector of strings
fruits <- c("banana", "apple", "orange")
# Print fruits
fruits</pre>
```



```
# Vector with numerical decimals in a sequence
numbers1 <- 1.5:6.5
numbers1</pre>
```

Vector with numerical decimals in a sequence where the last element is not used

numbers2 <- 1.5:6.3

numbers2

Result:

[1] 1.5 2.5 3.5 4.5 5.5 6.5

[1] 1.5 2.5 3.5 4.5 5.5









Lists: Versatile containers that can hold elements of different data types and even other data structures.

• To Create a list, use **list()** function:

```
# List of strings
thislist <- list("apple", "banana", "cherry")
# Print the list
thislist</pre>
```



Matrices: 2-dimensional data structures with rows and columns. A matrix can be created with **matrix()** function. Specify the **nrow** and **ncol** parameters to get the amount of rows and columns:

Example:

```
# Create a matrix
thismatrix <- matrix(c(1,2,3,4,5,6), nrow = 3, ncol = 2)
# Print the matrix
thismatrix</pre>
```

Note: Remember the c() function is used to concantenate items together.



Data Frames: Table-like structures commonly used in data analysis, where columns can be of different data types. While the first column can be **character**, the second and third can be **numeric** or **logical**. However, each columns should have the same type of data.

Use the **data.frame()** function to create a data frame:

```
# Create a data frame
Data_Frame <- data.frame (
   Training = c("Strength", "Stamina", "Other"),
   Pulse = c(100, 150, 120),
   Duration = c(60, 30, 45)
)

# Print the data frame
Data_Frame</pre>
```



Data Summarize: Use the **summary()** function to summarize the data from Data Frame:

Example:

```
Data_Frame <- data.frame (
    Training = c("Strength", "Stamina", "Other"),
    Pulse = c(100, 150, 120),
    Duration = c(60, 30, 45)
)</pre>
```

Data_Frame

summary(Data_Frame)

• You will learn more about the summary() function in the statistical part of the R Session.



Factors: Categorical data structures used for representing nominal or ordinal data. To create a factor, use the **factor()** function and add a vector as argument:

Example:

```
my_factor <- factor(c("Low", "Medium", "High"), levels = c("Low", "Medium", "High"))
```

```
# Create a factor
music_genre <- factor(c("Jazz", "Rock", "Classic", "Classic", "Pop", "Jazz", "Rock", "Jazz"))
# Print the factor
music_genre</pre>
```



Time Series: Data structures specialized for time-based data analysis.

Example:

```
my_time_series <- ts(c(5, 10, 15, 20), start = 2020, frequency = 4)
```

Environments: Data structures used to store variables and functions, often used for scoping in R.

```
my_environment <- new.env()
assign("x", 5, envir = my_environment)</pre>
```



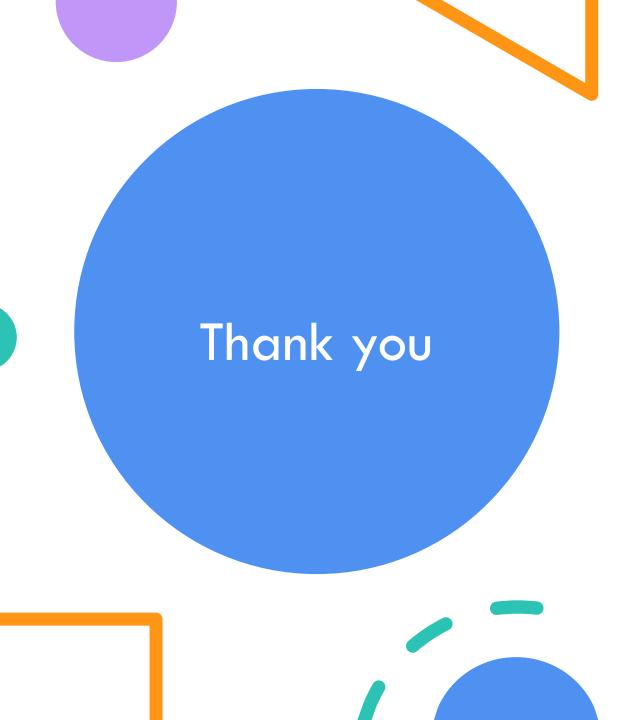
Arrays: Multi-dimensional data structures that can have more than two dimensions. We can use the **array()** function to create, and **dim** parameter to specify the dimensions:

```
# An array with one dimension with values ranging from 1 to 24 thisarray <- c(1:24) thisarray

# An array with more than one dimension

multiarray <- array(thisarray, dim = c(4, 3, 2))

multiarray
```





Future Connect Training Institute

Website: https://www.fctraining.co.uk/