2. What are the different data objects in R? and write syntax and example for each and every object

ANS:

1. Vector: A vector is a one-dimensional array that can store multiple values of the same data type. The c() function can be used to create a vector.

For example: **my\_vector <- c(1, 2, 3, 4, 5)**

1. Matrix: A matrix is a two-dimensional array that can store multiple values of the same data type. The matrix() function can be used to create a matrix.

For example: **my\_matrix <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2, ncol = 3)**

1. Array: An array is a multi-dimensional array that can store multiple values of the same data type. The array() function can be used to create an array.

For example: **my\_array <- array(c(1, 2, 3, 4, 5, 6), dim = c(2, 3, 1))**

1. Data frame: A data frame is a table-like data structure that can store multiple values of different data types. The data.frame() function can be used to create a data frame.

For example: **my\_data\_frame <- data.frame(name = c("John", "Mary", "Bob"), age = c(25, 30, 35), salary = c(50000, 55000, 60000))**

1. List: A list is an ordered collection of items that can store multiple values of different data types. The list() function can be used to create a list.

For example: **my\_list <- list(name = "John", age = 25, salary = 50000, hobbies = c("reading", "traveling"))**

1. Factor: A factor is a data object that represents categorical variables. It can store multiple values of the same data type. The factor() function can be used to create a factor.

For example: **my\_factor <- factor(c("red", "blue", "green", "blue", "red"))**

1. Logical: A logical value is a data object that can store one of two values: TRUE or FALSE. This data type is useful for storing Boolean values.

Example: **my\_logical <- c(T, F, T, T, F)**

6. write difference between break and next also write examples for both

ANS:

break is used to exit a loop prematurely. When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop. Here's an example:

**for (i in 1:10) { if (i == 5) { break } print(i) }**

next is used to skip an iteration of a loop. When a next statement is encountered inside a loop, the current iteration of the loop is skipped and the program control moves on to the next iteration. Here's an example:

**for (i in 1:10) { if (i %% 2 == 0) { next } print(i) }**

In summary, break statement is used to exit the loop completely, while next statement is used to skip the current iteration of the loop and move on to the next iteration.