Week_5_lab

Joseph

2024-07-07

```
#1.Lists
# Lists are the R objects which contain elements of different types like - numbers, strings, vectors an
# A list can also contain a matrix or a function as its elements. List is created using list() function
# # - Creating a list: list(element1, element2, ...)
student_list <- list(name = "John", age = 20, grades = c(85, 90, 95))
# - Accessing list elements: list_name$element_name or list_name[[index]]
print(student_list$name)
```

```
## [1] "John"
```

```
print(student_list[[2]])
```

```
## [1] 20
```

```
# - Modifying list elements: list_name$element_name <- new_value or list_name[[index]] <- new_value
student_list$age <- 21
print(student_list$age)
```

```
## [1] 21
```

```
#4. Data Frames:
# Data Frames are data displayed in a format as a table.
# Data Frames can have different types of data inside it.
#While the first column can be character, the second and third can be numeric or logical. However, each

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

#  Creating a data frame: data.frame(column1, column2, ...)
student_data <- data.frame(name = c("Alice", "Bob", "Charlie"), age = c(18, 19, 20))

# Accessing data frame columns: data_frame_name$column_name or data_frame_name[["column_name"]]
print(student_data$name)
```

```
## [1] "Alice"   "Bob"     "Charlie"
```

```
print(student_data[["age"]])
```

```
## [1] 18 19 20
```

```
# Adding a new column to a data frame: data_frame_name$new_column_name <- new_column_data
student_data$new_column <- c(90, 85, 95)
print(student_data)
```

```
##      name age new_column
## 1   Alice  18         90
## 2     Bob  19         85
## 3 Charlie  20         95
```

```
#5. Arrays:
# Arrays are the R data objects which can store data in more than two dimensions. For example
# If we create an array of dimension (2, 3, 4) then it creates 4 rectangular matrices each with 2 rows
# An array is created using the array() function.
#It takes vectors as input and uses the values in the dim parameter to create an array.
# Creating an array: array(data, dim = c(rows, columns))
student_array <- array(c(1, 2, 3, 4, 5, 6), dim = c(2, 3))
# Printing the array
print(student_array)
```

```
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
```

```
# Accessing array elements: array_name[row_index, column_index]
print(student_array[1, 2])
```

```
## [1] 3
```

```
# 2. Tuples:
# R tuple is basically an entity containing items belonging to different data types. The integral value
student_tuple <- tuple(name = "Mary", age = 22, Gender = "Female")

print(student_tuple)
```

```
## (name = "Mary", age = 22, Gender = "Female")
```

```
print(student_tuple$name)
```

```
## [1] "Mary"
```

```
#Accessing tuple elements: tuple_name$element_name or tuple_name[[index]]
print(student_tuple[[2]])
```

```
## [1] 22
```

```
print(student_tuple[2])
```

```
## (age = 22)
```

```r
print(student_tuple$Gender)
```

```
## [1] "Female"
```

```r
#3. Sets:
#  A set in mathematics is defined as the collection of unique elements and the
# order of the elements does not matter.

# Creating a set: set(element1, element2, ...)
student_set <- set("Alice", "Bob", "Charlie")

# Adding elements to a set: set_name <- union(set_name, new_element)
student_set <- union(student_set, "Dave")
# Printing the sets
print(student_set)
```

```
## [[1]]
## [1] "Alice"
##
## [[2]]
## [1] "Bob"
##
## [[3]]
## [1] "Charlie"
##
## [[4]]
## [1] "Dave"
```