Date operations:

R programming language provides several [functions](https://www.geeksforgeeks.org/functions-in-r-programming/) that deal with date and time. These functions are used to format and convert the date from one form to another form. R provides a format function that accepts the date objects and also a format parameter that allows us to specify the format of the date we needed. R provides various format specifiers which are mentioned below in Table-

| **Specifier** | **Description** |
| --- | --- |
| %a | Abbreviated weekday |
| %A | Full weekday |
| %b | Abbreviated month |
| %B | Full month |
| %C | Century |
| %y | Year without century |
| %Y | Year with century |
| %d | Day of month (01-31) |
| %j | Day in Year (001-366) |
| %m | Month of year (01-12) |
| %D | Data in %m/%d/%y format |
| %u | Weekday (01-07) Starts on Monday |
| # today date  date<-Sys.Date()  # abbreviated Day  format(date,format="%a")    # full Day  format(date,format="%A")    # weekday  format(date,format="%u") | | |

**Output**

[1] "Sat"  
  
[1] "Saturday"  
  
[1] "6"

# today date

date<-Sys.Date()

# default format yyyy-mm-dd

date

# day in month

format(date,format="%d")

# month in year

format(date,format="%m")

# abbreviated month

format(date,format="%b")

# full month

format(date,format="%B")

# Date

format(date,format="%D")

format(date,format="%d-%b-%y")

**Output**

[1] "2022-04-02"  
[1] "02"  
[1] "04"  
[1] "Apr"  
[1] "April"  
[1] "04/02/22"  
[1] "02-Apr-22"

|  |
| --- |
| # today date  date<-Sys.Date()    # year without century  format(date,format="%y")    # year with century  format(date,format="%Y")    # century  format(date,format="%C") |

**Output**

[1] "22"  
[1] "2022"  
[1] "20"

String parsing:

# R program to split a string

# Given String

gfg < - "naga Venkata lakshmi"

# Using strsplit() method

answer < - strsplit(gfg, " ")

print(answer)

# R program to split a string

# Given String

gfg <- "nagavenkatadurga"

# Using strsplit() method

answer <- strsplit(gfg, split = "[0-9]+")

print(answer)

Exercise:11

Introduction Dirty data problems:

Dirty data can be a result of various factors, including human error, data entry mistakes, software bugs, hardware malfunctions, or problems during data migration and integration

Missing values are represented in R by the NA symbol. NA is a special value whose properties are different from other values. NA is one of the very few reserved words in R

|  |
| --- |
| x<- c(**NA**, 3, 4, **NA**, **NA**, **NA**)  is.na(x) |

**Output:**

[1] TRUE FALSE FALSE TRUE TRUE TRUE

|  |
| --- |
| x <- c(1, 2, **NA**, 3, **NA**, 4)  d <- is.na(x)  x[! d] |

**Output:**

[1] 1 2 3 4

|  |
| --- |
| x <- c(1, 2, 0 / 0, 3, **NA**, 4, 0 / 0)  x  x[! is.na(x)] |

**Output:**

[1] 1 2 NaN 3 NA 4 NaN  
  
[1] 1 2 3 4

|  |
| --- |
| # Create a data frame with 5 rows and 3 columns  data <- data.frame(    A = c(1, 2, **NA**, 4, 5),    B = c(**NA**, 2, 3, **NA**, 5),    C = c(1, 2, 3, **NA**, **NA**)  )    # View the resulting data frame  data |

**Output:**

A B C  
1 1 NA 1  
2 2 2 2  
3 NA 3 3  
4 4 NA NA  
5 5 5 NA

**Find all the missing values in the data**

|  |
| --- |
| # Finding missing values in data.  sum(is.na(data)) |

**Output:**

[1] 5

**Find all the missing values in the columns**

* R

|  |
| --- |
| # Finding missing values column wise  colSums(is.na(data)) |

**Output:**

A B C   
1 2 2

**Visualization of missing values of a dataset:**

|  |
| --- |
| # Install and load the 'visdat' package  install.packages("visdat")  library(visdat)    # Create a data frame with missing values  data <- data.frame(    A = c(1, **NA**, 3, **NA**, 5),    B = c(**NA**, 2, **NA**, 4, **NA**),    C = c(1, 2, 3, **NA**, **NA**)  )    # Plot the missing value diagram  vis\_miss(data) |

**Output:**

