# Checking if you have the latest version of R

# If you dont have the latest version, get the latest one

# installing/loading the package:

if(!require(installr)) {

install.packages("installr")

require(installr)

} #load / install+load installr

updateR()

update.packages(ask = FALSE)

################################ Integer and Numeric Data Types #####################

# Creating an integer type

int1 <- 123L

class(int1)

exists("int1")

#Creating a numeric type

num1 <- 123.12

class(num1)

#Convert a numeric data type to integer and integer to numeric

as.integer(123.1243434)

as.integer(123.67)

as.numeric(123)

# A quick look at round and signif

round(123.12243343,2)

round(123.112)

?round()

signif(123.1243434)

signif(100.2)

signif(123.923,4)

#Ceiling,floor and truncate

ceiling(123.123)

ceiling(123)

floor(123.923)

trunc(123.923)

ceiling(123.123)

floor(123.123)

class(20)

class(20L)

# Operations on Integers and Numeric types

int1+20

num1+20

int1+20L

# Addition with literals

class(int1+20)

class(num1+20)

class(int1+20L)

# Integers are subsets of numeric.

# All integers are numeric but not vice versa

#Checking an object if they are integer or numeric types

is.numeric(int1)

is.integer(int1)

is.numeric(num1)

is.integer(num1)

# Type Conversions

as.integer(num1)

as.numeric(int1)

class(num1)

class(int1)

class(as.integer(num1))

class(as.numeric(int1))

# Inf and NaN

1/0

class(1/0)

0/0

class(0/0)

1/(1/0)

1/(0/0)

################################ Logical Data Types #####################

# Creating a logical type

log1 <- TRUE

log2 <- FALSE

class(log1)

class(log2)

# R is case-senstive and hence logical TRUE and FALSE should be in Caps

# The easiest way to represent a TRUE or FALSE is to use their first letter

T

F

# Logical to integer and numeric

log1 \* 10

as.integer(log1)

as.integer(log2)

as.numeric(log1)

as.numeric(log2)

# Integer and numeric to logical

as.logical(int1)

as.logical(num1)

as.logical(100)

as.logical(-100)

as.logical(0)

class(as.logical(100))

as.integer(as.logical(100))

################################ Character Data Types #####################

#Creating a character type

char1<-"VISA"

# converting character to other types

as.integer(char1)

as.numeric(char1)

as.logical(char1)

a <- as.integer(char1)

b <- as.numeric(char1)

c <- as.logical(char1)

class(a)

class(b)

class(c)

as.numeric("VISA")

as.numeric("char1")

as.numeric(char1)

as.numeric("1234.56")

as.integer("1234.56")

as.logical("1234")

as.numeric("12-34")

as.integer("123")

as.logical("TRUE")

as.character(TRUE)

as.logical("true")

# The NA object is available for all integer /numeric / logical data types

class(NA)

class(NA\_integer\_)

class(NA\_real\_)

# converting other data types to character

as.character(int1)

as.character(num1)

as.character(log1)

################################ Factor Data Types #####################

# Why factor types are needed ?

# A character variable - a nominal variable with a lots of possibilities

name <- c("Antony","Bob","Cathy","Diana")

# A character variable with finite levels but order doesn't matter

gender <- c("M","M","F","F")

# A character variable with finite levels but order provides more information

iq<-c("High","Low","Medium","Very High")

#Creating a factor data types

summary(gender)

genderF<-factor(gender)

class(genderF)

summary(genderF)

genderF

str(genderF)

iqF<-factor(iq,ordered = T)

summary(iq)

summary(iqF)

iqF

str(iqF)

iqF2<-factor(iq,

ordered=TRUE,

levels=c("Low","Medium","High","Very High"))

summary(iqF2)

iqF2

str(iqF2)

as.numeric(genderF)

as.logical(genderF)

as.logical(1)

as.character(genderF)

as.integer(iqF2)

as.integer(iqF)

as.integer(iq)

# Thus a nominal variable with finite levels can be converted to integers

# by converting them from factors to integers

unclass(genderF)

class(unclass(genderF))

unclass(iqF2)

unclass(iqF)

################################ Complex Data Types #####################

a <- 1+10i

class(a)

a <- complex(real=1, imaginary = 10)

a

sqrt(-1)

as.complex(-1)

# First convert -1 into complex and then take square root and not vice-versa

sqrt(as.complex(-1))

as.complex(sqrt(-1)) # wont work

Re(a) # Real part

Im(a) # Imaginary part

Mod(a)

sqrt(Re(a)^2 + (Im(a)^2))

Arg(a)

Conj(a)

a <- 1+10i

Conj(a) \* a

a <- complex(imaginary = 0, real = 0)

a

Arg(a)

Mod(a)

a1 <- complex(imaginary = 0, real =1)

a1

Arg(a1)

Mod(a1)

# Note that a corresponds to 0,0 and a1 corresponds to 1,0

# Imagine both these points wrt 0,0 and the unit circle in the Argand plane

b1 <- complex(imaginary = 1, real =0)

b1

Arg(b1)

pi/2 == Arg(b1)

Mod(b1)

c1 <- complex(imaginary = 0, real =-1)

c1

Arg(c1)

pi==Arg(c1)

Mod(c1)

d1 <- complex(imaginary = -1, real =0)

d1

Arg(d1)

-pi/2==Arg(d1)

Mod(d1)

a

as.numeric(a)

as.integer(a)

as.character(a)

as.logical(a)

as.complex(int1)

as.complex(num1)

as.complex(log1)

as.complex(log2)

as.complex(char1)

as.complex(iqF)

################################ Date Data Types #####################

# today -> Sys.Date()

# dates without times and time zones -> as.Date

# dates with time and timezones ->POSIXct and POSIXlt

Sys.Date()

datetext<-"2017-07-03"

class(datetext)

date1 <- as.Date(datetext)

date1

class(date1)

# dates are stored as no of days from Jan 1, 1970

unclass(date1)

class(date1)

typeof(date1)

mode(date1)

#convert a number to date

as.Date(unclass(date1),origin = "1970-1-1")

as.Date(unclass(date1),origin = "2017-1-1")

2017-1970

2064-2017

today=Sys.Date()

weekdays(today)

#Day Options are %d only

# %d - Day of the month

#Month Options are %m,%b and %B

# %m - Month (integer)

# %b - Month (abbreviated)

# %B - Month (full name)

#Year options are %y and %Y

# %y - Year (2 digits)

# %Y - Year (4 digits)

#%Y%m%d is the default format for as.Date

date2<-as.Date("2017/07/03")

date2

date2==date1

as.Date('12/26/2016', format="%m/%d/%Y")

as.Date('26/12/2016', format="%d/%m/%Y")

as.Date('26-12-2016',format="%d-%m-%Y")

as.Date('December 26, 2016',format = '%B%d,%Y')

as.Date('Dec 26, 2016',format = '%b%d,%Y')

as.Date('Dec 26 2016',format = '%b%d%Y')

as.Date('26Dec2016',format='%d%b%Y')

as.Date('26Dec,2016',format='%d%b,%Y')

as.Date('26Dec16',format='%d%b%y')

?POSIXct

as.POSIXct(date1)

as.POSIXlt(date1)

class(as.POSIXct(date1))

class(as.POSIXlt(date1))

unclass(as.POSIXct(date1))

unclass(as.POSIXlt(date1))