Data Types And Data Structures_Assignment_01

Shoyeb_Khan_2021H1540806P

27/09/2021

```
library(stringi)

############################## Exercise 1 ################################

# Find the class and type of 'x' in the following cases:-
#1. x=22   // *Modify the code to create or declare an integer value.*
x <- as.integer(22)
class(x)

## [1] "integer"

typeof(x)

## [1] "integer"

#2. y= 2
#   z=3
#   x=y>z

y <- as.integer(2)
class(y)

## [1] "integer"

typeof(y)

## [1] "integer"


z <- as.integer(3)
class(z)

## [1] "integer"

typeof(z)

## [1] "integer"

x<- y>z
x

## [1] FALSE

class(x)
```

```
## [1] "logical"
```

```
typeof(x)
```

```
## [1] "logical"
```

```
#3. x = 2i
```

```
z <- complex(real = 0 , imaginary = 2)
class(z)
```

```
## [1] "complex"
```

```
typeof(z)
```

```
## [1] "complex"
```

```
#4. x="20-09-2021"
```

```
#To change the format of the date , few adjustments are required. Follo
wing are those :
```

```
lct <- Sys.getlocale("LC_TIME")
Sys.setlocale("LC_TIME", "C")
```

```
## [1] "C"
```

```
p <- as.Date("2021-09-20",format= "%Y-%m-%d")
class(p)
```

```
## [1] "Date"
```

```
typeof(p)
```

```
## [1] "double"
```

```
############################## Exercise 2 ################################
```

```
#1. Find the output when 1+2i is converted to character type
```

```
cmp <- complex(real = 1 , imaginary = 2)
cmp_ch <- as.character(cmp)
cmp_ch
```

```
## [1] "1+2i"
```

```
class(cmp_ch)
```

```
## [1] "character"
```

```
typeof(cmp_ch)
```

```
## [1] "character"
```

```r
#2. Find output when "ProgrammingForAnalytics" is converted to numeric
type

course <- as.character("ProgrammingForAnalytics")
course_num <- as.numeric(course)

## Warning: NAs introduced by coercion

course_num

## [1] NA

#3. Given: x<-0:5, write code to output:
#   [1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  (and)
#   [1] "0" "1" "2" "3" "4" "5"

x_n <- seq(0,5,1)

x_n_loc <- as.logical(x_n)

x_n_loc

## [1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE

x_n_ch <- as.character(x_n)

x_n_ch

## [1] "0" "1" "2" "3" "4" "5"

#4. Given: x<-c("a","b","c")
#   Do all possible coercions to output [1] NA NA NA

x<-c("a","b","c")
x_cor <- as.numeric(x)

## Warning: NAs introduced by coercion

x_cor

## [1] NA NA NA

x_cor <- as.logical(x)
x_cor

## [1] NA NA NA

x_cor <- as.integer(x)

## Warning: NAs introduced by coercion

x_cor
```

```
## [1] NA NA NA

x_cor <- as.complex(x)

## Warning: NAs introduced by coercion

x_cor

## [1] NA NA NA

################################## Exercise 3 ##############################

#Fill the table with your understanding of Data Structures (Atomic vect
or, List, Dataframe, Array, Matrix)

#########################################################################
#|                |   Linear |   2 Dimensional   | N Dimensional |#
#|----------------|----------|-------------------|---------------|#
#|**Homogenous**  |  Vector  |     Matix         |     Array     |#
#|                |          |                   |               |#
#|**Heterogenous**|  List    |    DataFrame      |               |#
#########################################################################

################################ Exercise 4 ##############################

#Create a vector with a sequence of descending numbers from 20 to 0 in
steps of 2.

#(i) Write code to access all except the 2nd to 5th elements.
num <- seq(0,20,2)
num <- sort(num,decreasing = TRUE)

num[-c(2,5)]

## [1] 20 16 14 10  8  6  4  2  0

#(ii) Write code to access all numbers greater than 10 excluding the on
e at 2nd index.
num_grt <- num[num > 10 ]

num_grt <- num_grt[-c(2)]

num_grt

## [1] 20 16 14 12

#(iii) Write code to change values of all elements less than 10 to 0.

num_less <- num < 10
num[num_less] <- 0
```

```
num
```

```
##  [1] 20 18 16 14 12 10  0  0  0  0  0
```

```
############################## Exercise 5 ##############################
#Create a matrix with 2 columns and 4 rows by passing a vector having 4
repetitions of 1 and 2 (i.e., 1,2,1,2,... use rep() #command). Arrange
these elements in a row-wise manner.

col1_vec <- rep(1:2,8)
col1_vec
```

```
##  [1] 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2
```

```
matrix_num <- matrix(col1_vec, nrow = 4  , ncol = 2 , byrow = TRUE)
matrix_num
```

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

```
#(i) Write code to access the 2nd column of this matrix.

matrix_num[,2]
```

```
## [1] 2 2 2 2
```

```
#(ii) Name the columns: "c1", "c2". Name the rows: "r1", "r2", "r3", "r
4".

matrix_num_nam <- matrix(col1_vec, nrow = 4  , ncol = 2 , byrow = TRUE,
                         dimnames = list(c("r1","r2","r3","r4"),c("c1",
"c2")))

matrix_num_nam
```

```
##    c1 c2
## r1  1  2
## r2  1  2
## r3  1  2
## r4  1  2
```

```
#(iii) Write code to access the 2nd row using its row name.
matrix_num_nam["r2",c("c1","c2")]
```

```
## c1 c2
##  1  2
```

```
#(iv) Delete the first row
```

```r
matrix_num_nam <- matrix_num[-1,]
matrix_num_nam
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    1    2
## [3,]    1    2
```

```r
################################## Exercise 6 ##############################

##(i) Create a vector "V" which contains 10 random integer values betwe
en -100 and +100.
v <- sample(-100:100,10)
v
```

```
##  [1]  38  18 -30  -5  13  21 -87  40 -19  31
```

```r
##(ii) Create a two-dimensional 5×5 array "A" comprised of sequence of
even integers greater than 25.
a_even <- array(seq(from = 26, length.out = 25 , by = 2),c(5,5))
a_even
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   26   36   46   56   66
## [2,]   28   38   48   58   68
## [3,]   30   40   50   60   70
## [4,]   32   42   52   62   72
## [5,]   34   44   54   64   74
```

```r
##(iii) Create a list "S" containing sequence of 20 capital letters, st
arting with ‘C’.
S <- LETTERS[match("C",LETTERS) : (match("C",LETTERS)+20)]
S
```

```
##  [1] "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R"
 "S" "T" "U"
## [20] "V" "W"
```

```r
my_vector <- v

my_array <- a_even

my_list <- S

##Create a list named "l" containing all the previously created objects.
  Name them "my_vector", "my_array"  and "my_list" ##respectively.

l <- c(my_vector,my_array,my_list)

l
```

```
##  [1] "38"  "18"  "-30" "-5"  "13"  "21"  "-87" "40"  "-19" "31"  "26
"  "28"
## [13] "30"  "32"  "34"  "36"  "38"  "40"  "42"  "44"  "46"  "48"  "50
"  "52"
## [25] "54"  "56"  "58"  "60"  "62"  "64"  "66"  "68"  "70"  "72"  "74
"  "C"
## [37] "D"   "E"   "F"   "G"   "H"   "I"   "J"   "K"   "L"   "M"   "N"
   "O"
## [49] "P"   "Q"   "R"   "S"   "T"   "U"   "V"   "W"
```

##Without running any R command, answer the following questions pertaining to the exercise :-

##1. How many elements are there in the list?
 # Ans : 56

##2. what is the result of l[[3]]?
 # Ans : -25 (It will be a random value as of running this simulation

 # it is -25.)

##3. How would you access random-th letter in the list element "my_list"?
 # Ans : Using the sample function i.e. sample(mylist,1)

##4. If you convert list l to a vector, what will be the type of it's elements?
 # Ans : character

##5. Can this list be converted to an array? What will be the data type of elements in array?
 # Ans : Yes. data type will be character.

##6. How would you add a new element to this list?
 # Ans : using append function i.e. append(l,"5)

################################ Exercise 7 ################################

#Write a program to create a Data Frame by passing vectors for name (character), age (integer) and vaccinated (logical).

#1. Print the number of rows using dim().
#2. Write code to change the age column into complex data type.
#3. Use "as" function to check if data frame can be coerced into other data types or data structures.

```
data_frame_det <- data.frame("Name:" = c("Michael Scott","Dwight Schrut
                                         e","Pam Morgan Beasley"),
                             "Age:"        = c(43,41,36),
```

```
                              "Vaccinated:" = c(TRUE,TRUE,TRUE))

dim(data_frame_det)

## [1] 3 3

data_frame_det$Age <- c(complex(real = 23 , imaginary = 1),complex(real
                        = 21, imaginary = 2),complex(real = 22 , imagina
                                                              ry = 3))

other_data_ch_ty <- as.character(data_frame_det)

other_data_ch_ty

## [1] "c(\"Michael Scott\", \"Dwight Schrute\", \"Pam Morgan Beasley\")
"
## [2] "c(23+1i, 21+2i, 22+3i)"

## [3] "c(TRUE, TRUE, TRUE)"

other_data_lst_ty <- as.list(data_frame_det)

other_data_lst_ty

## $Name.
## [1] "Michael Scott"      "Dwight Schrute"     "Pam Morgan Beasley"
##
## $Age
## [1] 23+1i 21+2i 22+3i
##
## $Vaccinated
## [1] TRUE TRUE TRUE

other_data_vct_ty <- as.vector(data_frame_det)

other_data_vct_ty

##                Name.   Age Vaccinated
## 1      Michael Scott 23+1i       TRUE
## 2     Dwight Schrute 21+2i       TRUE
## 3 Pam Morgan Beasley 22+3i       TRUE

############################## Exercise 8 ###################################

#Debug the following and run the correct code
#1. num = c(1, 2, 3, 4, 5)
#  name = c("one", "two", "four", "five")
#  df = data.frame(num,name)
#   print(df)
#2. x = c(1, "BITS", 5, 7.2, True, 1+i)
#   print(X)
```

```r
#1. Corrected Code
num = c(1, 2, 3, 4, 5)

name = c("one", "two", "three", "four","five")

df = data.frame(num,name)

print(df)
```

```
##   num  name
## 1   1   one
## 2   2   two
## 3   3 three
## 4   4  four
## 5   5  five
```

```r
#2. Corrected Code
x= c(1, "BITS", 5, 7.2, TRUE, complex(real = 1 , imaginary = 1))
print(x)
```

```
## [1] "1"    "BITS" "5"    "7.2"  "TRUE" "1+1i"
```